PRACTICAL 11

AIM: To study association rule learning and implement the Apriori algorithm using the Market_Basket_Optimization.csv dataset. Utilize the apriori function from the apyori.py file to discover frequent itemsets and generate association rules from the market basket data. Analyze the relationships between different products, identify common purchase patterns, and extract valuable insights about customer buying behavior. Evaluate the effectiveness of the Apriori algorithm in uncovering meaningful associations within the dataset, considering various support, confidence, and lift thresholds.

Code:

Import Libraries:

```
import pandas as pd
!pip install apyori
from apyori import apriori
```

```
Collecting apyori
  Downloading apyori-1.1.2.tar.gz (8.6 kB)
  Preparing metadata (setup.py) ... done
Building wheels for collected packages: apyori
  Building wheel for apyori (setup.py) ... done
  Created wheel for apyori: filename=apyori-1.1.2-py3-none-any.whl size=5954 sha256=a93cd91610c0dae80cfea861a2d73229da1ec79ae614eb19c67b58a47cf24427
  Stored in directory: /root/.cache/pip/wheels/c4/1a/79/20f55c470a50bb3702a8cb7c94d8ada15573538c7f4baebe2d
Successfully built apyori
Installing collected packages: apyori
Successfully installed apyori-1.1.2
```

Load Dataset:

```
dataset = pd.read_csv('/content/Market_Basket_Optimisation.csv', header=None)

# Preparing the transactions as a list of lists transactions = [] for i in range(0, len(dataset)):
transactions.append([str(dataset.values[i, j]) for j in range(0, len(dataset.columns)) if
str(dataset.values[i, j]) != 'nan'])

#Applying the Apriori Algorithm rules = apriori(transactions,
min_support = 0.003, min_confidence = 0.2,
min_lift = 3, min_length = 2)
results = list(rules)

#Extracting and Displaying
Results for rule in results: items = [x
for x in rule.items] print(f"Rule:
{items}") print(f"Support:
{rule.support}") for ordered_stat in
rule.ordered_statistics:
```

Name: DHRUV SHERE
Enrollment No: 23012022021
Batch: 4IT-B-2

```
    print(f"Confidence: {ordered_stat.confidence}")
    print(f"Lift: {ordered_stat.lift}")

        print("====================================")
```

```
Rule: ['escalope', 'mushroom cream sauce']
Support: 0.005732568990801226
Confidence: 0.3006993006993007
Lift: 3.790832696715049
====================================
Rule: ['escalope', 'pasta']
Support: 0.005865884548726837
Confidence: 0.3728813559322034
Lift: 4.700811850163794
====================================
Rule: ['fromage blanc', 'honey']
Support: 0.003332888948140248
Confidence: 0.2450980392156863
Lift: 5.164270764485569
====================================
Rule: ['ground beef', 'herb & pepper']
Support: 0.015997866951073192
Confidence: 0.3234501347708895
Lift: 3.2919938411349285
====================================
Rule: ['ground beef', 'tomato sauce']
Support: 0.005332622317024397
Confidence: 0.3773584905660377
Lift: 3.840659481324083
====================================
Rule: ['light cream', 'olive oil']
Support: 0.003199573390214638
Confidence: 0.20512820512820515
Lift: 3.1147098515519573
====================================
Rule: ['olive oil', 'whole wheat pasta']
Support: 0.007998933475536596
Confidence: 0.2714932126696833
Lift: 4.122410097642296
====================================
Rule: ['pasta', 'shrimp']
Support: 0.005065991201173177
Confidence: 0.3220338983050847
Lift: 4.506672147735896
====================================
Rule: ['spaghetti', 'milk', 'avocado']
Support: 0.003332888948140248
Confidence: 0.41666666666666663
Lift: 3.215449245541838
```