EX NO: **SCALABILITY ALGORITHMS: DEVELOP SCALABLE CLUSTERING ALGORITHMS, DEVELOP SCALABLE**

DATE: **APRIORI ALGORITHM**

AIM:

BACKGROUND THEORY:

APRIORI ALGORITHM:

To improve the efficiency of level-wise generation of frequent itemsets, an important property is used called Apriori *property* which helps by reducing the search space. All non-empty subset of frequent itemset must be frequent. The key concept of Apriori algorithm is its anti-monotonicity of support measure. Apriori assumes that All subsets of a frequent itemset must be frequent (Apriori property). If an itemset is infrequent, all its supersets will be infrequent.

DBSCAN:

Density-based spatial clustering of applications with noise (DBSCAN) is a clustering algorithm used in machine learning to partition data into clusters based on their distance to other points. Its effective at identifying and removing noise in a data set, making it useful for data cleaning and outlier detection.

PROCEDURE:

1) APRIORI ALGORITHM:

1. Load Data:

o Use the “File” widget to load your large dataset.

2. Python Script:

o Drag the “Python Script” widget to the canvas.
o Connect the “File” widget to the “Python Script” widget.
o Copy and paste the provided Python scripts into the “Python Script” widget.
o Ensure that the dataset file name in the script matches the file you loaded.

2) DBCSAN ALGORITHM:

1. Load Data:

o Drag the “File” widget to the canvas.
o Load your dataset file (e.g., dataset.csv).

2. Agglomerative Clustering:

o Drag the “DBSCAN” widget to the canvas.
o Connect the “File” widget to the “DBSCAN” widget.
o Configure the widget to use DBSCAN (default behavior).

3. Visualize Clustering:

o Drag the “box-plot” widget to the canvas.
o Connect the “DBSCAN” widget to the “Box-Plot” widget.

OUTPUT:

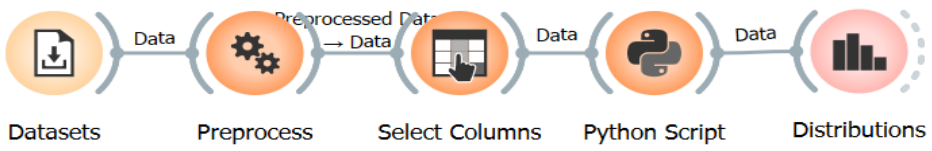
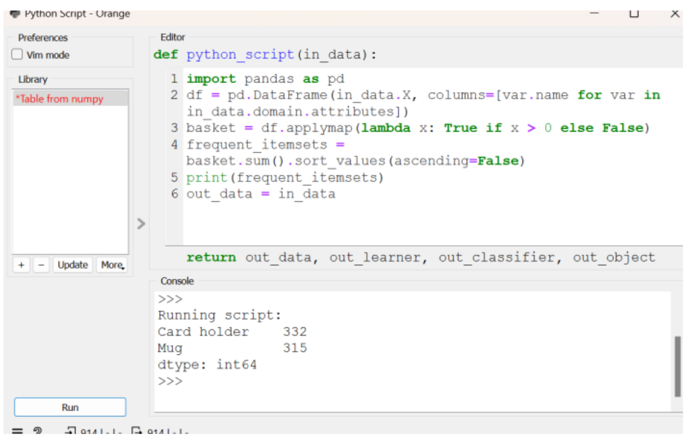


FIG 10.1: IMPLEMENTATION OF APRIORI ALGORITHM



```
def python_script(in_data):
import pandas as pd
df = pd.DataFrame(in_data.X, columns=[var.name for var in
in_data.domain.attributes])
basket = df.applymap(lambda x: True if x > 0 else False)
frequent_itemsets =
basket.sum().sort_values(ascending=False)
print(frequent_itemsets)
out_data = in_data

return out_data, out_learner, out_classifier, out_object
```

```
>>>
Running script:
Card holder    332
Mug            315
dtype: int64
>>>
```

FIG 10.1.1: APRIORI ALGORITHM USING PYTHON SCRIPT

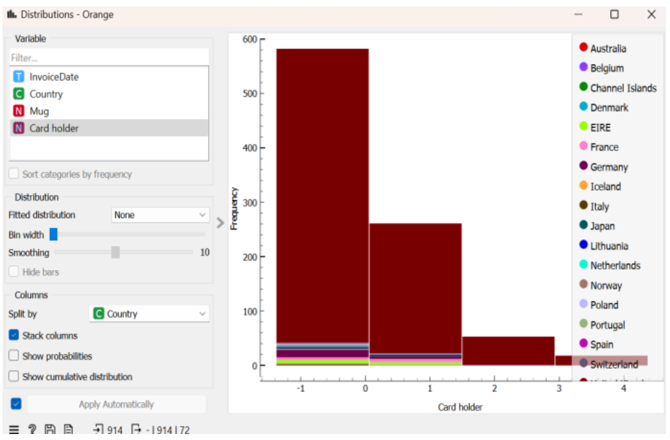


FIG 10.1.2: DISTRIBUTION OF APRIORI ALGORITHM

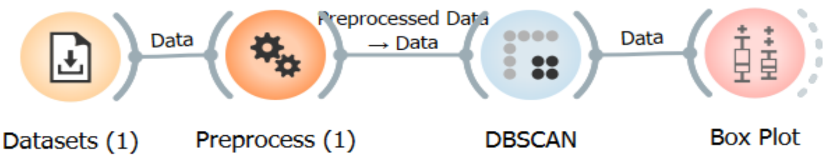


FIG 10.2: IMPLEMENTATION OF DBSCAN ALGORITHM

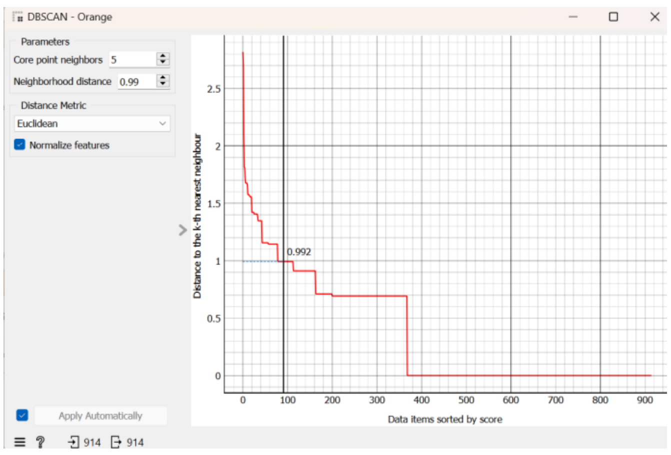


FIG 10.2.1: VISUALIZATING DBSCAN USING EUCLEDIAN

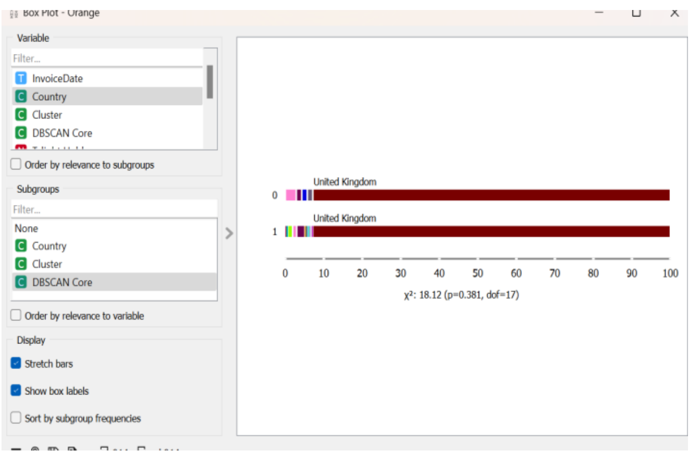


FIG 10.2.2: VISUALIZING DBSCAN USINH BOX-PLOT

RESULT: