**Name:** Nikhil Sanjay Shinde

**PRN NO: 122B1F120  
Roll No:** TYITB120 **Assignment No. 5**

**AIM:** Assignment on Apriori Algorithm.

**PREREQUISITE:** Python programming

**THEORY:**

The Apriori algorithm is one of the traditional algorithms of association rule mining. It finds its application chiefly in market basket analysis, helping to determine the frequent itemsets in large datasets and infer strong association rules.

The Apriori algorithm follows a bottom-up philosophy, where it begins with a single item and gradually extends it step by step to larger-sized itemsets, provided that these itemsets frequently occur in the database. It is based on the fact that all subsets of a frequent itemset are themselves frequent.

**Apriori Algorithm Concept:**

Let's think of a supermarket example: if people purchasing bread also purchase butter, then that is something valuable to know. The Apriori algorithm reveals such correlations.

It works with two fundamental metrics:

**Support:** How often an itemset occurs in the database.

**Confidence:** How often items in Y occur in transactions that have X.

For instance, out of 100 transactions, if 40 include bread and 30 include both bread and butter, the support of {bread, butter} is 30%, and the confidence of the rule bread → butter is 75%.

Apriori assists in the elimination of rules that have minimum levels of support and confidence and hence is useful in recommendation systems and market research.

**DATASET INFORMATION**

The dataset used is based on the **Iris dataset**, which contains measurements of iris flowers from three species: *Iris-setosa*, *Iris-versicolor*, and *Iris-virginica*.

**Attributes:**

* **x0**: Constant column (could be dropped during preprocessing)
* **x1**: Sepal length in cm
* **x2**: Sepal width in cm
* **x3**: Petal length in cm
* **x4**: Petal width in cm
* **type**: Species label (categorical: Iris-setosa, Iris-versicolor, Iris-virginica)

Although Apriori is traditionally used for **transactional data**, this dataset was adapted by **binarizing continuous values into item-like categories**. For example, feature ranges were transformed into symbolic transactions like:

"SepalLength\_High", "PetalWidth\_Small", "Species\_Setosa"

**DATA PREPROCESSING**

To apply Apriori, we transformed numerical data into categorical items by **discretizing values into bins**, like:

* SepalLength: low, medium, high
* PetalLength: small, large
* Species: Setosa, Versicolor, Virginica

Then, each row (flower sample) was treated as a **transaction** of symbolic items.

This binarization enables us to convert the dataset to a list of transactions suitable for the Apriori algorithm.

**IMPLEMENTATION DETAILS**

The Apriori algorithm was implemented using Python’s mlxtend library as follows:

from mlxtend.preprocessing import TransactionEncoder

from mlxtend.frequent\_patterns import apriori, association\_rules

# Example preprocessing

transactions = [

['SepalLength\_High', 'PetalLength\_Low', 'Species\_Setosa'],

...

]

# Encode transactions

te = TransactionEncoder()

te\_ary = te.fit\_transform(transactions)

df = pd.DataFrame(te\_ary, columns=te.columns\_)

# Apply Apriori

frequent\_itemsets = apriori(df, min\_support=0.2, use\_colnames=True)

# Generate rules

rules = association\_rules(frequent\_itemsets, metric="confidence", min\_threshold=0.6)

**RESULTS AND OBSERVATIONS**

* **Support threshold**: 0.2
* **Confidence threshold**: 0.6

Rules like the following were generated:

If PetalLength\_Small → Species\_Setosa (confidence = 1.0, lift = 2.9)

If SepalLength\_High and PetalWidth\_Wide → Species\_Virginica (confidence = 0.8)

These rules show strong associations between morphological features and species classification.

**VISUALIZATION**

Rules can be visualized using scatter plots:

import matplotlib.pyplot as plt

plt.scatter(rules['support'], rules['confidence'], alpha=0.7)

plt.xlabel('Support')

plt.ylabel('Confidence')

plt.title('Association Rules')

plt.grid(True)

plt.show()

**Working Mechanism of Apriori Algorithm:**

* Set a Minimum Support and Confidence
* These minimum support thresholds are utilized to decide the association rule strength.
* Generate Candidate Item sets
* Identify all 1-itemsets that meet the minimum support.
* Generate Frequent Item sets
* Apply the Apriori principle to prune itemsets: if an itemset is not frequent, then all its supersets are not frequent.
* Generate Association Rules
* From frequent itemsets, derive rules that meet the minimum confidence.
* Evaluate Rules
* Measures such as Lift, Confidence, and Support are employed to judge the utility of the rule.

**Applications of Apriori Algorithm:**

* **Market Basket Analysis:** To identify sets of items often purchased together.
* **Recommendation Systems**: Suggest products based on purchase patterns.
* **Web Usage Mining:** Identify common navigation paths on a website.
* **Medical Diagnosis:** Identify symptom patterns that tend to occur together.
* **Inventory Management:** Cluster products that are sold together to maximize shelf space.

**Advantages of Apriori Algorithm:**

* **Easy to Understand:** Easy to implement and easy to understand.
* **Efficient Pruning:** Decreases the number of itemsets to process.
* **Strong Theoretical Basis:** Based on Apriori property for optimal performance.

**Limitations of Apriori Algorithm:**

* **Computationally Costly:** Slow for large databases due to the formation of numerous candidate sets.
* **Requires Multiple Passes:** Needs multiple passes over the entire database.
* **Inefficient for Low Support Thresholds:** Forms a huge number of candidate itemsets.

**Conclusion:**

 Apriori successfully identified feature patterns that align with specific species.

 Converting numerical measurements into symbolic bins made this dataset compatible with transactional algorithms.

 Although not a classic use case, this adaptation shows Apriori's flexibility.

 The most confident rules had **biological interpretability**, reinforcing the model’s credibility.

The Apriori algorithm is a basic algorithm in data mining for discovering interesting patterns and relationships among items in transactional databases. Though it has performance issues on big data, its conceptual simplicity and robust analytical foundation make it a popular method in association rule mining and retail analysis.