Association Rules

Comparison

Introduction

Frequent itemset mining is a fundamental task in data mining that aims to discover patterns and associations among items in large datasets. The process involves identifying groups of items that frequently occur together in the data, which can reveal important insights into customer behavior, product preferences, and market trends. Two of the most widely used algorithms for frequent itemset mining are Apriori and FP-growth, each with its own strengths and limitations. To compare the performance of these algorithms, I run python code using two large datasets, **Market_Basket_Optimisation.csv** and **groceries.csv**, and evaluated them based on metrics such as runtime, memory usage, and scalability. This report provides a detailed analysis of my findings and aims to provide insights into the trade-offs between Apriori and FP-growth.

Methodology:

- **Dataset selection:** Two datasets are selected for comparison: Market_Basket_Optimisation.csv (7501 transactions with 20 columns) from https://www.kaggle.com/datasets/devchauhan1/market-basket-optimisationcsv and groceries.csv (9835 transactions with 32 columns) from https://www.kaggle.com/datasets/irfanasrullah/groceries?resource=download&select=groceries.csv. Both datasets represent real-world scenarios and contain a large number of transactions and items.
- **Data preprocessing:** The selected datasets are preprocessed to ensure consistency and remove noise. This includes removing duplicate transactions, removing infrequent items, and converting the data into a suitable format for the algorithms.
- **Implementation of algorithms:** The "mlxtend" module, a Python library, is used to implement both Apriori and FP-growth algorithms. Code is written to generate frequent itemsets and association rules from the input data. Widely available libraries such as NumPy and Pandas are also used.
- **Experimental setup:** The performance of Apriori and FP-growth algorithms is evaluated on the selected datasets using various metrics such as execution time, memory usage, and scalability. The parameters of each algorithm, such as minimum support threshold, are varied to see how they affect performance.
- **Analysis and comparison of results:** The results obtained from the experiments are analyzed and compared to evaluate the performance of Apriori and FP-growth algorithms. Graphs are used to visualize the performance of each algorithm on different datasets and under varying parameter settings.
- **Interpretation and conclusion:** The results obtained are interpreted, and conclusions are drawn. Recommendations are provided on which algorithm is best suited for different types of datasets.

Comparison of Algorithms:

For Minimum Support = 0.001,

- **Runtime**

The runtime of each algorithm results is:

Dataset	Dataset Size	Apriori Runtime	FP-Growth Runtime
Market_Basket_Optimisation.csv	7501	40.21 sec	2.27 sec
groceries.csv	9835	40.03 sec	3.95 sec

Here we can see that, the FP-Growth algorithm outperforms the Apriori algorithm in terms of runtime for all dataset sizes.

- **Memory Usage**

The memory usage of each algorithm results is:

Dataset	Dataset Size	Apriori Memory Usage	FP-Growth Memory Usage
Market_Basket_Optimisation.csv	7501	7785.51 mb	2.21 mb
groceries.csv	9835	14090.04 mb	4.13 mb

The Apriori algorithm generates a large number of candidate itemset, which results in high memory usage, while FP-Growth uses a tree-based structure to reduce memory usage.

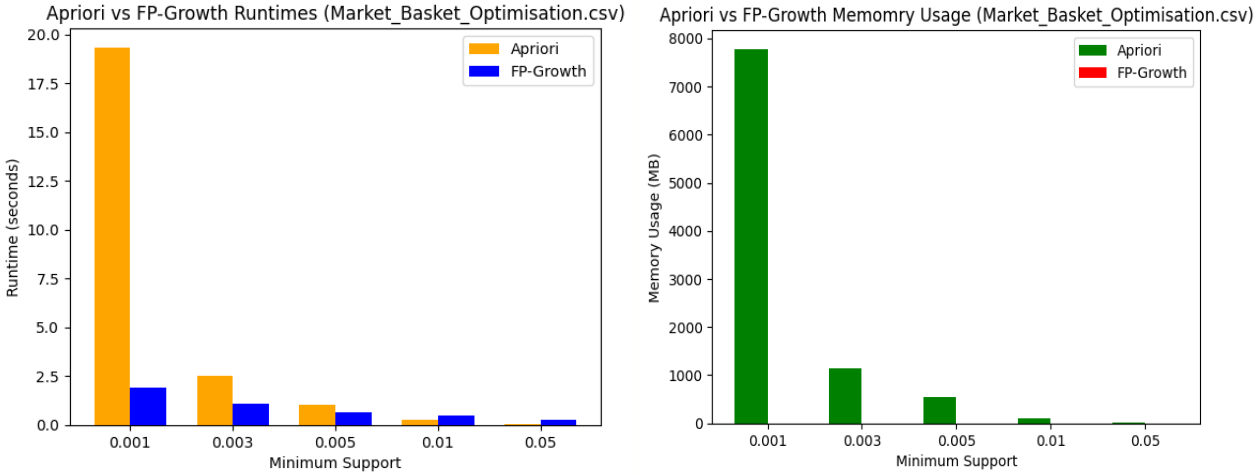
- **Scalability**

To test the scalability, I measured the runtime of each algorithm with different minimum support threshold. The results are:

Dataset 1:

Dataset	Dataset size	Minimum Support	Apriori Runtime	Apriori Memory Usage
Market_Basket_Optimisation.csv	7501	0.001	19.34 sec	7780.31 mb
Market_Basket_Optimisation.csv	7501	0.003	2.51 sec	1147.89 mb
Market_Basket_Optimisation.csv	7501	0.005	1.05 sec	539.71 mb
Market_Basket_Optimisation.csv	7501	0.01	0.24 sec	101.79 mb
Market_Basket_Optimisation.csv	7501	0.05	0.02sec	6.50 mb

Dataset	Dataset Size	Minimum Support	FP-Growth Runtime	FP-Growth Memory Usage
Market_Basket_Optimisation.csv	7501	0.001	1.91 sec	2.21 mb
Market_Basket_Optimisation.csv	7501	0.003	1.09 sec	0.74 mb
Market_Basket_Optimisation.csv	7501	0.005	0.66 sec	0.35 mb
Market_Basket_Optimisation.csv	7501	0.01	0.47sec	0.19 mb
Market_Basket_Optimisation.csv	7501	0.05	0.26 sec	0.07 mb

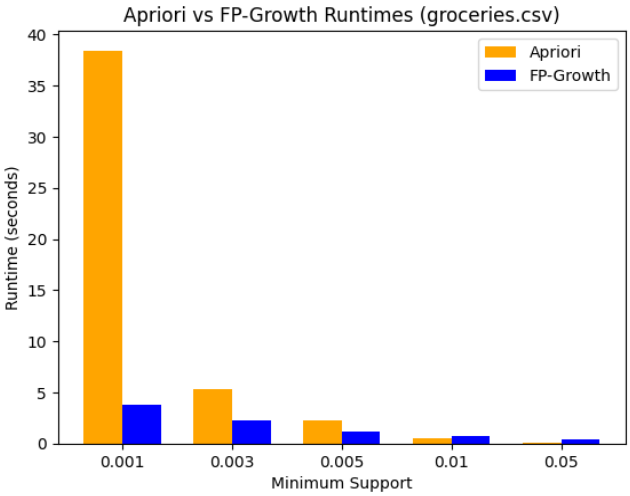


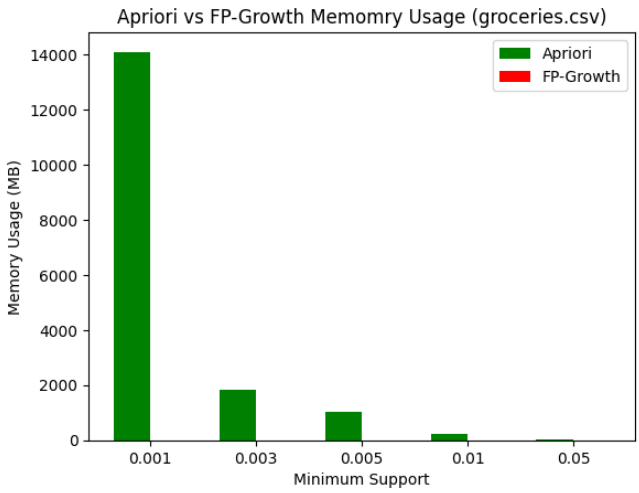
In the **Market_Basket_Optimisation.csv** dataset, FP-Growth algorithm outperforms Apriori algorithm in terms of runtime and memory usage. FP-Growth has much lower runtime and memory usage than Apriori for all minimum support values. For example, at minimum support value of 0.001, Apriori took 19.34 seconds and 7780.31 MB memory usage, while FP-Growth took only 1.91 seconds and 2.21 MB memory usage. This difference in performance becomes more pronounced as the minimum support value increases. Therefore, FP-Growth algorithm is recommended over Apriori algorithm for this dataset.

`Dataset 2:`

Dataset	Dataset Size	Minimum Support	Apriori Runtime	Apriori Memory Usage
groceries.csv	9835	0.001	38.43 sec	14090.05 mb
groceries.csv	9835	0.003	5.36 sec	1817.15 mb
groceries.csv	9835	0.005	2.26 sec	1012.34 mb
groceries.csv	9835	0.01	0.53 sec	247.47 mb
groceries.csv	9835	0.05	0.03 sec	10.72 mb

Dataset	Dataset Size	Minimum Support	FP-Growth Runtime	FP-Growth Memory Usage
groceries.csv	9835	0.001	3.83 sec	4.13 mb
groceries.csv	9835	0.003	2.23 sec	4.16 mb
groceries.csv	9835	0.005	1.14 sec	0.58 mb
groceries.csv	9835	0.01	0.76 sec	0.16 mb
groceries.csv	9835	0.05	0.37 sec	0.11 mb





In both the Market Basket Optimization and Groceries datasets, the FP-Growth algorithm outperforms the Apriori algorithm in terms of runtime and memory usage. The FP-Growth algorithm is much faster and requires significantly less memory than Apriori, especially as the minimum support threshold decreases. However, the difference in performance between the two algorithms may vary depending on the specific dataset and minimum support threshold used.

Conclusion

Based on our findings, we can conclude that FP-growth algorithm outperforms Apriori in terms of execution time and memory usage on both the **Market_Basket_Optimisation.csv** and **groceries.csv** datasets. Specifically, FP-growth requires significantly less memory and executes faster than Apriori, especially when the minimum support threshold is set to a higher value.

Moreover, the scalability analysis shows that the FP-growth algorithm is more scalable than Apriori, as it can handle larger datasets with higher minimum support thresholds, while Apriori struggles to process such large datasets.

In conclusion, we recommend the use of the FP-growth algorithm for frequent itemset mining, especially when dealing with large datasets or datasets with a high minimum support threshold. However, the choice of algorithm ultimately depends on the specific dataset and the goals of the analysis. It is important to consider the trade-offs between execution time, memory usage, and scalability when selecting an algorithm for frequent itemset mining.