**Association Rule Mining Project Report**

**Name:** Sushil Saindane  
**NJIT UCID:** sbs8  
**Email Address:** sbs8@njit.edu  
**Date:** 10/10/2024  
**Professor:** Yasser Abduallah  
**Course:** CS 634101 Data Mining

**Abstract**

This project explores the implementation and application of Association Rule Mining techniques, specifically focusing on the Apriori Algorithm, FP-Growth, and a custom brute force method. By analyzing transaction data from five different retail stores (Amazon, Best Buy, Nike, Walmart, and Target), we aim to uncover meaningful associations between products and compare the performance of different algorithms.

**Introduction**

Association Rule Mining is a crucial technique in data mining that helps discover interesting relationships between items in large datasets. This project implements and compares three different approaches to association rule mining:

1. A custom brute force method
2. The Apriori algorithm (using the mlxtend library)
3. The FP-Growth algorithm (using the mlxtend library)

We apply these techniques to retail transaction data to identify frequent itemsets and generate association rules, providing valuable insights for business decision-making.

**Core Concepts and Principles**

**Frequent Itemset Discovery**

The project revolves around identifying sets of items that frequently occur together in transactions. This forms the basis for generating association rules.

**Support and Confidence**

Two key metrics used in our analysis:

* Support: Measures how frequently an itemset appears in the dataset.
* Confidence: Indicates the likelihood that an item Y is purchased when item X is purchased.

**Association Rules**

Rules in the form of "If X, then Y" are generated, showing which items are likely to be purchased together.

**Project Workflow**

1. **Data Generation and Preprocessing**
   * Created synthetic transaction data for five stores
   * Implemented functions to save and load transactions from CSV files
2. **Algorithm Implementation**
   * Developed a custom brute force method for finding frequent itemsets and generating rules
   * Utilized mlxtend library implementations of Apriori and FP-Growth algorithms
3. **User Interface**
   * Created an interactive command-line interface for users to select stores and set parameters
4. **Analysis and Comparison**
   * Implemented functions to compare results from different algorithms
   * Measured and compared execution times of each method

**Implementation Details**

**Data Handling**

* Utilized pandas for data manipulation
* Implemented custom functions for CSV file operations

**Brute Force Method**

* Iteratively generated candidate itemsets
* Calculated support and confidence for each potential rule

**Library Methods**

* Used mlxtend's implementations of Apriori and FP-Growth
* Leveraged pandas DataFrames for efficient data processing

**Performance Measurement**

* Used Python's time module to measure execution time for each algorithm

**Results and Analysis**

[Include specific results from your runs here. For example:] When analyzing the Nike store transactions with a minimum support of 20% and minimum confidence of 20%, we obtained the following results:

1. Brute Force Method:
   * Generated 6 association rules
   * Execution time: X.XXX seconds
2. Apriori Algorithm:
   * Generated 6 association rules
   * Execution time: X.XXX seconds
3. FP-Growth Algorithm:
   * Generated 6 association rules
   * Execution time: X.XXX seconds

Sample rules generated:

* Rule 1: {'Crew Socks (6-Pack)'} → {'Graphic Gym Bag'} (Confidence: 57.14%, Support: 20.00%)
* Rule 2: {'Dri-FIT Headband'} → {'Dri-FIT T-Shirt'} (Confidence: 57.14%, Support: 20.00%)

[Include more detailed results and analysis here]

**Comparison of Algorithms**

Our analysis showed that while all three methods produced the same number of rules for the given parameters, their execution times varied:

1. Brute Force Method: Slowest but provides a baseline for comparison
2. Apriori Algorithm: Faster than brute force, especially for larger datasets
3. FP-Growth: Generally the fastest, particularly effective for sparse datasets

[Include more detailed comparisons based on your specific results]

**Conclusion**

This project successfully implemented and compared three different approaches to association rule mining. We found that while the custom brute force method provides a good understanding of the underlying process, the library implementations of Apriori and FP-Growth offer significant performance advantages, especially for larger datasets. The association rules generated from our retail transaction data provide valuable insights into customer purchasing behavior, which can be used for product placement, marketing strategies, and inventory management.

**Future Work**

* Implement additional algorithms for comparison (e.g., Eclat)
* Analyze real-world transaction data from actual retail stores
* Develop a graphical user interface for easier interaction with the program
* Explore ways to visualize the generated association rules

**GitHub Repository**

[Include your GitHub repository link here]

**References**

1. Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. Proc. 20th int. conf. very large data bases, VLDB, 1215, 487-499.
2. Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. ACM sigmod record, 29(2), 1-12.
3. Mlxtend library documentation: <http://rasbt.github.io/mlxtend/>

This report provides a comprehensive overview of your Association Rule Mining project, including the implementation details, analysis of results, and comparisons between different algorithms. You may need to adjust some sections based on your specific implementation details and results. Don't forget to include any screenshots or specific output from your program to illustrate the results and user interface.