A4

August 13, 2023

0.1 Report on Mining Frequent, Maximal, and Closed Itemsets with Emojis

---

1. Calculation of Frequent Itemsets:

Dataset Transactions:

```
1:
2:
3:
4:
5:
6:
7:
8:
```

Minimum Support Count: 3

Frequent Itemsets: 1. : 8 2. : 4 3. : 5 4. : 5 5. : 3 6. : 5 7. : 5 8. : 2 (This is not frequent as support is less than 3) 9. : 2 (This is not frequent as support is less than 3) 10. : 3 11. : 2 (This is not frequent as support is less than 3) 12. : 2 (This is not frequent as support is less than 3) 13. : 3

---

2. Identification of Closed and Maximal Frequent Itemsets:

Closed Frequent Itemsets: - Itemsets that have no supersets with the same support count. 1. : 8 (Supersets such as or don't have support count 8) 2. : 5 (No immediate supersets like have support count 5) 3. : 3 (No immediate supersets like have support count 3)

Maximal Frequent Itemsets: - Itemsets that are frequent but none of their immediate supersets are frequent. 1. (It's frequent with support 3, but no supersets like are frequent)

---

3. Differences Between Maximal and Closed Frequent Itemsets:

- **Maximal Frequent Itemset**: Is about the itemset's relationship to its supersets. It is frequent, but none of its supersets are frequent. *Example*: is a maximal frequent itemset because even though it appears 3 times, none of its supersets like are frequent.

- **Closed Frequent Itemset**: Concerns the itemset's support count. It is frequent, and none of its immediate supersets have the same frequency. *Example*:   has a support count of 5. None of its supersets like   have the same support count.

---

4. Advantages and Disadvantages in the Context of Data Mining:

Advantages:

- **Maximal Frequent Itemsets**:
  - Reduces the number of itemsets: Since it focuses only on the largest itemsets that are frequent, it can decrease the number of itemsets to be considered.
- **Closed Frequent Itemsets**:
  - Provides concise representation: It offers a more concise representation of the frequent patterns without loss of information.

Disadvantages:

- **Maximal Frequent Itemsets**:
  - Might miss out on valuable insights: By focusing on the maximal sets, one could miss out on insights provided by the smaller sets.
- **Closed Frequent Itemsets**:
  - May not always reduce data size: In cases where many itemsets have the same support count, identifying closed frequent itemsets might not significantly reduce the number of patterns.

Conclusion:

Mining for frequent, maximal, and closed itemsets in a dataset, especially one as interesting as sequences of Emojis, can provide valuable insights. The choice between maximal and closed itemsets depends on the specific data mining goals and the characteristics of the dataset.

0.2 Task 2: Python Code

[1]:

```
# Required libraries
import pandas as pd
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori

# Sample dataset
dataset = [
    [' ', ' ', ' ', ' '],
    [' ', ' ', ' '],
    [' ', ' ', ' '],
    [' ', ' ', ' '],
    [' ', ' ', ' '],
    [' ', ' '],
    [' ', ' '],
    [' ', ' ']
```

```python
]

# Encoding data
te = TransactionEncoder()
te_ary = te.fit(dataset).transform(dataset)
df = pd.DataFrame(te_ary, columns=te.columns_)

# Applying the Apriori algorithm
# minimum support is 3/8 since there are 8 transactions and we need at least 3 transactions
# for an itemset to be considered frequent

frequent_itemsets = apriori(df, min_support=3/8, use_colnames=True)

# Function to find maximal and closed itemsets
def maximal_closed_itemsets(frequent_itemsets):
    maximal = []
    closed = []

    for i in range(len(frequent_itemsets)):
        is_maximal = True
        is_closed = True
        for j in range(len(frequent_itemsets)):
            # Checking if the set is a subset and is not itself
            if i != j:
                if frequent_itemsets['itemsets'].iloc[i].issubset(frequent_itemsets['itemsets'].iloc[j]):
                    is_maximal = False
                    if frequent_itemsets['support'].iloc[i] == frequent_itemsets['support'].iloc[j]:
                        is_closed = False
                        break

        if is_maximal:
            maximal.append(frequent_itemsets['itemsets'].iloc[i])
        if is_closed:
            closed.append(frequent_itemsets['itemsets'].iloc[i])

    return maximal, closed

maximal_itemsets, closed_itemsets = maximal_closed_itemsets(frequent_itemsets)

# Printing the results
print("Frequent Itemsets:\n", frequent_itemsets)
print("\nMaximal Itemsets:\n", maximal_itemsets)
print("\nClosed Itemsets:\n", closed_itemsets)
```

```
Frequent Itemsets:
    support   itemsets
0    0.500        ( )
1    0.625        ( )
2    0.625        ( )
3    1.000        ( )
4    0.500     ( , )
5    0.375     ( , )
6    0.625     ( , )
7    0.625     ( , )
8    0.375  ( , , )

Maximal Itemsets:
 [frozenset({' ', ' '}), frozenset({' ', ' ', ' '})]

Closed Itemsets:
 [frozenset({' '}), frozenset({' ', ' '}), frozenset({' ', ' '}),
frozenset({' ', ' '}), frozenset({' ', ' ', ' '})]
```