**NAME:** Kushal Srinivas  
**NJIT UCID:** ks2429  
**Email Address:** ks2429@njit.edu  
**Date:** 08/10/2024  
**Professor:** Yasser Abduallah  
**CS 634 Data Mining**  
 **Midterm Project Report**

**Implementation and Code Usage: Apriori Algorithm in Retail Data Mining**

**Abstract**

In this project, I explore the Apriori Algorithm, a key data mining technique used to identify associations in retail transactions. By implementing the algorithm and leveraging core data mining principles, I assess its ability to uncover frequent itemsets and generate actionable association rules. This report demonstrates the design and development of a custom tool to mine valuable insights from retail transaction data.

**Introduction**

Data mining involves uncovering hidden patterns and relationships within large datasets. In retail, understanding these associations can guide business strategies. The Apriori Algorithm is a well-established method for mining frequent itemsets and generating association rules. This project explores its application in retail, focusing on the Nike dataset. We apply key data mining principles such as support, confidence, and frequent itemset generation.

**Core Concepts and Principles**

* **Frequent Itemset Discovery**: The Apriori Algorithm identifies sets of items that frequently occur together in transactions.
* **Support and Confidence**:
  + *Support* measures how often an item or itemset appears in the dataset.
  + *Confidence* indicates how likely items are purchased together.
* **Association Rules**: Strong rules are derived from frequent itemsets, showing the likelihood of co-purchases.

**Project Workflow**

1. **Data Loading and Preprocessing**:  
   Transaction data is loaded from a CSV file, cleaned, and preprocessed to ensure item uniqueness and proper sorting.
2. **Determining Minimum Support and Confidence**:  
   User-defined thresholds for minimum support and confidence filter out less relevant patterns.
3. **Generating Frequent Itemsets**:  
   The Apriori Algorithm iterates through item combinations, incrementing the itemset size (K = 1, 2, 3, etc.), and filters frequent itemsets based on the support threshold.
4. **Association Rule Generation**:  
   Association rules are derived from the frequent itemsets and evaluated using confidence.

**Implementation**

Using Python's mlxtend library for Apriori and FP-Growth, the project also implements a brute-force algorithm for comparison. The dataset used contains Nike transactions involving items like "Running Shoes" and "Sweatshirts."

* **Brute Force Implementation**: Calculates all possible combinations of items and retains those meeting the minimum support.
* **Apriori Algorithm**: Efficiently generates frequent itemsets by reducing the number of combinations.
* **FP-Growth Algorithm**: Another efficient method for frequent itemset mining, used for comparison.

**Results and Evaluation**

The project evaluates each algorithm's performance, comparing execution times and results:

* **Brute Force**: Accurate but slow for large datasets.
* **Apriori**: Efficient, balancing speed and accuracy.
* **FP-Growth**: Similar to Apriori but faster for large datasets.

| **Algorithm** | **Time (seconds)** | **Frequent Itemsets** | **Association Rules** |
| --- | --- | --- | --- |
| Brute Force | 0.75 | 12 | 6 |
| Apriori | 0.30 | 12 | 6 |
| FP-Growth | 0.25 | 12 | 6 |

**Conclusion**

This project demonstrates the practical application of the Apriori Algorithm in retail data mining. By identifying strong associations in Nike transactions, the project reveals the value of frequent itemset mining for retail decision-making. The results show that Apriori and FP-Growth are more efficient than brute force while producing accurate results.

**Screenshots**

Screenshots of the code and outputs have been included to demonstrate functionality. The source code, datasets, and visualizations are attached to this report for reference.

Figure 1 : walmart Item Names CSV file  
Figure 2 : Nike Transactions CSV file

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Below are the screenshots of the code from the phyton file:

A screenshot of a computer

Description automatically generated

The list of transactions for three stores—Nike, Amazon, and Walmart—is generated by the script, with randomly selected items for each store. 20 transactions are created per store, and each transaction contains 2 to 6 items. The transactions are then saved into a CSV file named store\_transactions.csv. The code effectively produces a DataFrame with transactions and saves the outcome in the CSV.

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

This code generates a series of predetermined transactions for three different stores: Nike, Amazon, and Walmart. Each store has its own specific inventory of items, and the transactions are stored in separate CSV files based on the store and database number. For example, the transactions for Nike are stored as nike\_transactions\_db1.csv, and so forth.

The main steps involved in this process are:

- Defining the list of items for each store.

- Predefining sets of transactions for each store.

- Creating a function to store each store's transactions in a CSV file.

- Storing the transactions for Nike, Amazon, and Walmart.

Creating the dataitems and creating 20 transactions.

A screenshot of a computer program

Description automatically generated

The performance of three algorithms—Brute Force, Apriori, and FP-Growth—is compared using transaction datasets from Nike, Amazon, and Walmart. Each algorithm is assessed for frequent itemset mining and association rule generation based on user-defined minimum support and confidence thresholds.

The key steps are as follows:

- Using predefined transaction data from Nike, Amazon, and Walmart.

- Taking minimum support and confidence levels from the user to determine frequent itemsets and association rules.

- Implementing the Brute Force Algorithm, which checks all combinations of items to find frequent itemsets.

- Applying the Apriori & FP-Growth algorithms on the transaction data for standard frequent itemset mining.

- Printing the results for each algorithm, including frequent itemsets, association rules, and execution times, for comparison.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

The output displays frequent itemsets and association rules for transactions at Amazon and Walmart. Frequent itemsets represent the combinations of items purchased together, while association rules forecast how buying one item influences the likelihood of buying another.

At Amazon:

Illustration: {Charger, Laptop} is present in 2 transactions.

Rule: Purchasing a Charger often results in the purchase of a Laptop (confidence = 600%).

At Walmart: Illustration: {Milk, Bread} is present in 3 transactions.

Rule: Purchasing Milk frequently results in the purchase of Bread (confidence = 1200%).

**The Final output**

A screenshot of a computer program

Description automatically generated

Association rules:A computer screen shot of a program

Description automatically generated

Other

**GitHub Repository:**

The complete code and dataset are available on my GitHub repository:

Colab link: <https://github.com/kushal-srinivas/nike-walmart>