《数据挖掘与机器学习》实验报告

题　　目：美国餐厅关联规则挖掘

学　　号：19024075

姓　　名：陶盛皿

1. 实验要求

利用 Apriori 算法进行关联规则挖掘的实现。

2. 实验内容

2.1 综述

利用 Aprioro 算法对部分城市的公民常去的餐厅进行关联规则挖掘。

2.2 数据集描述

该数据记录了从 1996 年 9 月到 1999 年 4 月与 Entree Chicago 餐厅推荐系统的交互。会话文件中的每一行表示用户与系统交互，记录该用户常去的餐厅。

2.3 数据描述

TXT 总数：8，包含城市：Atlanta，Boston，Chicago，Los Angeles，New Orleans，New York，San Francisco，Washington DC。

总计：4152 条记录。

2.4 结论

总体的支持度 threshold 很小，超过 0.5 的只有一条记录，故设置了 0.2，才能涵盖 2-频繁项集，可见 8 个城市餐厅总支持度（即跨城市）并不显著。但仍能通过总体数据的 Top 支持度的 1-频繁项集，得出八个城市居民常去的 Top3 的餐厅。

205 Extraordinary Service
75 Extraordinary Food
53 Extraordinary Decor

总体数据结果，如下：

All

id	support	itemsets
8	0.521435453	frozenset({205})
2	0.473265896	frozenset({75})
1	0.342244701	frozenset({53})
13	0.342244701	frozenset({75, 205})
0	0.302745665	frozenset({52})
10	0.297447013	frozenset({250})
6	0.282996146	frozenset({192})
11	0.280105973	frozenset({253})
3	0.270712909	frozenset({76})
12	0.265895954	frozenset({53, 205})
5	0.253612717	frozenset({191})
7	0.236271676	frozenset({204})
4	0.231454721	frozenset({174})
14	0.228082852	frozenset({192, 191})
9	0.208815029	frozenset({231})

由于总体支持度普遍较小，于是对八个城市分别进行了关联规则挖掘。minSupport 的阈值设置统一设置为 0.5。

2.4.1 亚特兰特

Atalanta

id	support	itemsets
1	0.842105263	frozenset({85})
6	0.744360902	frozenset({124})
11	0.646616541	frozenset({124, 85})
4	0.616541353	frozenset({101})
5	0.590225564	frozenset({123})
2	0.582706767	frozenset({90})
3	0.582706767	frozenset({91})
12	0.582706767	frozenset({90, 91})
0	0.533834586	frozenset({39})
9	0.511278195	frozenset({85, 101})
7	0.507518797	frozenset({90, 85})
8	0.507518797	frozenset({91, 85})
13	0.507518797	frozenset({90, 91, 85})
10	0.503759398	frozenset({123, 85})

结论：Top3 的包含 1-频繁项集以及 2 频繁项集，故 85 号和 124 号是亚特兰特居民常去的餐厅。

即：

85	For the Young and Young at Heart
124	Italian

2.4.2 波士顿

Boston

id	support	itemsets
0	0.718535469	frozenset({136})
1	0.542334096	frozenset({137})

结论：波士顿居民最常去的餐厅是 136 和 137，其中两者的支持度差异显著，故认为波士顿最常去的餐厅是 136.

即：

136	Long Drive

2.4.3 芝加哥

Chicago

id	support	itemsets
0	0.617777778	frozenset({50})
1	0.54962963	frozenset({129})
2	0.521481481	frozenset({157})

即：

50	Fair Decor
129	Korean
157	Open on Sundays

2.4.4 洛杉矶

LA

id	support	itemsets
1	0.764573991	frozenset({138})
2	0.67264574	frozenset({140})
0	0.556053812	frozenset({117})
3	0.547085202	frozenset({138, 140})

结论：包含 2-频繁项集，138 和 140，由于阈值区分度明显，故认

为 138 和 140 是洛杉矶居民最常去的餐厅，并且具有关联性。

antecedents	consequents	antecedent support	consequent support	support	confidence	lift	leverage	conviction
138	140	0.76	0.67	0.55	0.72	1.06	0.03	1.15
140	138	0.67	0.76	0.55	0.81	1.06	0.03	1.26

即：

138 Malaysian
140 Menus in Braille

2.4.5 新奥尔良市

New Orleans

	support	itemsets
0	0.788343558	frozenset({63})
1	0.788343558	frozenset({64})
2	0.788343558	frozenset({64, 63})

即：

63 Down-Home
64 Down-Home Creole

2.4.6 纽约市

New York

	support	itemsets
0	0.509591326	frozenset({48})

2.4.7 旧金山

San Francisco

	support	itemsets
1	0.607748184	frozenset({127})
0	0.585956416	frozenset({106})
2	0.581113801	frozenset({128})

2.4.8 华盛顿

Washington DC

	support	itemsets
2	0.812820513	frozenset({131})
0	0.712820513	frozenset({91})
5	0.594871795	frozenset({91, 131})
3	0.566666667	frozenset({132})
4	0.551282051	frozenset({133})
1	0.538461538	frozenset({108})
6	0.530769231	frozenset({131, 132})

3. 算法思想

- Support(A) = 包含 A 的记录数量 / Total 记录
- Conf(A->B) = 包含 A 和 B 的记录数量 / 包含 A 的记录
- Lift(A->B) = Conf(A->B) / Support(A) = 包含 A 和 B 的记录数量 / Total 记录的数量

Apriori 算法核心思想：若一个项集是非频繁项集， 那么它的所有超集都是非频繁项集，对非频繁项集的根节点进行剪枝，使用递归进行遍历数据集。最小支持度手动设定，通过支持度反映频繁项。

4. 代码实现（部分）

```
la = pd.read_csv(".//data//los_angeles.txt", sep='\t', engine = 'python')

la.columns = Name

d = la['idx'].str.split(' ')

D = pd.DataFrame(d)

D. head()
```

```
All = pd.concat([A, B, C, D, E, F, G, H], ignore_index=True)

All.shape
```

```
from mlxtend.frequent_patterns import apriori, association_rules
```

```
from mlxtend.preprocessing import TransactionEncoder

dataA = A['idx'].to_list()

teA = TransactionEncoder()

aa = teA.fit_transform(dataA)

df_a = pd.DataFrame(aa)

resultA1 = apriori(df_a, min_support = 0.5, use_colnames = True)

asscA = association_rules(resultA1, min_threshold = 0.5)

resultA1.to_csv('boston_res.csv')
```