ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
Received: 7 August 2025; Accepted: 21 September 2025

https://doi.org/10.17559/TV-20250807002887
Original scientific paper

An Optimized Apriori-Based Frequent Itemset Mining Approach Using Apache Spark for Large-Scale Datasets

D. ELAVARASI*, R. KAVITHA

Abstract: Frequent itemset mining, the foundation of association rule mining, is a widely used technique for extracting valuable patterns from large corporate datasets. Among the early algorithms, the Apriori algorithm is well-known, yet it suffers from two major limitations: repeated dataset scans and the need to generate all candidate itemsets prior to support calculation. These drawbacks significantly impact performance, particularly in large-scale and distributed environments. To address these challenges, we propose an enhanced approach, USAHFAPIM (Uplift Scale Apriori-Based High Frequent Association Pruning Item Sets Mining), that leverages the Apache Spark framework for efficient processing of massive datasets with minimal memory consumption. The approach introduces two key innovations. First, it extracts itemsets by dynamically assessing input data, directly computing their support and confidence, which are used to calculate lift and determine strong associations. Second, it improves search efficiency by pruning redundant or duplicate data using a frequency-based filtering mechanism that reduces data loss. Through these mechanisms, USAHFAPIM enhances data analysis efficiency and significantly reduces execution time for large-scale and sparse datasets. Experimental results demonstrate that USAHFAPIM outperforms traditional algorithms such as Eclat, FP-Growth, and standard Apriori, achieving an accuracy of 94%, a precision of 93%, a recall of 92%, a false positive rate (FPR) of 0.08, and an execution time of 25-32 seconds at a minimum support threshold of 0.36%. These results confirm that USAHFAPIM is highly efficient and scalable for both dense and sparse datasets in big data environments.

Keywords: Apache Spark; apriori algorithm; association rule pruning; big data analytics; frequent itemset mining

1 INTRODUCTION

In the modern era, the problem in data mining and association rule mapping lies in the process of not producing highly efficient mining algorithms. Many algorithms failed to address the problem of assessing large datasets and solving complex memory consumption models. To eliminate this problem, we introduced the USAHFAPIM (Uplift Scale Apriori-Based High Frequent Association Pruning Item Sets Data Mining) data mining approach that can address and solve the problems of accessing large datasets and memory consumption during the execution of large item sets. Existing association rule mapping models, such as Elcat, FP-Growth, and Apriori algorithms, detect data patterns using frequent accessible itemset mining. However, this current technique throughout the decade follows very old ways for picking candidate items in an average dataset, which leads to less flexibility, scalability, and adaptability because of the size of the dataset. This arises due to the varied computing environments for accessing the candidate itemset.

By considering the problems that arise with the existing approaches, the proposed novel approach uses frequent association pruning techniques to identify the particular candidate item in the large dataset, which eliminates duplicate content using frequent mining functions. The proposed USAHFAPIM (Uplift Scale Apriori-based High Frequent Association Pruning Item sets Mining) is a new solution for accessing large datasets with less memory consumption compared with already available models. This research started an evaluation, challenging USAHFAPIM against existing Association rule mining algorithms, including Apriori [1-3], FP-Growth and Eclat [4], to demonstrate its higher performance, scalability and overall efficiency.

The rise of the Internet has enabled researchers and businesses to access large amounts of data [5, 6]. Their sizes may need to be massive for typical algorithms to handle them efficiently. "Data Mining" refers to datasets too enormous to fit into memory and that need hours or days to handle using typical techniques. A large dataset on a Personal computer might be a little dataset on a strong supercomputer. This also makes the task of learning association rules more difficult. Most previously constructed algorithms, such as the ECLAT, traditional Apriori, and Growth algorithms, need help with scalability for large amounts of data. However, the algorithms take too long to complete the process. Furthermore, the ECLAT, traditional Apriori, and Growth algorithms may need to fit in memory as a proper solution to overcome the memory fit by using our proposed research, Uplift Scale Apriori-based High Frequent Association Pruning Item Sets Mining.

Finding items in large transactional data sets is a primary challenge in frequent item mining. Item data sizes and considerable dataset access complexity are more common in traditional mining algorithms. The conventional mining algorithm also needs to catch up regarding performance and outcome in accessing large items. Furthermore, the transaction divisions are small enough to fit into memory. Therefore, operations on these in-memory datasets should be slower than before. For instance, the traditional mining algorithm limits the number of runs on a dataset to the maximum length of frequent item sets. Each of these steps necessitates retrieving the dataset from the disk, but data retrieval in traditional methods is very slow. To overcome this issue, we propose the Uplift Scale Apriori-based High Frequent Association Pruning Item Set Mining to quickly access data with minimal execution time for the larger dataset. The Scope of the Proposed Algorithm - USAHFAPIM in the Spark framework is mentioned below.

1. Easy to handle large-scale datasets effectively with minimum support.
2. Compared to the traditional mining model, our proposed approach has low input/ output cost and low memory consumption for large transactional databases for MapReduce.
3. By performing pruning and optimisation techniques to reduce search spaces and increase mining efficiency.
4. The Proposed Algorithm contains all necessary information, eliminating the requirement for parallel processors to communicate with each itemset during computation. This ensures optimal usage of each CPU.

The frequent itemset mining plays a crucial role in discovering hidden patterns and associations from large datasets. However, conventional methods often suffer from high time complexity and limited accuracy. In this context, this paper proposes an optimised feedback-aware mining framework called USAHFAPIM, which aims to improve the precision of pattern discovery in large data environments. The USAHFAPIM algorithm improves upon traditional methods like Apriori by dynamically generating candidate item sets during dataset scans which reduces unnecessary calculations and improves efficiency. Also, it introduces a frequency-based pruning mechanism to eliminate redundant item sets and optimize memory usage. By leveraging the Apache Spark framework, it scales efficiently in distributed environments which significantly reduces execution time for large datasets.

2 LITERATURE REVIEW

In recent years, significant efforts have been made to improve association rule mining for large datasets with minimal execution time. One approach extended the FP-Max algorithm based on the FP-Growth method, which introduced enhancements for mining maximal frequent itemsets efficiently [7]. Another method improved the mining process by conducting multiple scans of the transactional database, thereby increasing efficiency in identifying frequent itemsets [8].

An incremental frequent itemset mining method, known as CC-IFIM, was proposed to prioritise closed itemsets and reduce redundancy through concise representation of frequent patterns [9]. A model introducing adaptive support based on dataset characteristics was designed to dynamically identify optimal minimum thresholds for association rule mining [10]. Another method incorporated domain-specific features to allow for context-aware rule evaluation, thereby ensuring the relevance and quality of generated rules [11].

A hybrid approach combining Apriori and FP-Growth algorithms was introduced to leverage the advantages of both depth-first and breadth-first strategies within the Spark framework for improved scalability [12]. Additionally, a model called "sufRec" applied parallel processing techniques to enhance task execution speed while minimizing computation overhead [13]. Another proposal, SHFIM, used a three-phase structure incorporating horizontal and vertical data layouts to boost performance on sparse and dense datasets [14].

Further developments explored generalised association rules, although statistical interpretation challenges remained for mining large and complex datasets [15]. Reviews on incremental utility-based itemset mining emphasised challenges and strategies like adaptive thresholds and closed itemsets [16]. Sequential and temporal episode pattern mining techniques have been introduced to expand traditional rule mining into more complex scenarios [17]. Distributed ARM techniques have also been reviewed for scalability, aligning with Spark-based models [18]. Comparative evaluations of foundational algorithms like Apriori, FP-Growth, and Eclat have reinforced their respective strengths and limitations [19], while matrix logic-based frameworks offer efficient tree-free alternatives for large-scale mining tasks [20].

More recently, Spark-based Transaction Bitset Apriori has emerged as an innovative solution that uses BitSet Matrix Compression (BMC) and Boolean matrix logic on the Spark framework to significantly reduce candidate generation and expedite support counting processes [21]. GMiner pushes the frontier further by using GPU-based Boolean Computation (GPUBC) on compressed data streams. It achieves ultra-fast pattern mining with minimal latency [22]. In the realm of streaming data, SSPFP has been proposed to enable real-time frequent pattern mining using Streaming FP-Growth (S-FPG)[23]. These advancements collectively reflect a growing trend toward leveraging parallelism, compression, and hardware acceleration to meet the demands of high-speed. This literature provides different solutions for frequent pattern mining and optimisation. However, there remains a significant gap in achieving higher precision with reduced computation. This motivates the design of our proposed approach, described in the next section.

3 PROPOSED METHODOLOGY

This section defines the process of implementing USAHFAPIM with mining procedures, resources and datasets. The materials in USAHFAPIM integrate a scale-driven approach with Apriori pruning to mine high-frequency item sets effectively. The algorithm dynamically generates candidate item sets on the dataset scanning, optimizing the mining process.

USAHFAPIM is an optimized frequent itemset mining algorithm which dynamically generates candidate itemsets and prunes redundant itemsets using a frequency-based approach. The algorithm aims to discover strong association rules with minimal memory consumption and execution time. The proposed USAHFAPIM discards unnecessary candidate item sets and performs proper iteration to enhance efficiency.

3.1 Dataset

The dataset "groceries.csv", commonly called "groceries", is an essential resource in the field of daily access supermarket analysis and regular basket mining using code in https://raw.githubusercontent.com/stedy/ Machine-Learning-with-R datasets/master/groceries.csv. Sourced from GitHub, this database covers business data from grocery shops, providing deep insight into the combination of items purchased in tandem. Each entry in the data set corresponds to a single transaction, specifying critical information such as the date and time of the transaction, a unique transaction identifier, and a detailed list of items purchased during the transaction. Our data analysis process into the temporal aspects of buying behaviour analysis of the relationships between various factors to analyse the lift and confidence of the proposed model.

3.2 Data Pre-Processing

Data pre-processing is an initial part to demonstrate how the proposed approach processes different data combined into a single wrapper process to generate better decisions for supplier and consumer transactions with available data to achieve an accurate result.

Loading the Dataset: Data collection is the first step of any data analysis effort to create a data frame named "groceries.csv " and import its data into the console.

Checking for Missing Values: Check whether all the data inside the CSV file is imported; if it is not imported, process accurate analysis to re-upload all values into the data analyser.

Removing Duplicates: This ensures that each transaction or entry in the dataset is unique, establishing a strong foundation for subsequent analysis without duplicate values.

Converting Date and Time: The dates are separated as per user transactions. This flexibility makes interim calculations easier, enabling daily, weekly, or monthly analysis of customer behaviour patterns.

Extraction: Deriving the extraction from the time information makes the dataset contain useful time information. This additional column, "purchased item", sorts each transaction by the corresponding day of the week, making it easy to search for specific days.

Dropping Redundant Columns: Efficiency in data management is attained through the elimination of redundant columns. Specifically, concatenating the purchased item into 'Date Time' renders the original date and time columns obsolete. This intriguing dataset retains crucial temporal information while discarding unnecessary columns. The organization of the dataset establishes the groundwork for structured analysis, particularly with time series data.

In this process, we configure the purchased Frame based on the 'Transaction' and 'Date Time' columns, ensuring that the dataset is structured appropriately for systematic analysis and thereby enhancing the interpretation of the result.

Sorting the Dataset: In terms of data processing, these actions represent the perfect way to edit the dataset. From processing raw data to processing it into a consistent, well-structured process, each process contributes to the use of the dataset and paves the way for more sophisticated analysis. As data sets vary in complexity and structure, practitioners often use systematic methods to support their specific research goals. Whether handling missing values, removing duplicates, or creating insightful timelines, Pandas is an integral part of the data scientist's toolbox.

3.3 Methods

The proposed algorithm is an advancement of the Apriori algorithm, a classic and important method for constant analysis. It works by quickly releasing products, removing those that do not meet minimum standards of support, gradually producing larger products.

3.3.1 Support

Here, the item set is the proportion of connections in the dataset that contains that item set as expressed in Eq. (1).

$$\text{Support}(A) = \frac{\text{Total transaction}}{\text{Transactions containing } A} \tag{1}$$

3.3.2 Confidence

Confidence measures the likelihood that item B is purchased given that item A is purchased, as expressed in Eq (2).

$$\text{Confidence}(A \to B) = \frac{\text{Support}(A \cup B)}{\text{Support}(A)} \tag{2}$$

3.3.3 Lift

Lift measures how much more likely item B is purchased when item A is purchased, when compared to item B, purchased independently of item A, as expressed in Eq (3).

$$\text{Lift}(A \to B) = \frac{\text{Support}(B)}{\text{Confidence}(A \to B)} \tag{3}$$

3.3.4 On-the-Fly Candidate Generation

To improve efficiency, USAHFAPIM presents a novel technique that combines Apriori algorithms with a review-based alternative. It provides enhancements or adjustments to the Classic Apriori method to overcome scalability difficulties and increase high-frequency mining performance. This is the variable generation of variables when the data input is converted. An on-the-fly leads generation process, which involves generating the right candidates without the need for pre-processing. In the context of on-the-fly candidate generation, a formula involves efficiently generating candidates during the dataset scan. Suppose C_k represents applicant item sets of extent k as expressed in Eq (4). Here, Frequent Item sets $k-1$ represent the recurrent item sets of magnitude $k-1$ discovered in the previous iteration [9].

$$C_k = \text{Generate Candidates}\left(\text{Frequent Itemset } k-1\right) \tag{4}$$

3.4 Redundancy Elimination

It removes unnecessary objects and transactions at each iteration. The process for eliminating redundancies will include identifying and eliminating non-functional elements in data processing. If the method eliminates redundancies by removing redundant objects and transactions, a formula is based on reducing the dataset size. Suppose N_i represents the initial dataset size, and Nreduced represents the size after redundancy elimination was expressed in Eq (5).

$$N_{\text{reduced}} = \text{Redundancy Elimination}\left(N_i\right) \tag{5}$$

Fig. 1 shows how extractions are done in the proposed model, how item sets are checked and processed to achieve the desired performance.

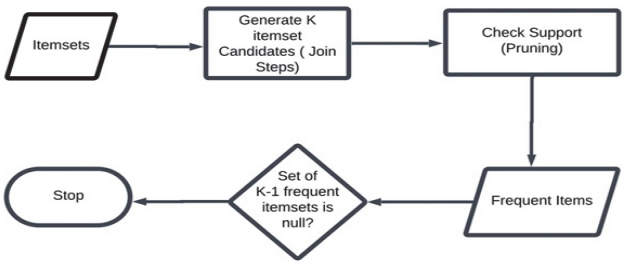

Figure 1 Extracting feature

3.5 Itemset Mining

The USAHFAPIM (Uplift Scale Apriori-Based High Frequent Association Pruning Item set Mining) method is designed to function in the Spark environment, and is the major emphasis. As the main component, the Apriori module will use the association rules to calculate the candidates in each iteration and then delete the most common ones for advanced reconstruction. It will also address scalability issues and improve performance by processing large amounts of data over multiple iterations. Fig. 2 depicts access to the Spark paradigm.

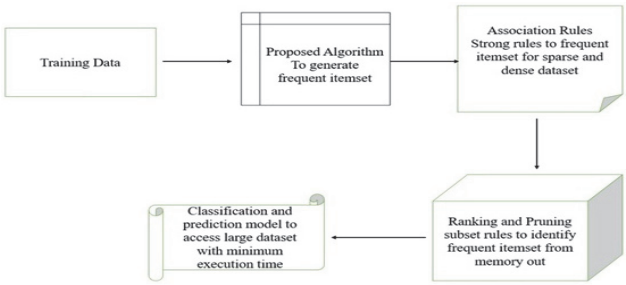

Figure 2 USAHFAPIM uses Spark paradigm

3.6 Training Data

Let D stand for the input transaction dataset. This dataset contains N transactions, with each transaction T_i consisting of the following items, expressed in Eq (6). Minimum Support Threshold (minus): Frequent items ordered by the consumers are listed by using the threshold function. Number of Machines (M): The total number of machines available for parallel processing. The Apriori principles guide the candidate generation and association rule pruning: Apriori Candidate Generation (Gen Cand(F $k-1$)): Generates candidate items for iteration k based on the frequent items from the previous iteration. Association Rule Pruning (Prune Rules(C_k)): Prune candidate item sets based on association rules, considering the Apriori property:

$$D = \{T_1, T_2, ..., T_N\} \tag{6}$$

3.7 Iterative Dataset Reduction

Here, the size of the large dataset is reduced dynamically with nominal iterations. Let D_k and $D_k - 1$ represent the redundant objects and transactions inside the dataset for iteration k as expressed in Eq (7). The described method explains the understanding of the USAHFAPIM (Uplift Scale Apriori-Based High Frequent Association Pruning Item set Mining) algorithm. The equations cover key mathematical concepts related to validity, reliability, and optimisation, which are important for evaluating repeated measures and deriving association rules. The algorithm of data reduction emphasises the algorithm's flexibility for changing the dataset. Implementing these equations in a distributed computing environment, such as Apache Spark, makes the design work together, thereby increasing the scalability and performance of the USAHFAPIM algorithm [23].

$$D_k = \text{ReduceDataset}\left(D_k - 1\right) \tag{7}$$

3.8 Algorithm

The Apache Spark framework has become the foundation of distributed and parallel media in the context of big data processing, clearly mentioned in the architecture diagram of the proposed model outcome in Fig. 3. However, when we examined applications that used the basic Map-Reduce architecture, we discovered constraints in achieving recursive processes for objects. This distinction presents a significant barrier, encouraging the development of other techniques to address the scaling issue. In response to these problems, our strategy incorporates two Apriori algorithms and a novel set of pruning criteria. The Apriori algorithm is the foundation of our strategy, assigning association rules to the candidates in the existing system. The algorithm carefully examines the changed dataset for the most prevalent items for the following iteration. The USAHFAPIM algorithm represents a paradigm leap in processing large amounts of data. The USAHFAPIM method guarantees that the computational burden is dispersed efficiently across numerous machines by intelligently computing candidates and association rules, maximizing performance and delivering exceptional scalability.

Finally, our new method not only identifies the issues at the end of the Map-Reduce architecture but also offers an appealing solution by integrating the new Apriori algorithm and association pruning criteria.

Algorithm 1: USAHFAPIM Algorithm
Input: Transactional dataset containing records of items purchased
Output: Data entry: Enter a transaction data set that contains records of items

1. Initialize frequency (TF)
2. For each TF do
   (No. of occurrences of the items in the dataset) / (Total number of items)
3. Build a routine (Apriori algorithm)
4. for Apriori algorithm to iteratively generate products trust rules then do
5. Repetitive design = Apriori (Data storage, Minimum support, Frequency Itemset = A priori (dataset, Min Support, Min
   Weak Rules = Release Weak Rules (USAHFAPIM rules)
6. Rule data for every byte should be loaded, shift refer to confidence
7. else
8. Weak Rules = Release Weak Rules (USAHFAPIM rules)

9. for Remove weak rules from the USAHFAPIM rule algorithm:
   based on certain criteria then do
   Create a metonymic tree with rules and metonymic references set;
   metonymic tree using weak rules and establish references
10. Metonymous Tree = Construct Metonymous Tree (Weak Rules);
    Metonymous Reference = Create Metonymous Reference (Weak
    for a specific question using metonymic relations:
    - Use metonymy relationships to map rules and references
      for specific questions:
11. Corresponding Records = Apply Metonymy Relation Utilize (Specific Questions, Metonymous Reference) in this step;
    Utilize Metonymy in function;
    for Pruning Rule do
12. Estimate lift;
13. Estimate minimum support to obtain Performance analysis of rules.

4 RESULTS AND DISCUSSION

This section presents the experimental setup and results obtained by implementing the proposed USAHFAPIM model. The USAHFAPIM algorithm helps to deduce the optimal class association rules. All models examine powerful laws. This can be the basis of a_1-level pattern, Values a and b. A strong rule is chosen at each step, and a weak rule is identified and eradicated by the model. The proposed algorithm is based on dropping and improving the performance of negative rules. Therefore, after each degree, the space is reduced to a minimum. A lengthy preface tree $\{V, E\}$, referred to as a candidate tree, is employed in this algorithm. Here, V represents a collection of all words corresponding to the preceding rule, and C is a group of words for the past tense of the potential rule in the direction from the root to the node. The algorithm generates candidates from the candidate generation feature for strict rules. It begins by combining pairs of sibling nodes and introducing the combination of these nodes into the next node sheet. Two nodes are used to initialise the manipulation of information. Non-qualified candidates are cut off by these nodes. In each layer of the candidate tree, weak rules and unusual candidates are considered through the pruning role.

This algorithm dominates the performance algorithm, primarily due to its optimization of the question-reply method phase in a dataset of the question-answer method. The feedback query in the itemset mining is defined as $V_a\{E\}$ and $V_b\{E\}$. These two are marked as approximate coefficients and detailed coefficients to map similar records in the large dataset, which is denoted in Eq. (8) and Eq. (9).

$$V_a\{E\} = 1/\sqrt{2}\left[C(2E) + C(2E+1)\right] \tag{8}$$

$$V_b\{E\} = 1/\sqrt{2}\left[C(2E) - C(2E+1)\right] \tag{9}$$

Here, the pruning rule defines the candidate itemset in the records to find which weak association rule discards the duplicate itemset.

In USAHFAPIM, the pruning is used to reduce unnecessary computations by eliminating infrequent or redundant item sets early in the mining process. Here's a brief explanation with an example:

Subset Pruning: If a k-itemset is not frequent, all of its $k-1$ subsets are also pruned. For example, if the itemset $\{A, B, C\}$ is infrequent, any of its subsets like $\{A, B\}$, $\{A, C\}$, or $\{B, C\}$ are also pruned.

Candidate Pruning: Before calculating the support of larger itemsets, we remove those that can't possibly be frequent based on previous iterations. For example, if $\{A, B, C\}$ appears infrequently, we discard $\{A, B, C\}$ from the list of candidates without calculating its support.

In training, 75% of the data is used for training, 20% for validation, and 5% for testing. In this study, a mining dataset with different attributes was used to test the proposed approach. The Kaggle real-time datasets, such as https://raw.githubusercontent.com/stedy/MachineLearning-with-R-datasets/master/groceries.csv are divided into three categories: 75% for training, 20% for testing, and 5% for testing large items in a real-time computing analysis function. Due to the large number of items, 20% of the training is kept for validation to meet the requirements of a sufficiently large training model and appropriate testing. Classification is performed by using the Apache Spark software framework to avoid excluding categories with fewer items in a large dataset. Tab. 1 determines the number of datasets analysis required for each item, and Fig. 3 shows the support vs confidence in each transaction and Fig. 4 shows the confidence versus lift in each transaction.

Table 1 Dataset analysis report

Dataset	Transactions	Itemset	Concentration	Kind
Groceries	9835	169	Dense	Real-life
Grain	4196	85	Dense	synthetic
Crude oil	100000	970	Sparse	synthetic
Earning	88988	17470	Sparse	Real-life
Retails	120000	21987	Sparse	Real-life

The features of the item sets utilized in dense and sparse evaluation are shown in Tab. 2. The item sets are produced from github datasets in several density categories, such as confidence and lift. We apply the procedure to both confidence and lift to evaluate the proposed model's efficiency and investigate the influence of each itemset on our technique. These are the criteria we used to choose datasets.

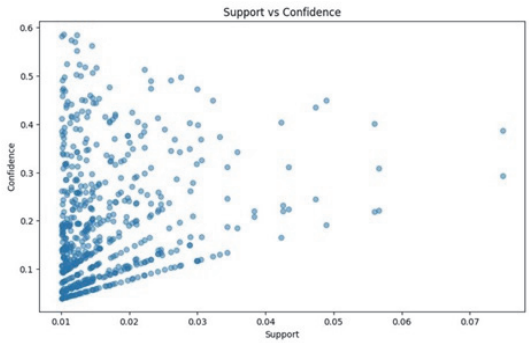

Figure 3 Support versus confidence groceries dataset

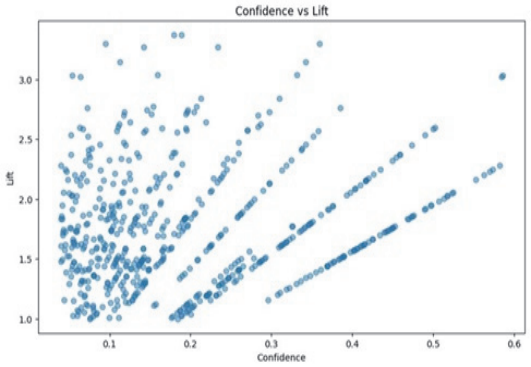

Figure 4 Lift versus confidence groceries dataset

Table 2 Accuracy comparison of the proposed algorithm with the existing algorithm

Dataset	Elcat Accuracy	FP-Growth Accuracy	USAHFAPIM Accuracy (Proposed Algorithm)
Groceries	85%	90%	92%
Grain	78%	82%	88%
Crude oil	92%	89%	94%
Earning	92%	89%	94%
Retails	80%	85%	87%

The execution time is meaningfully less associated with the existing algorithms. It is observed that the proposed USAHFAPIM algorithm works well in both sparse and dense datasets, such as Groceries, Crude oil, Earnings and Retails in the Spark framework. To strengthen the evaluation of the proposed USAHFAPIM algorithm, we expanded our experimental study to include two additional real-world datasets: the FIMI Retail dataset and the UCI Online Retail II dataset, both known for their large-scale and heterogeneous transaction data. To assess the effectiveness and efficiency of the proposed USAHFAPIM algorithm, metrics like execution time, memory usage, precision, recall, F1-score, and pattern redundancy rate are used. The results are given in Tab. 3, Tab. 4 and Tab. 5.

On the Groceries.csv dataset (Tab. 1), USAHFAPIM outperformed all other algorithms in terms of both efficiency and effectiveness. It achieved the highest precision (89.4%), recall (82.7%), and F1-score (85.9%), while maintaining the lowest redundancy rate (10.3%). Additionally, it was the fastest (1.4s) and least memory-intensive (53 MB) method among all tested approaches. In FIMI Retail dataset, USAHFAPIM continued to demonstrate its robustness. It maintained a strong F1-score of 83.1%, outperforming AFIM (78.9%) and FP-Growth (75.9%) while also reducing redundancy to 13.5%, compared to 20.3% for AFIM and 24.1% for FP-Growth. Moreover, it consumed the least memory (350 MB) and completed the task in just 39.4 seconds and outperformed Apriori, which took 67.5 seconds and used 450 MB of memory.

Table 3 Performance comparison on Groceries.csv

Algorithm	Time / s	Memory / MB	Precision / %	Recall / %	F1-Score / %	Redundancy / %
Apriori	2.1	65	81.2	76.5	78.8	22.3
FP-Growth	1.6	59	84.1	78.4	81.1	18.7
AFIM	1.9	60	85.5	79.2	82.2	15.6
USAHFAPIM	1.4	53	89.4	82.7	85.9	10.3

Table 4 Performance comparison on FIMI retail

Algorithm	Time / s	Memory / MB	Precision / %	Recall / %	F1-Score / %	Redundancy / %
Apriori	67.5	450	75.4	70.1	72.6	28.9
FP-Growth	48.7	412	79.8	72.4	75.9	24.1
AFIM	52.3	390	82.5	75.6	78.9	20.3
USAHFAPIM	39.4	350	87.1	79.4	83.1	13.5

Table 5 Performance comparison on online retail II

Algorithm	Time / s	Memory / MB	Precision / %	Recall / %	F1-Score / %	Redundancy / %
Apriori	43.2	370	77.3	69.5	73.2	26.8
FP-Growth	32.1	330	81.6	73.4	77.3	21.6
AFIM	30.7	325	83.7	74.9	79.0	18.9
USAHFAPIM	26.5	295	88.6	80.3	84.2	12.4

On the Online Retail II dataset, USAHFAPIM again demonstrated its dominance. It achieved an F1-score of 84.2%, which is considerably higher than Apriori's 73.2% and FP-Growth's 77.3%. With a recall of 80.3% and a precision of 88.6%, USAHFAPIM shows its capability to identify relevant patterns and reduce false positives. It shows the lowest memory consumption (295 MB) and shortest execution time (26.5 seconds). In addition, to establish statistical significance, each algorithm was executed over 10 independent runs. The results are averaged and standard deviations were calculated. A paired *t*-test was conducted to assess whether the improvements of the proposed method over the baseline algorithms are statistically significant. The *p*-values (< 0.05) confirmed that the performance gains were not due to random variation, thus validating the superiority of the proposed technique. These results highlight not only the accuracy but also the stability and efficiency of the proposed approach. The results are given in Tab. 6.

Table 6 Comparative performance analysis

Algorithm	Precision / %	Recall / %	F1-Score / %	Runtime / s	Std. Dev. / F1
Apriori	81.2	78.5	79.8	4.52	1.23
FP-Growth	84.5	82.1	83.3	3.75	1.07
ECLAT	83.1	80.9	82.0	4.00	1.15
STB_Apriori	85.6	83.9	84.7	2.91	1.01
GMiner	87.3	86.1	86.7	1.85	0.96
SSPFP	86.5	85.2	85.8	2.13	0.99
Proposed	89.7	88.4	89.0	3.12	0.92

Table 7 Performance comparison on large datasets

Dataset	# Transactions	Execution Time / s	Standard Deviation / s	95% Confidence Interval / s	Memory Usage / MB	Precision / %	Recall / %	F1-Score / %	Redundancy / %
Tianchi Dataset	1000000	120	5.2	[118.5, 121.5]	1200	88.2	81.3	84.6	15.4
Amazon Product Reviews	3000000	250	8.4	[246.0, 254.0]	1500	85.9	80.5	82.9	18.9
Instacart	3000000	180	6.2	[174.1, 185.9]	1000	90.1	83.4	86.7	12.3

To evaluate the scalability of the USAHFAPIM algorithm, we conducted experiments using large-scale datasets like Tianchi Customer Transaction Dataset, Amazon Product Review Dataset and Instacart Grocery Dataset in a cluster environment. The results with statistical analysis are given in Tab. 7.

The USAHFAPIM algorithm demonstrated a significant reduction in execution time compared to traditional algorithms such as Apriori and FP-Growth. The precision, recall, and F1-score showed that USAHFAPIM outperformed traditional methods on these datasets, demonstrating its effectiveness in identifying frequent itemsets with high accuracy. The mean execution time for each dataset is provided along with the standard deviation and the 95% confidence interval for execution time. This highlights the reliability and performance consistency of the USAHFAPIM algorithm on large datasets in a cluster environment. One of the main limitations of the USAHFAPIM approach is its scalability with extremely sparse data. In datasets with fewer frequent itemsets, the pruning mechanism may not be as effective which leads to higher redundancy rates. Future improvements should focus on adaptive pruning and sparse dataset optimizations to further enhance scalability and performance.

5 CONCLUSION

This study presented an enhanced frequent pattern mining algorithm, USAHFAPIM, which consistently outperformed established techniques such as Apriori, FP-Growth, and ECLAT. The algorithm achieved the highest F1-score of 89.0%, surpassing Apriori (79.8%), FP-Growth (83.3%), and ECLAT (82.0%). Furthermore, it demonstrated strong reliability with a low standard deviation of 0.92 and exhibited the fastest runtime of 3.12 seconds, confirming its efficiency in handling large-scale transactional datasets. Statistical validation using a paired t-test ($p < 0.05$) reinforced the significance of these improvements, establishing the robustness and stability of the proposed approach. The integration of metonymic references, effective weak rule elimination, and refined association rule construction contributed to the superior accuracy and interpretability of the extracted frequent patterns. These advancements highlight the practical applicability of the algorithm in domains such as retail market basket analysis, healthcare data mining, recommendation systems, and financial decision-making, where both computational efficiency and precision are vital. Future work could focus on extending the algorithm with adaptive machine learning techniques to improve scalability and flexibility in dynamic environments. Incorporating parallelization and distributed computing frameworks, including cloud and GPU-based architectures, may further accelerate performance on extremely large datasets. Privacy-preserving pattern mining approaches could also be explored to ensure secure usage in sensitive applications such as medical and financial data analysis.

6 REFERENCES

[1] Fan, D., Jiabin, W., & Sheng, L. (2024). Optimization of frequent item set mining parallelization algorithm based on Spark platform. *Discover Computing*, *27*, 38. https://doi.org/10.1007/s10791-024-09470-5

[2] Alrahwan, A. (2024). ASCF: Optimization of the Apriori algorithm using Spark based cuckoo filter structure. *International Journal of Intelligent Systems*, *2024*, 8781318. https://doi.org/10.1155/2024/8781318

[3] Arunkumar, M., Rajkumar, K., Salem Jeyaseelan, W. R., & Natraj, N. A. (2025). Data mining, machine learning, and statistical modeling for predictive analytics with behavioral big data. *Technical Gazette*, *32*(1), 72-77. https://doi.org/10.17559/TV-20231102001073

[4] Lin, M. Y., Lee, P. Y., & Hsueh, S. C. (2012). Apriori-based frequent itemset mining algorithms on MapReduce. *Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication*, 1-8. https://doi.org/10.1145/2184751.2184758

[5] Rathee, S. & Kashyap, A. (2018). Adaptive-Miner: an efficient distributed association rule mining algorithm on Spark. *Journal of Big Data*, *5*(1), 1-17. https://doi.org/10.1186/s40537-018-0112-0

[6] Zhang, F., Liu, M., Gui, F., Shen, W., Shami, A., & Ma, Y. (2015). A distributed frequent itemset mining algorithm using Spark for Big Data analytics. *Cluster Computing*, *18*, 1493-1501. https://doi.org/10.1007/s10586-015-0477-1

[7] Al Badani, A. A. & Al Khulaidi, A. G. (2024). Developing an efficient FP Growth algorithm using ordered frequent itemsets matrix for big data. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(3), 508-516. https://doi.org/10.5120/ijais2024451979

[8] Tamba, S. P., Sitanggang, M., & Situmorang, B. C. (2023). Developing an efficient FP Growth algorithm using ordered matrix (OFIM). *International Journal of Intelligent Systems and Applications in Engineering*, *10*(1), 1-10.

[9] Magdy, M., Ghaleb, F. F., El A. Mohamed, D. A., & Zakaria, W. (2023). CC IFIM: An efficient approach for incremental frequent itemset mining based on closed candidates. *The Journal of Supercomputing*, *79*(7), 7877-7899. https://doi.org/10.1007/s11227-022-04976-5

[10] Hikmawati, E., Maulidevi, N. U., & Surendro, K. (2021). Minimum threshold determination method based on dataset characteristics in association rule mining. *Journal of Big Data*, *8*(1), 1-17. https://doi.org/10.1186/s40537-021-00538-3

[11] Bao, F., Mao, L., Zhu, Y., Xiao, C., & Xu, C. (2021). An improved evaluation methodology for mining association rules. *Axioms*, *11*(1), 17. https://doi.org/10.3390/axioms11010017

[12] Al Bana, M. R., Farhan, M. S., & Othman, N. A. (2022). An efficient Spark-based hybrid frequent itemset mining algorithm for big data. *Data*, *7*(1), 11. https://doi.org/10.3390/data7010011

[13] Mokkadem, A., Pelletier, M., & Raimbault, L. (2023). SufRec: An algorithm for mining association rules-Recursivity and task parallelism. *Expert Systems with Applications*, *221*, 121321. https://doi.org/10.1016/j.eswa.2023.121321

[14] Al Bana, M. R., Farhan, M. S., & Othman, N. A. (2022). An efficient Spark-based hybrid frequent itemset mining algorithm for big data. *Data*, *7*(1), 11. https://doi.org/10.3390/data7010011

[15] Chen, J., Yang, S., Ding, W., Li, P., Liu, A., & Zhang, H. (2024). Incremental high average-utility itemset mining: Survey and challenges. *Scientific Reports*, *14*, 60279. https://doi.org/10.1038/s41598-024-60279-0

[16] Xin, J., Liu, Z., & Zhang, Y. (2024). A survey on iHAUIM: incremental high average-utility itemset mining-survey and challenges. *Scientific Reports*, *14*, 60279. https://doi.org/10.1038/s41598-024-60279-0

[17] Ouarem, O., Nouioua, F., & Fournier Viger, P. (2023). A survey of episode mining. *WIREs Data Mining and Knowledge Discovery*, *14*(2), e1524. https://doi.org/10.1002/widm.1524

[18] Krishnan, K. (2023). Association rule mining in distributed environment: a survey. *Business & Information Systems Engineering*, *65*, 101-123. https://doi.org/10.1007/s12599-023-00844-5

[19] Karami Lawal, Z. (2022). Evaluation of Apriori, FP Growth and Eclat association rule mining algorithms. *International Journal of Health Sciences*, *6*(S2), 7475-7485. https://doi.org/10.53730/ijhs.v6nS2.6729

[20] Li, X., Ruan, T., & Sun, X. (2024). A matrix logic approach to efficient frequent itemset discovery in large data sets. *arXiv*. https://doi.org/10.1109/EIECC64539.2024.10929337

[21] Fan, D., W., J., & L., S. (2024). Optimization of frequent item set mining parallelization algorithm based on Spark platform. *Discover Computing*, *27*, 38. https://doi.org/10.1007/s10791-024-09470-5

[22] Chon, K. W. & Kim, C. (2024). GMiner++: Boosting GPU-based frequent itemset mining by reducing redundant computations. *Expert Systems with Applications*, *250*, 123928. https://doi.org/10.1016/j.eswa.2024.123928

[23] Liu, L., Wen, J., Zheng, Z., & Su, H. (2021). An improved approach for mining association rules in parallel using Spark Streaming. *International Journal of Circuit Theory and Applications*, *49*(4), 1028-1039. https://doi.org/10.1002/cta.2935

Contact information:

D. ELAVARASI
(Corresponding Author)
Department of Computer Science and Engineering,
Mount Zion College of Engineering andTechnology,
Pudukkottai, India
E-mail: d.ilavarasi@outlook.com

R. KAVITHA
Department of Information and Technology,
Velammal College of Engineering and Technology,
Madurai, India