Department of Computer Science and Engineering (Data Science)

Machine Learning – IV

Experiment 6

Name: Umang Kirit Lodaya **SAP ID: 60009200032**

Batch: D11

Aim:

To implement and comprehend the Apriori algorithm, a fundamental technique for mining frequent itemsets in large datasets.

Theory:

Introduction to Apriori Algorithm:

- Developed by Agrawal and Srikant, Apriori is a classical algorithm for association rule mining.
- It identifies frequent itemsets by iteratively generating candidates and pruning infrequent ones.

Algorithm Overview:

- **Step 1: Candidate Generation**
  - Generate candidate itemsets of length k from frequent itemsets of length k-1.
  - The algorithm starts with frequent individual items and incrementally builds longer itemsets.
- **Step 2: Pruning**
  - Eliminate candidates that contain infrequent subsets, as any superset containing an infrequent subset would also be infrequent.
- **Step 3: Counting Support**
  - Scan the dataset to count the support of each candidate itemset.
  - Discard itemsets below a specified minimum support threshold.

Step-by-Step Implementation:

- **Step 1: Frequent Itemset Generation**
  - Implement a function to find frequent itemsets of length 1.
  - Iteratively generate longer itemsets until no more can be formed.
- **Step 2: Candidate Generation**
  - Write a function to generate candidate itemsets of length k from frequent itemsets of length k-1.
- **Step 3: Pruning**
  - Implement a pruning mechanism to eliminate candidates with infrequent subsets.
- **Step 4: Support Counting**
  - Scan the dataset and count the support for each candidate itemset.
  - Discard itemsets below the minimum support threshold.
- **Step 5: Association Rule Generation (Optional)**
  - If desired, implement a step to generate association rules from the frequent itemsets.

Implementation Tips:

- Efficiently store and index datasets for faster itemset counting.
- Experiment with different minimum support thresholds to observe their impact.

Lab Experiments to be Performed in This Session:

Execute the Apriori algorithm on a dataset to gain insights into its functionality and operation.

NAME: UMANG KIRIT LODAYA

SAP ID: 60009200032

BATCH: D11

APRIORI ALGORITHM

In [1]:
```
import pandas as pd
from itertools import combinations
```

In [2]:
```
data = {
    "T1": "COKE,FRIES,NUGGETS",
    "T2": "BURGER,COKE,FRIES",
    "T3": "COKE,FRIES,NUGGETS",
    "T4": "BURGER,FRIES,NUGGETS",
    "T5": "BURGER,COKE,FRIES,NUGGETS"
}

data = pd.DataFrame(data.items(), columns = ["TRANSACTIONS", "ITEMS BOUGHT"])
data["ITEMS BOUGHT"] = data["ITEMS BOUGHT"].apply(lambda x: x.upper().split(","))
data
```

Out[2]:

	TRANSACTIONS	ITEMS BOUGHT
0	T1	[COKE, FRIES, NUGGETS]
1	T2	[BURGER, COKE, FRIES]
2	T3	[COKE, FRIES, NUGGETS]
3	T4	[BURGER, FRIES, NUGGETS]
4	T5	[BURGER, COKE, FRIES, NUGGETS]

In [3]:
```
SUPPORT_THRESHOLD = 2
CONFIDENCE_THRESHOLD = 0.6
```

In [4]:
```python
COUNT = {}
for items in data["ITEMS BOUGHT"]:
    for item in items:
        COUNT[item] = COUNT.get(item, 0) + 1

pd.DataFrame(COUNT.items(), columns = ["ITEMS", "SUPPORT"])
```

Out[4]:

	ITEMS	SUPPORT
0	COKE	4
1	FRIES	5
2	NUGGETS	4
3	BURGER	3

In [5]:
```python
UNIQUE_ITEMS = list(COUNT.keys())
UNIQUE_ITEMS
```

Out[5]: ['COKE', 'FRIES', 'NUGGETS', 'BURGER']

In [6]:
```python
FREQUENT_ITEMS = []
for item in UNIQUE_ITEMS:
    if COUNT[item] ≥ SUPPORT_THRESHOLD:
        FREQUENT_ITEMS.append(item)

FREQUENT_ITEMS.sort()
FREQUENT_ITEMS
```

Out[6]: ['BURGER', 'COKE', 'FRIES', 'NUGGETS']

In [7]:
```python
MAX_PICK = len(max(data["ITEMS BOUGHT"], key = lambda x: len
(x)))
pick = 2

while pick ≤ MAX_PICK:
    temp = set()
    possibleItemSets = list(combinations(FREQUENT_ITEMS, pic
k))
    for itemset in possibleItemSets:
        COUNT[" ".join(itemset)] = 0
        A = set(itemset)
        for items in data["ITEMS BOUGHT"]:
            B = set(items)
            if A.intersection(B) == A:
                COUNT[" ".join(itemset)] += 1

        if COUNT[" ".join(itemset)] ≥ SUPPORT_THRESHOLD:
            for item in A:
                temp.add(item)

    FREQUENT_ITEMS = list(temp)
    pick += 1
```

In [8]:
```python
FINAL_ITEMSET = {}
for k, v in COUNT.items():
    if v ≥ SUPPORT_THRESHOLD:
        FINAL_ITEMSET[k] = v

pd.DataFrame(FINAL_ITEMSET.items(), columns = ["ITEMS", "SUPPO
RT"])
```

Out[8]:

	ITEMS	SUPPORT
0	COKE	4
1	FRIES	5
2	NUGGETS	4
3	BURGER	3
4	BURGER COKE	2
5	BURGER FRIES	3
6	BURGER NUGGETS	2
7	COKE FRIES	4
8	COKE NUGGETS	3
9	FRIES NUGGETS	4
10	COKE NUGGETS FRIES	3
11	COKE BURGER FRIES	2
12	NUGGETS BURGER FRIES	2

In [9]:
```python
last = -1
FINAL = []
for items in list(FINAL_ITEMSET.keys())[::-1]:
    if last == -1:
        last = len(items)

    if len(items) == last:
        FINAL.append(items.split())

FINAL
```

Out[9]: `[['NUGGETS', 'BURGER', 'FRIES']]`

In [10]:
```python
last
```

Out[10]: `20`

In [11]:
```python
RULES = {}
for items in FINAL:
    supportItems = COUNT[" ".join(items)]
    for i in range(len(items)):
        A = list(items[:i] + items[i + 1 :])
        B = items[i]

        try:
            supportA = COUNT[" ".join(A)]
        except:
            supportA = COUNT[" ".join(A[::-1])]

        A = ", ".join(A)
        RULES[A + " -> " + B] = round(supportItems / supportA, 2)

        supportA = COUNT[B]
        RULES[B + " -> " + A] = round(supportItems / supportA, 2)
```

In [12]:
```python
pd.DataFrame(RULES.items(), columns = ["RULES", "CONFIDENCE"
```

Out[12]:

	RULES	CONFIDENCE
0	BURGER, FRIES -> NUGGETS	0.67
1	NUGGETS -> BURGER, FRIES	0.50
2	NUGGETS, FRIES -> BURGER	0.50
3	BURGER -> NUGGETS, FRIES	0.67
4	NUGGETS, BURGER -> FRIES	1.00
5	FRIES -> NUGGETS, BURGER	0.40

In [13]:
```python
INTERESTING = []
for k, v in RULES.items():
    if v >= CONFIDENCE_THRESHOLD:
        INTERESTING.append(k)

for rule in INTERESTING:
    print(rule)
```

```
BURGER, FRIES -> NUGGETS
BURGER -> NUGGETS, FRIES
NUGGETS, BURGER -> FRIES
```