Exercise A2

Statistical Analysis of Big Data

Amanda Magzal 207608647

Consider the following transaction data:

Table 1: Transaction Data

TID	Items
1	{Bread, Milk}
2	{Bread, Diapers, Beer, Eggs}
3	{Milk, Diapers, Beer, Cola}
4	{Bread, Milk, Diapers, Beer}
5	{Bread, Milk, Diapers, Cola}

(a) Association Rules

Let $X = \{\text{Milk, Diapers}\}$ and $Y = \{\text{Beer}\}$, with association rule $X \to Y$.

The support count of itemset I is denoted by $\sigma(I)$.

Support:

$$s(X \to Y) = \frac{\sigma(X \cup Y)}{N}$$

Where N refers to the total number of transactions.

In the given transaction data, 2 out of the 5 transactions contain $X \cup Y = \{\text{Milk, Diapers, Beer}\}$ (ID: 3, 4).

$$s(X \to Y) = \frac{2}{5} = 0.4$$

Confidence:

$$c(X \to Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}$$

In the given transaction data, the support count for X is 3 (ID: 3, 4, and 5), and he support count for $X \cup Y$ is 2 (ID: 3 and 4). Hence, 2 out of the 3 transactions that contain X also contain Y.

$$c(X \to Y) = \frac{2}{3} = 0.67$$

(b) Apriori Algorithm Pseudo-Code

Let C_k denote the k-itemsets candidates, and F_k denote the frequent k-itemsets.

Algorithm 1: Apriori Algorithm

```
k = 1
F_1 = find all frequent 1-itemsets
while F_k ≠ ∅ do
    k = k + 1
    C_k = generate k-itemsets candidates using F_{k-1}
    C_k = prune candidates using C_k and F_{k-1}
    for each candidate in C_k do
        calculate the support count
    end
    F_k = candidates in C_k with support count > minsup
end
return ∪_k F_k
```

(c) Applying the Apriori Algorithm

The required *minsup* is 60%, hence the min support count is 3.

1. For each item in the transaction data, calculate the support count and generate the 1-itemsets candidates.

Table 2: 1-itemsets Candidates

Items	Support Count
{Bread}	4
{Milk}	4
{Diapers}	4
{Beer}	3
{Eggs}	1
{Cola}	2

2. Compare each candidate's support count with the minimum support count. The items {Eggs} and {Cola} do not satisfy the minimum support and therefore are not frequent.

Table 3: Frequent 1-itemsets

Items	Support Count
{Bread}	4
{Milk}	4
{Diapers}	4
{Beer}	3

3. Generate 2-itemsets candidates from the frequent 1-itemsets, and calculate the support count of each item.

Table 4: 2-itemsets Candidates

Items	Support Count
{Bread, Milk}	3
{Bread, Diapers}	3
{Bread, Beer}	2
{Milk, Diapers}	3
{Milk, Beer}	2
{Diapers, Beer}	3

4. Compare each candidate's support count with the minimum support count. The items {Bread, Beer} and {Milk, Beer} do not satisfy the minimum support and therefore are not frequent.

Table 5: Frequent 2-itemsets

Items	Support Count
{Bread, Milk}	3
{Bread, Diapers}	3
{Milk, Diapers}	3
{Diapers, Beer}	3

5. Generate 3-itemsets candidates from the frequent 2-itemsets.

Table 6: 3-itemsets

Items
{Bread, Milk, Diapers}
{Bread, Milk, Beer}
{Bread, Diapers, Beer}
{Milk, Diapers, Beer}

Prune the 3-itemsets candidates using the Apriory property - all subsets of frequent items must also be frequent.

- The itemset {Bread, Milk, Beer} includes the subset {Milk, Beer} which is not frequent. Therefore, it cannot be a frequent itemset.
- The itemset {Bread, Diapers, Beer} includes the subset {Bread, Beer} which is not frequent. Therefore, it cannot be a frequent itemset.
- The itemset {Milk, Diapers, Beer} includes the subset {Milk, Beer} which is not frequent. Therefore, it cannot be a frequent itemset.

Table 7: 3-itemsets Candidates

Items	Support Count
{Bread, Milk, Diapers}	2

6. Compare the candidate's support count with the minimum support count. The item {Bread, Milk, Diapers} does not satisfy the minimum support and therefore is not frequent. Hence, there are no frequent 3-itemsets.