Darshan UNIVERSITY
योग: कर्मसु कौशलम्

# Data Mining

# Lab - 7 (Part 2)

137 | Vishal Baraiya | 23010101014

## Step 1: Load the Dataset

Load the `Tdata.csv` file and display the first few rows.

In [1]:
```
import pandas as pd
```

In [2]:
```
Tdata = pd.read_csv("Tdata.csv")
Tdata
```

Out[2]:

| | Transaction | bread | butter | coffee | eggs | jam | milk |
|---|---|---|---|---|---|---|---|
| **0** | T1 | 1 | 1 | 0 | 0 | 0 | 1 |
| **1** | T2 | 1 | 1 | 0 | 0 | 1 | 0 |
| **2** | T3 | 1 | 0 | 0 | 1 | 0 | 1 |
| **3** | T4 | 1 | 1 | 0 | 0 | 0 | 1 |
| **4** | T5 | 1 | 0 | 1 | 0 | 0 | 0 |
| **5** | T6 | 0 | 0 | 1 | 1 | 1 | 0 |

## Step 2: Drop the 'Transaction' Column

We're only interested in the items (not the transaction IDs).

In [3]:
```
df_item = Tdata.drop(columns=["Transaction"])
df_item.head()
```

Out[3]:

| | bread | butter | coffee | eggs | jam | milk |
|---|---|---|---|---|---|---|
| **0** | 1 | 1 | 0 | 0 | 0 | 1 |
| **1** | 1 | 1 | 0 | 0 | 1 | 0 |
| **2** | 1 | 0 | 0 | 1 | 0 | 1 |
| **3** | 1 | 1 | 0 | 0 | 0 | 1 |
| **4** | 1 | 0 | 1 | 0 | 0 | 0 |

## Step 3: Count Single Items

See how many transactions include each item.

In [4]:
```
df_item.sum()
```

Out[4]:
```
bread     5
butter    3
coffee    2
eggs      2
jam       2
milk      3
dtype: int64
```

## Step 4: Define Apriori Function

This function finds frequent itemsets of size 1, 2, and 3 with minimum support.

In [5]:
```
from itertools import combinations

def find_frequent_itemsets(df,min_support):
    n = len(df)
    result = []

    for k in [1,2,3]: # for 1-item, 2-item, 3-item
        for items in combinations(df.columns,k):
            mask = df[list(items)].all(axis = 1)
            support = mask.sum()/n
            if support >= min_support:
                result.append((frozenset(items),round(support,2)))

    return result
```

## Step 5: Run Apriori

Set `min_support = 0.6` and display the frequent itemsets.

In [6]:
```
frequent_itemsets = find_frequent_itemsets(df_item,min_support=0.5)
```

```
for itemset, support in frequent_itemsets:
    print(f"{set(itemset)} -> support : {support}")
```

```
{'bread'} -> support : 0.83
{'butter'} -> support : 0.5
{'milk'} -> support : 0.5
{'butter', 'bread'} -> support : 0.5
{'bread', 'milk'} -> support : 0.5
```

## Step 6 Display as a DataFrame

In [7]:
```
result_df = pd.DataFrame(frequent_itemsets, columns=['Itemset','Support'])
result_df
```

Out[7]:

| | Itemset | Support |
|---|---|---|
| 0 | (bread) | 0.83 |
| 1 | (butter) | 0.50 |
| 2 | (milk) | 0.50 |
| 3 | (butter, bread) | 0.50 |
| 4 | (bread, milk) | 0.50 |

In [ ]:
```

```

# Orange Tool : - >Generate Same Frequent Patterns in Orange tools

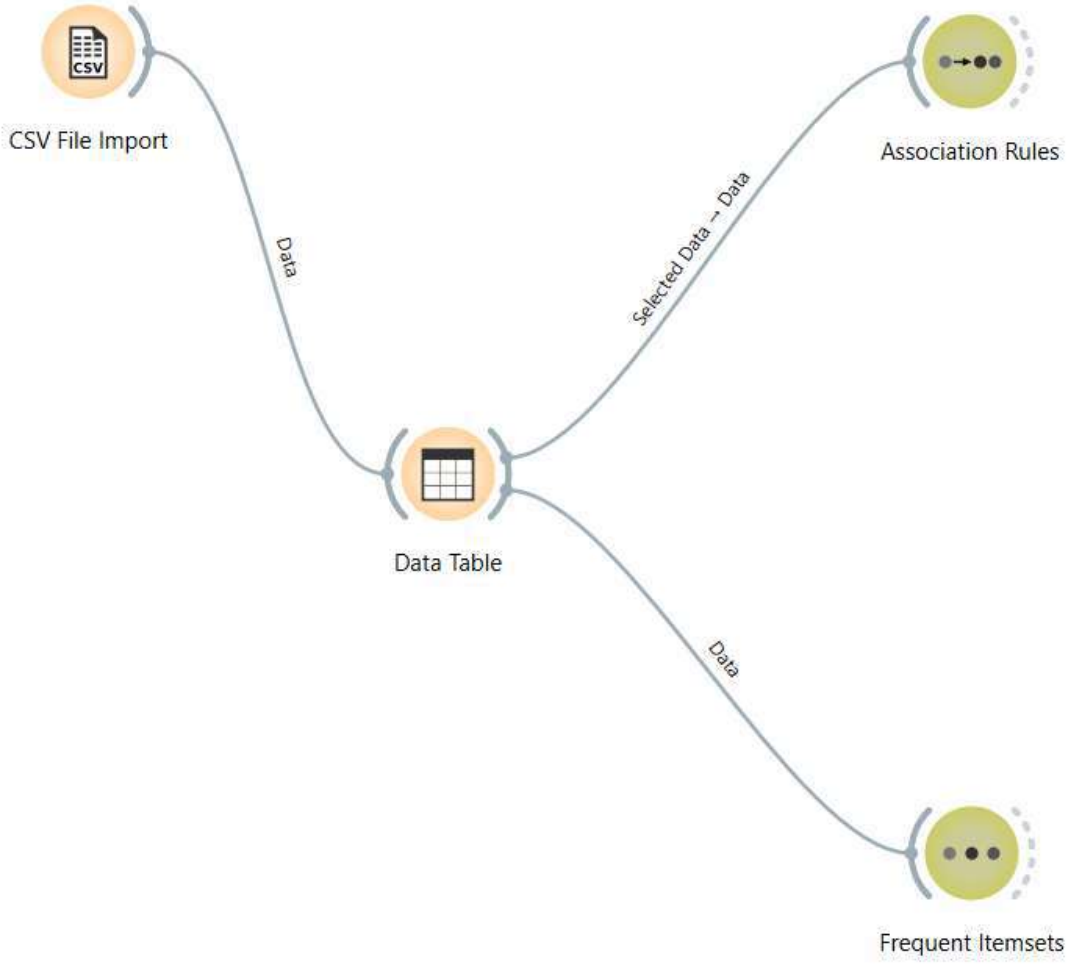
In [ ]:
```

```

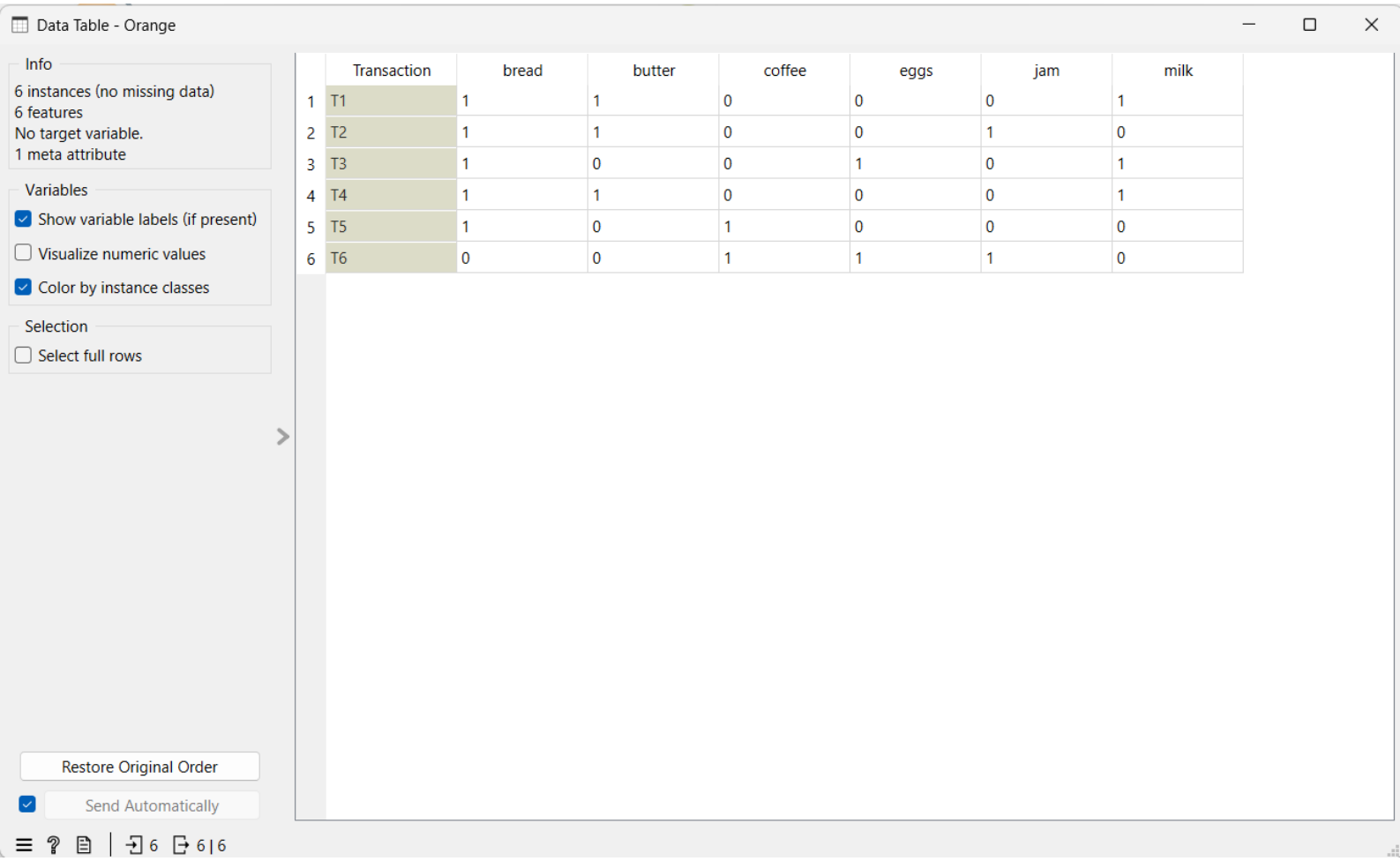
# Extra : - > Define Apriori Function without itertools

In [8]:
```
# def find_frequent_itemsets(df, min_support):
#     n = len(df)
#     result = []

#     columns = list(df.columns)

#     # 1-itemsets
#     for i in range(len(columns)):
#         item1 = columns[i]
#         mask = df[item1] == 1
#         support = mask.sum() / n
#         if support >= min_support:
#             result.append((frozenset([item1]), round(support, 2)))
```

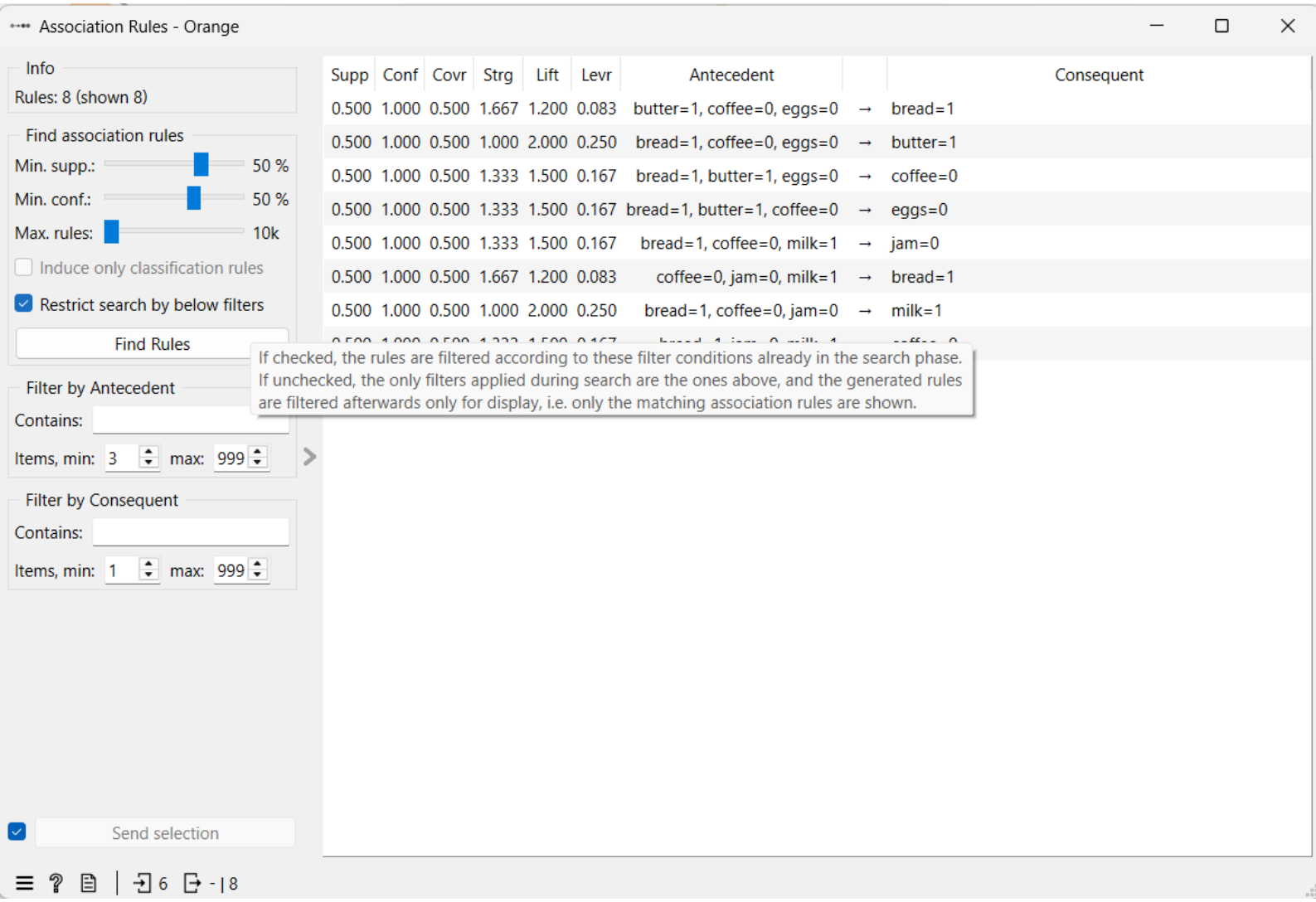
```
#     # 2-itemsets
#     for i in range(len(columns)):
#         for j in range(i + 1, len(columns)):
#             item1 = columns[i]
#             item2 = columns[j]
#             mask = df[item1] & df[item2]
#             support = mask.sum() / n
#             if support >= min_support:
#                 result.append((frozenset([item1, item2]), round(support, 2)))

#     # 3-itemsets
#     for i in range(len(columns)):
#         for j in range(i + 1, len(columns)):
#             for k in range(j + 1, len(columns)):
#                 item1 = columns[i]
#                 item2 = columns[j]
#                 item3 = columns[k]
#                 mask = df[item1] & df[item2] & df[item3]
#                 support = mask.sum() / n
#                 if support >= min_support:
#                     result.append((frozenset([item1, item2, item3]), round(suppor

#     return result
```

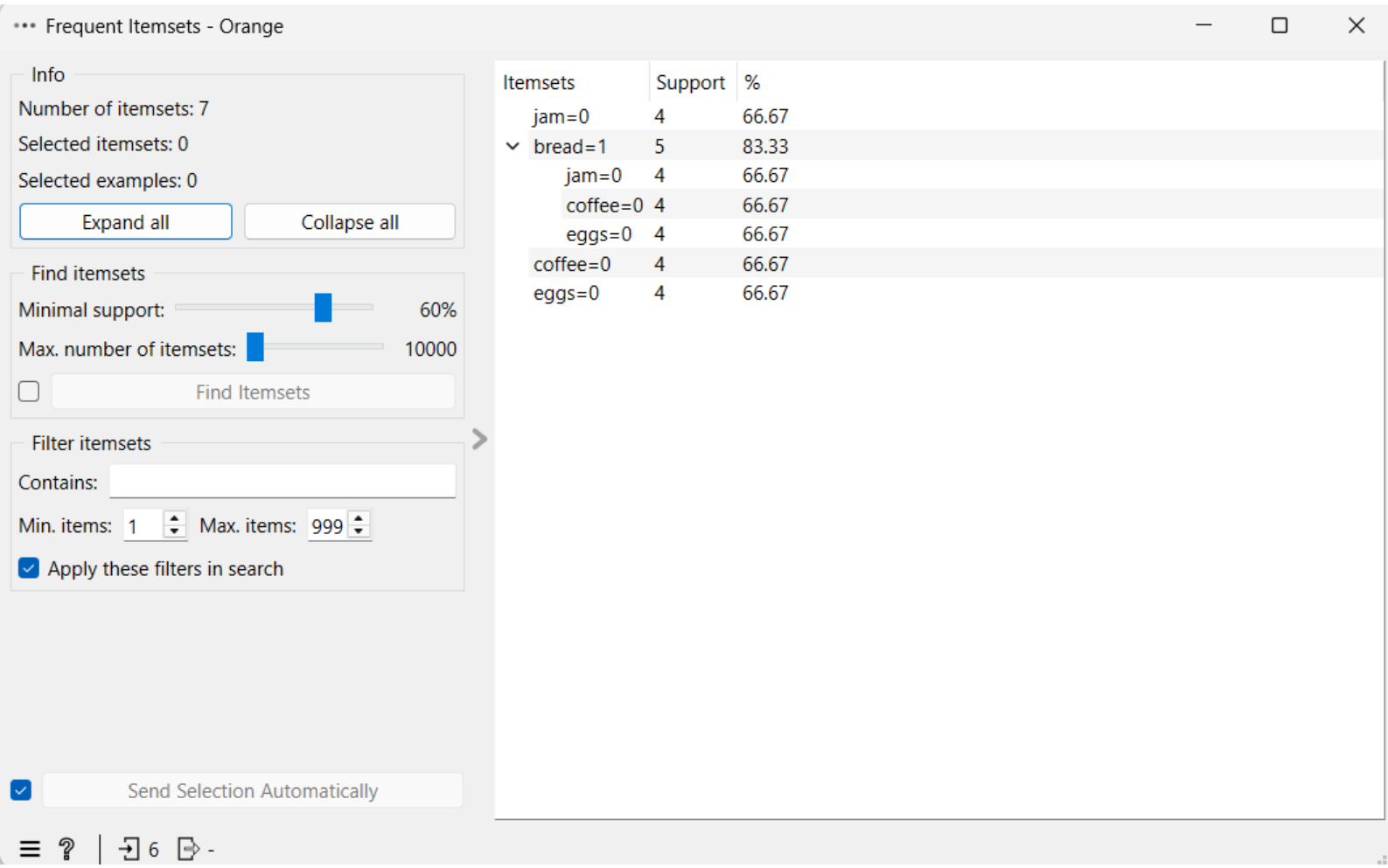
In [ ]:


CSV File Import
Data
Data Table
Selected Data → Data
Association Rules
Data
Frequent Itemsets




| | Transaction | bread | butter | coffee | eggs | jam | milk |
|---|---|---|---|---|---|---|---|
| 1 | T1 | 1 | 1 | 0 | 0 | 0 | 1 |
| 2 | T2 | 1 | 1 | 0 | 0 | 1 | 0 |
| 3 | T3 | 1 | 0 | 0 | 1 | 0 | 1 |
| 4 | T4 | 1 | 1 | 0 | 0 | 0 | 1 |
| 5 | T5 | 1 | 0 | 1 | 0 | 0 | 0 |
| 6 | T6 | 0 | 0 | 1 | 1 | 1 | 0 |


Association Rules - Orange
Info
Rules: 8 (shown 8)
Find association rules
Min. supp.: 50 %
Min. conf.: 50 %
Max. rules: 10k
Induce only classification rules
Restrict search by below filters
Find Rules
If checked, the rules are filtered according to these filter conditions already in the search phase. If unchecked, the only filters applied during search are the ones above, and the generated rules are filtered afterwards only for display, i.e. only the matching association rules are shown.
Filter by Antecedent
Contains:
Items, min: 3 max: 999
Filter by Consequent
Contains:
Items, min: 1 max: 999
Send selection
6
8
Supp Conf Covr Strg Lift Levr Antecedent Consequent
0.500 1.000 0.500 1.667 1.200 0.083 butter=1, coffee=0, eggs=0 → bread=1
0.500 1.000 0.500 1.000 2.000 0.250 bread=1, coffee=0, eggs=0 → butter=1
0.500 1.000 0.500 1.333 1.500 0.167 bread=1, butter=1, eggs=0 → coffee=0
0.500 1.000 0.500 1.333 1.500 0.167 bread=1, butter=1, coffee=0 → eggs=0
0.500 1.000 0.500 1.333 1.500 0.167 bread=1, coffee=0, milk=1 → jam=0
0.500 1.000 0.500 1.667 1.200 0.083 coffee=0, jam=0, milk=1 → bread=1
0.500 1.000 0.500 1.000 2.000 0.250 bread=1, coffee=0, jam=0 → milk=1


Frequent Itemsets - Orange
Info
Number of itemsets: 7
Selected itemsets: 0
Selected examples: 0
Expand all
Collapse all
Find itemsets
Minimal support:
60%
Max. number of itemsets:
10000
Find Itemsets
Filter itemsets
Contains:
Min. items:
1
Max. items:
999
Apply these filters in search
Send Selection Automatically
6
-
Itemsets
Support
%
jam=0 4 66.67
bread=1 5 83.33
jam=0 4 66.67
coffee=0 4 66.67
eggs=0 4 66.67
coffee=0 4 66.67
eggs=0 4 66.67