

# 数据挖掘互评作业二：频繁模式与关联规则挖掘

计算机学院 JohnsonGUO

代码仓库地址: <https://github.com/johnsongwx/data-mining-coursework>

## 数据挖掘互评作业二：频繁模式与关联规则挖掘

### 作业题目选择

#### 网页浏览行为关联规则挖掘

1. 数据预处理
2. 频繁模式挖掘
3. 关联规则挖掘，计算其支持度和置信度
4. 规则评价，结果评估
5. 结果可视化

## 作业题目选择

### 网页浏览行为关联规则挖掘

任务: 分析用户在网站上的浏览行为数据，挖掘潜在的跳转规律，为网站提供优化导航结构的建议。

数据集: [UCI ML Repository - Anonymous Microsoft Web Data](#)

关联规则挖掘：

1. 数据预处理: 清洗数据，处理缺失值，提取用户浏览记录。
2. 数据探索性分析: 分析最常被访问的页面、页面访问量分布等。
3. 关联规则挖掘: 使用Apriori算法或FP-growth算法，根据用户浏览记录计算频繁项集和关联规则。
4. 结果评估: 计算关联规则的支持度、置信度和提升度，得出强关联规则。
5. 结果分析与应用: 分析得到的关联规则，为网站提供导航结构优化建议，以提升用户体验。

### 网页浏览行为关联规则挖掘

本作业结构：按照作业要求的流程，首先给出带有备注信息的代码，和必要的文字解读，随后给出运行结果。

## 1. 数据预处理

导入所需的库，包括 `pandas` 用于数据处理，`TransactionEncoder` 和 `apriori` 用于频繁模式挖掘，`association_rules` 用于生成关联规则，以及 `matplotlib.pyplot` 和 `seaborn` 用于数据可视化。

```
1 import pandas as pd
2 from mlxtend.preprocessing import TransactionEncoder
3 from mlxtend.frequent_patterns import apriori, association_rules
4 import matplotlib.pyplot as plt
5 import seaborn as sns
```

根据每一条数据第一个字段，进行分别处理。

“A”表示attribute，需要将后续内容加入属性字典中；

“C”表示一个case的起始，需要继续读入后续的0个以上的vote；

“V”表示一个case中的一个vote。

```
1 with open("anonymous-msweb.data", "r", encoding="utf-8") as f:
2     lines = f.readlines()
3
4     # attr_lines = []
5     attr_dic = {}
6     case_dic = {}
7     vote_list = []
8     # case_vote_list = []
9
10    valid_attr = ["A", "C", "V"]
11    for i, line in enumerate(lines):
12        content = line.strip().split(",")
13
14        attr = content[0]
15        if attr not in valid_attr:
16            continue
17
18        if attr == "A":
19            attr_dic[content[1]] = [content[3], content[4]]
20        elif attr == "C":
21            # c要连着处理之后的v
22            cur_vote = []
23            num = i+1
24            if num >= len(lines):
25                break
26            while lines[num].strip().split(",")[0] == 'v':
27                cur_vote.append(lines[num].strip().split(",")[1])
28                num += 1
29                if num >= len(lines):
30                    break
31            vote_list.append(cur_vote)
32            case_dic[content[1]] = cur_vote[1:]
33        else:
34            continue
```

## 2. 频繁模式挖掘

在开展频繁模式挖掘时，我们首先需要对数据集进行转换。

在使用apriori函数时，输入的数据集需要进行一些预处理才能正确执行。具体来说，需要将数据集转换为适合apriori函数的形式，可以使用TransactionEncoder类来完成转换。

```
1     te = TransactionEncoder()
2     # 将每一行case_vote数据转换为一个[1, 285]的向量，其中285是在全部数据中出现的不同attr的总
    数，如果这行case_vote数据中有某一个attr，那么对应位置为True，否则为False
3     # te_ary总共为[32711,285]，32711为case_dic的总数，也就是所有case的总数
4     te_ary = te.fit_transform(vote_list)
5     df = pd.DataFrame(te_ary, columns=te.columns_)
6
7     # 使用apriori函数进行频繁模式挖掘
8     frequent_itemsets = apriori(df, min_support=0.05, use_colnames=True)
9
10    print(frequent_itemsets)
```

运行结果如下图所示。



frequent\_itemsets\*



	▼ support	◆ itemsets
3	0.33126	frozenset({'1008'})
9	0.28685	frozenset({'1034'})
2	0.25872	frozenset({'1004'})
6	0.16294	frozenset({'1018'})
18	0.16080	frozenset({'1008', '1034'})
5	0.15616	frozenset({'1017'})
4	0.14148	frozenset({'1009'})
0	0.13607	frozenset({'1001'})
8	0.09844	frozenset({'1026'})
1	0.09073	frozenset({'1003'})
15	0.07792	frozenset({'1009', '1008'})
17	0.07306	frozenset({'1018', '1008'})
7	0.06490	frozenset({'1025'})
16	0.06123	frozenset({'1017', '1008'})
13	0.06044	frozenset({'1004', '1008'})
12	0.05943	frozenset({'1001', '1018'})
11	0.05521	frozenset({'1003', '1001'})
10	0.05475	frozenset({'1035'})
14	0.05228	frozenset({'1004', '1034'})



frequent\_itemsets

### 3. 关联规则挖掘，计算其支持度和置信度

```
1 # 导出关联规则
2 rules = association_rules(frequent_itemsets, metric="confidence",
3 min_threshold=0.1)
4 print(rules)
5
6 # 计算支持度和置信度
7 rules['support'] = rules['support'].round(4)
8 rules['confidence'] = rules['confidence'].round(4)
```

运行结果如下图所示。

	÷ antecedents	÷ consequents	÷ antecedent support	÷ consequent support	÷ support	÷ confidence	÷ lift	÷ leverage	÷ conviction	÷ zhangs_metric	÷ chi_square
0	frozenset({'10...	frozenset({'100...	0.09073	0.13607	0.05520	0.60850	0.60837	0.04286	2.20666	0.85385	6193.96933
1	frozenset({'10...	frozenset({'100...	0.13607	0.09073	0.05520	0.40580	0.40567	0.04286	1.53011	0.89866	6193.96933
2	frozenset({'10...	frozenset({'101...	0.13607	0.16294	0.05940	0.43680	0.43654	0.03726	1.48614	0.72567	2827.55741
3	frozenset({'101...	frozenset({'100...	0.16294	0.13607	0.05940	0.36470	0.36455	0.03726	1.35994	0.74896	2827.55741
4	frozenset({'10...	frozenset({'100...	0.25872	0.33126	0.06040	0.23360	0.23346	-0.02527	0.87257	-0.36060	493.01635
5	frozenset({'10...	frozenset({'100...	0.33126	0.25872	0.06040	0.18240	0.18233	-0.02527	0.90671	-0.38467	493.01635
6	frozenset({'10...	frozenset({'103...	0.25872	0.28685	0.05330	0.20600	0.20601	-0.02093	0.89813	-0.34633	364.64573
7	frozenset({'10...	frozenset({'100...	0.28685	0.25872	0.05330	0.18580	0.18581	-0.02093	0.91040	-0.35514	364.64573
8	frozenset({'10...	frozenset({'100...	0.14148	0.33126	0.07790	0.55080	0.55060	0.03106	1.48865	0.46423	1170.68463
9	frozenset({'10...	frozenset({'100...	0.33126	0.14148	0.07790	0.23520	0.23516	0.03106	1.12259	0.59598	1170.68463
10	frozenset({'101...	frozenset({'100...	0.15616	0.33126	0.06120	0.39210	0.39192	0.00950	1.10013	0.18394	100.52096
11	frozenset({'10...	frozenset({'101...	0.33126	0.15616	0.06120	0.18480	0.18475	0.00950	1.03520	0.23211	100.52096
12	frozenset({'101...	frozenset({'100...	0.16294	0.33126	0.07310	0.44840	0.44863	0.01909	1.21237	0.31209	395.90282
13	frozenset({'10...	frozenset({'101...	0.33126	0.16294	0.07310	0.22060	0.22067	0.01909	1.07392	0.39065	395.90282
14	frozenset({'10...	frozenset({'103...	0.33126	0.28685	0.16080	0.48540	0.48541	0.06578	1.38589	0.61172	3123.18220
15	frozenset({'10...	frozenset({'100...	0.28685	0.33126	0.16080	0.56060	0.56058	0.06578	1.52189	0.57362	3123.18220

### 4. 规则评价，结果评估

对规则进行评价，使用Lift、卡方指标；对挖掘结果进行分析。

```
1 # 评价规则：使用Lift和卡方指标
2 rules['lift'] = rules['support'] / rules['antecedent support']
3 rules['chi_square'] = (df.shape[0] * (rules['support'] - rules['antecedent
4 support'] * rules['consequent support']) ** 2) \
5 / (rules['antecedent support'] * rules['consequent
6 support'] * (1 - rules['antecedent support']) * (1 - rules['consequent support']))
7
8 # 对挖掘结果进行分析
9 sorted_rules = rules.sort_values(by='lift', ascending=False)
10
11 print("关联规则挖掘结果：")
12 print(sorted_rules)
```

结果如下图所示。

	÷ antecedents	÷ consequents	÷ antecedent support	÷ consequent support	÷ support	÷ confidence	÷ lift	÷ leverage	÷ conviction	÷ zhangs_metric	÷ chi_square
0	frozenset({'1003'})	frozenset({'1001'})	0.09073	0.13607	0.05520	0.60850	0.60837	0.04286	2.20666	0.85385	6193.96933
15	frozenset({'1034'})	frozenset({'1008'})	0.28685	0.33126	0.16080	0.56060	0.56058	0.06578	1.52189	0.57362	3123.18220
8	frozenset({'1009'})	frozenset({'1008'})	0.14148	0.33126	0.07790	0.55080	0.55060	0.03106	1.48865	0.46423	1170.68463
14	frozenset({'1008'})	frozenset({'1034'})	0.33126	0.28685	0.16080	0.48540	0.48541	0.06578	1.38589	0.61172	3123.18220
12	frozenset({'1018'})	frozenset({'1008'})	0.16294	0.33126	0.07310	0.44840	0.44863	0.01909	1.21237	0.31209	395.90282
2	frozenset({'1001'})	frozenset({'1018'})	0.13607	0.16294	0.05940	0.43680	0.43654	0.03726	1.48614	0.72567	2827.55741
1	frozenset({'1001'})	frozenset({'1003'})	0.13607	0.09073	0.05520	0.40580	0.40567	0.04286	1.53011	0.89866	6193.96933
10	frozenset({'1017'})	frozenset({'1008'})	0.15616	0.33126	0.06120	0.39210	0.39192	0.00950	1.10013	0.18394	100.52096
3	frozenset({'1018'})	frozenset({'1001'})	0.16294	0.13607	0.05940	0.36470	0.36455	0.03726	1.35994	0.74896	2827.55741
9	frozenset({'1008'})	frozenset({'1009'})	0.33126	0.14148	0.07790	0.23520	0.23516	0.03106	1.12259	0.59598	1170.68463
4	frozenset({'1004'})	frozenset({'1008'})	0.25872	0.33126	0.06040	0.23360	0.23346	-0.02527	0.87257	-0.36060	493.01635
13	frozenset({'1008'})	frozenset({'1018'})	0.33126	0.16294	0.07310	0.22060	0.22067	0.01909	1.07392	0.39065	395.90282
6	frozenset({'1004'})	frozenset({'1034'})	0.25872	0.28685	0.05330	0.20600	0.20601	-0.02093	0.89813	-0.34633	364.64573
7	frozenset({'1034'})	frozenset({'1004'})	0.28685	0.25872	0.05330	0.18580	0.18581	-0.02093	0.91040	-0.35514	364.64573
11	frozenset({'1008'})	frozenset({'1017'})	0.33126	0.15616	0.06120	0.18480	0.18475	0.00950	1.03520	0.23211	100.52096
5	frozenset({'1008'})	frozenset({'1004'})	0.33126	0.25872	0.06040	0.18240	0.18233	-0.02527	0.90671	-0.38467	493.01635

## 5. 结果可视化

```

1      # 可视化频繁项集支持度
2      plt.figure(figsize=(10, 6))
3      plt.subplot(1, 2, 1)
4      sns.barplot(x='support', y='itemsets',
5                  data=frequent_itemsets.sort_values(by='support', ascending=False),
6                  orient='h')
7      plt.title('Frequent Itemsets - Support')
8      plt.xlabel('Support')
9      plt.ylabel('Itemsets')
10
11     plt.tight_layout()
12     plt.show()
13
14     # 可视化关联规则评估指标
15     plt.figure(figsize=(8, 6))
16     sns.scatterplot(x='lift', y='confidence', data=rules)
17     plt.title('Association Rules - Lift vs. Confidence')
18     plt.xlabel('Lift')
19     plt.ylabel('Confidence')
20     plt.show()

```

结果如下图所示：

