Introduction

The first objective was to implement the A-Priori algorithm to find frequent itemsets with a support of at least **s** in a dataset of sales transactions. The second was to develop and implement an algorithm for generating association rules with the support of at least **s** and confidence of at least **c** between frequent itemsets discovered using the A-Priori algorithm in a dataset of sales transactions.

Implementation

1. **read_dataset function:**

   Reads a dataset file containing transactions and converts each line into a list of integers (transactions).

2. **get_frequent_itemsets function:**

   Identifies frequent itemsets of size 1 (single items) that meet the minimum support **s**.

3. **generate_candidates function:**

   Generates candidate itemsets of size **k** by combining frequent itemsets of size k-1. The support of the subset is always greater or equal to the support of the superset so if the subset is not frequent then the superset will not be frequent.

4. **filter_candidates function:**

   Filters candidate itemsets by calculating their support and retaining only those meeting the minimum support **s**.

5. **apriori function:**

   Implements the A-Priori algorithm to find all frequent itemsets of sizes up to **max_k**.

6. **generate_association_rules function**

   Generates association rules from the frequent itemsets based on minimum support **s** and confidence threshold **c**

- The data consisted of 100 000 baskets, each consisting on different number of items
- A support threshold of 1% (proposed in the lectures) was used giving us a support of **s=1000**
- For the association rule generation, a confidence threshold of **c=0.6** was used
- A max tuple size of **max_k = k_tuples = 3** was used i.e max itemset of 3

Instructions for Building and Running

- Download the sales transaction dataset from this (link)
- Install the necessary Python package

```
pip3 install -r requirements.txt
```

- Run instruction

```
python3 apriori.py
```

Results and Performance Analysis

- As for finding itemsets with a support of at least **s=1000** we found a lot of itemsets with **k=1** that satisfies this condition, hence we refer you to reading the complete results in the **result.txt** file
- As for frequent itemsets with **k=2** and **k=3** we got {368, 682}: 1193, {368, 829}: 1194, {825, 39}: 1187, {704, 825}: 1102, {704, 39}: 1107, {227, 390}: 1049, {722, 390}: 1042, {217, 346}: 1336, {829, 789}: 1194, {704, 825, 39}: 1035
- As for generating association rules with a minimum support **s=1000** and confidence **c=0.6** we got
    - {704} -> {825} with confidence: 0.61
    - {704} -> {39} with confidence: 0.62
    - {704, 825} -> {39} with confidence: 0.94
    - {704, 39} -> {825} with confidence: 0.93
    - {825, 39} -> {704} with confidence: 0.87
- The runtime for the algorithm (finding all frequent itemsets and generating association rules) was around 183 seconds