Name: Om Makwana
Class-Roll No.: TY9-30
Batch: B
PRN: 22UF17360CM093

Experiment No. 5

Aim : Implementation of Association Rule Mining algorithm (Apriori).

Introduction :

- Association Rule Mining: Discovers relationships between items in a dataset.
- Apriori Algorithm: A popular algorithm for association rule mining based on frequent itemsets.
- Frequent Itemsets: Sets of items frequently occurring together in a dataset.
- Two-Step Process:
    - Frequent Itemset Generation: Finds itemsets meeting a minimum support threshold.
    - Rule Generation: Creates association rules from frequent itemsets (e.g., "if milk, then bread").

Procedure :

1. Import the necessary libraries:
2. Define a function to get frequent itemsets
3. Define a function to generate candidate itemsets
4. Define the Apriori algorithm
5. Use the Apriori algorithm to find frequent itemsets

Program Code and Output:

```
# Dwm exp5 ty9-30
from itertools import combinations

# Function to get frequent itemsets based on minimum support
def get_frequent_itemsets(transactions, min_support):
    itemsets = {}
    for transaction in transactions:
        for item in transaction:
            if item in itemsets:
                itemsets[item] += 1
            else:
                itemsets[item] = 1

    # Filter itemsets to only include those that meet or exceed the minimum support
    frequent_itemsets = {item: support for item, support in itemsets.items() if support >= min_support}
    return frequent_itemsets

# Function to generate candidate itemsets of size k
def get_candidate_itemsets(frequent_itemsets, k):
    candidates = []
    frequent_items = list(frequent_itemsets.keys())
    for combination in combinations(frequent_items, k):
        candidates.append(combination)
    return candidates

# Apriori algorithm to find all frequent itemsets
def apriori(transactions, min_support):
    k = 1
    # Initial set of frequent itemsets
    frequent_itemsets = get_frequent_itemsets(transactions, min_support)
    all_frequent_itemsets = [frequent_itemsets]

    # Iterate to find larger itemsets
    while frequent_itemsets:
        k += 1
        # Generate candidate itemsets of size k
        candidates = get_candidate_itemsets(frequent_itemsets, k)
        candidate_supports = {candidate: 0 for candidate in candidates}

        # Calculate support for each candidate itemset
        for transaction in transactions:
            for candidate in candidates:
                if set(candidate).issubset(set(transaction)):
                    candidate_supports[candidate] += 1

        # Filter candidate itemsets to only include those that meet or exceed the minimum support
        frequent_itemsets = {itemset: support for itemset, support in candidate_supports.items() if support >= min_support}
        if frequent_itemsets:
            all_frequent_itemsets.append(frequent_itemsets)

    return all_frequent_itemsets

# Example usage
transactions = [
    ['milk', 'bread', 'butter'],
    ['bread', 'butter'],
    ['milk', 'bread'],
    ['milk', 'butter'],
    ['bread', 'butter'],
    ['milk', 'bread', 'butter']
]

min_support = 2
frequent_itemsets = apriori(transactions, min_support)
print(frequent_itemsets)
```

```
[{'milk': 4, 'bread': 5, 'butter': 5}, {('milk', 'bread'): 3, ('milk', 'butter'): 3, ('bread', 'butter'): 4}]
```

Start coding or generate with AI.

Conclusion :

The Apriori algorithm is a fundamental tool in association rule mining, capable of uncovering meaningful relationships within large transactional datasets. By applying a minimum support threshold, it efficiently identifies frequent itemsets and generates strong association rules. Implementing Apriori in Python provides a systematic way to analyze data, revealing hidden patterns that support informed decision-making across various fields. Its simplicity, effectiveness, and practical applications make it a powerful technique for discovering valuable insights from complex data.

Review Questions :

1. **What is the Apriori algorithm in Association Mining Rule?**

Ans:

The **Apriori algorithm** is a popular method in **association rule mining** used to find frequent itemsets in a dataset and derive association rules. It works by **iteratively identifying itemsets** that occur frequently together and expanding them step-by-step, using the principle that **all subsets of a frequent itemset must also be frequent**.

2. **What is the significance of support, confidence, and lift in Apriori?**

Ans:

In the Apriori algorithm, **support, confidence, and lift** are key metrics to evaluate association rules:

- **Support**: Indicates how frequently an itemset appears in the dataset.
  Formula: Support(A) = (Transactions containing A) / (Total transactions)

- **Confidence**: Measures how often items in **Y** appear in transactions that contain **X** (for rule X → Y).
  Formula: Confidence(X → Y) = Support(X ∪ Y) / Support(X)

- **Lift**: Shows the strength of a rule over the random co-occurrence of X and Y.
  Formula: Lift(X → Y) = Confidence(X → Y) / Support(Y)

Significance:

- **Support** filters out rare combinations.

- **Confidence** shows the reliability of the rule.

- **Lift** indicates if X and Y are positively or negatively correlated (Lift > 1 means a strong association).

Github Link: https://github.com/OmMakwana249/DWM-Experiments