Application of Apriori algorithm in commodity association rule analysis

Guangyue Ren
1024041126
Nanjing University of Posts and Telecommunications
School of Computer Science
Nanjing, China

***Abstract*—This paper discusses in detail the application of Apriori algorithm in commodity association rule analysis. In today's data-driven era, retail companies are faced with the challenge of how to extract valuable information from massive transaction data. Association rule mining, as an effective data mining technology, can reveal the potential associations between commodities. The Apriori algorithm is widely used in commodity association rule analysis because of its simplicity and efficiency. This paper first introduces the basic principles and implementation process of the Apriori algorithm, then discusses the optimization method of the algorithm, and evaluates its performance under different parameter settings through experiments. In addition, this paper compares the performance characteristics of the Apriori algorithm and the FP-Tree algorithm, pointing out their respective advantages and disadvantages and applicable scenarios. Finally, this paper summarizes the application value of the Apriori algorithm in commodity association rule analysis and puts forward prospects for future research directions.**

***Index Terms*—Apriori, association rule mining, commodity association rule analysis, data mining, FP-Tree**

I. Introduction

In today's data-driven era, enterprises and organizations are faced with the challenge of how to extract valuable information from massive amounts of data. Especially in the retail industry, understanding the relationship between customer purchase behavior and commodities is crucial to optimizing inventory management, formulating marketing strategies, and improving customer satisfaction. Association rule mining, as an effective data mining technology, can help us discover hidden patterns and laws in data. As one of the classic algorithms for association rule mining, the Apriori algorithm is widely used in commodity association rule analysis because of its simplicity and efficiency. This article will introduce in detail the application of the Apriori algorithm in commodity association rule analysis, including the basic principles of the algorithm, implementation process, optimization methods, and experimental evaluation.

In the retail industry, customers' purchasing behaviors are often not isolated, but have certain correlations. For example, customers who buy bread may also buy milk, and customers who buy diapers may also buy beer. The correlations between these commodities contain rich market information. If they can be accurately mined, they will have an important impact on the business decisions of enterprises. However, traditional data analysis methods are difficult to discover these complex correlations from a large amount of transaction data. The emergence of association rule mining technology provides a new idea to solve this problem. As a classic association rule mining algorithm, the basic idea of Apriori algorithm is to discover association rules between commodities through frequent item sets [1]. Frequent item sets refer to item sets that appear frequently in a data set, and association rules are based on frequent item sets and describe the conditional relationship between item sets [2]. For example, the rule "$A \rightarrow B$" means that under a certain confidence level, customers who buy commodity A are more likely to also buy commodity B. By mining these association rules, enterprises can better understand customers' purchasing habits, thereby conducting targeted marketing activities and increasing sales and customer loyalty.

II. Related Works

In the field of association rule mining, in addition to the Apriori algorithm, there are many other excellent algorithms and research works. The FP-Growth algorithm is a frequent pattern mining algorithm that does not generate candidate item sets [4]. It compresses the data set into a compact form by constructing the FP-Tree data structure, thereby reducing the number of scans of the data set and improving the efficiency of the algorithm. The Eclat algorithm adopts a depth-first search strategy and calculates the support of the item set through the intersection transaction ID, which has a low space complexity. In addition, there are some research works dedicated to optimizing the performance of the Apriori algorithm, such as improving the generation and pruning process of candidate item sets and adopting parallel computing technology. These studies provide a rich theoretical basis and practical guidance for the development of association rule mining technology, and also provide a reference for us to choose appropriate algorithms in practical applications.

III. Problem Statement

Commodity association rule analysis is an important data mining task, and its core goal is to mine frequent item sets and association rules from a large amount of transaction data to provide data support for business decisions [3]. Through this analysis, we can reveal the potential relationship between commodities, such as which commodities are often purchased together (frequent item sets), and whether the purchase of certain commodities will significantly trigger the purchase of another commodity (association rules). This has important guiding significance for fields such as marketing, inventory management, and customer behavior analysis [5]. Specifically, the problem that needs to be solved in this task is how to efficiently extract frequent item sets and association rules that meet the preset threshold conditions from the data set. The two key parameters here are minimum support and minimum confidence:

- Minimum Support: Measures the frequency of an item set in a data set, defined as the ratio of the number of transactions containing the item set to the total number of transactions. Support reflects the importance of an item set, and itemsets with higher support mean that they are more common in the data set. For example, in supermarket transaction data, the support of ”milk” and ”bread” as an item set may be high because they are often purchased together by consumers.
- Minimum Confidence: Used to evaluate the reliability of association rules, defined as the ratio of transactions that also contain the consequent in transactions containing the antecedent. The higher the confidence, the more reliable the rule. For example, the rule ”If you buy bread, you are likely to buy milk” has a high confidence, which means that customers who buy bread usually also buy milk.

A. The challenge of parameter setting

In practical applications, reasonably setting the minimum support and minimum confidence is a balancing process:

- Too high support: may lead to the omission of some important but less frequent patterns. Although such low-frequency patterns are rare, they may have great value to the business, such as the purchase combination of high-profit goods.
- Too low support: will lead to the generation of too many candidate itemsets and frequent itemsets, increase computational complexity and memory overhead, and may even contain many irrelevant noise itemsets.

Similarly, the setting of confidence will directly affect the mining results:

- Too high confidence: will limit the generation of rules, making the mining results too conservative and omitting potential useful rules.
- Too low confidence: may generate a large number of unreliable and meaningless rules, increasing the complexity of subsequent analysis.

Therefore, in order to obtain high-quality mining results, it is necessary to find a reasonable balance between support and confidence. This process often requires combining domain knowledge, understanding the characteristics of the data, and making multiple experimental adjustments.

B. Key issues in algorithm design

In addition to parameter setting, designing an efficient algorithm is another important aspect of solving commodity association rule analysis. Since real transaction data sets usually contain millions or even hundreds of millions of transaction records, how to process such large-scale data becomes a technical challenge. The main issues include:

- Sparsity of data: transaction data is usually very sparse, and the number of items involved in a single transaction is much smaller than the total number of items. This feature requires the algorithm to be able to efficiently process sparse matrices.
- Hugeness of search space: As the number of items increases, the number of potential candidate item sets grows exponentially.
- Real-time mining requirements: In some application scenarios, such as e-commerce recommendation systems, mining algorithms need to generate results in real time to provide customers with instant shopping suggestions.

To address these challenges, commonly used methods include:

- Apriori algorithm: Reduce the number of candidate itemsets through pruning strategies, and only expand the superset of frequent itemsets, which significantly reduces the computational complexity.
- FP-Growth algorithm: Use a compact data structure (FP-Tree) to store transaction information, avoid multiple database scans, and is suitable for processing large-scale data sets.
- Improved optimization algorithm: Combine parallel computing and distributed storage (such as Hadoop and Spark) to further improve processing efficiency.

C. Practical Applications

The application of commodity association rule analysis is very extensive. For example:

- Market basket analysis: helps retailers optimize commodity placement and promotion strategies. For example, put commodities that are often purchased together on adjacent shelves to increase sales.
- Recommendation system: Generate personalized commodity recommendations based on user purchase history to improve user experience.
- Inventory management: Predict changes in demand for related commodities, optimize inventory configuration, and reduce unsalable and out-of-stock phenomena.

In short, commodity association rule analysis not only provides insights into consumer behavior by discovering deep patterns in transaction data, but also provides strong support for enterprises to optimize operational decisions and enhance competitiveness. In the future, with the continuous growth of data scale and continuous improvement of algorithms, research and application in this field will continue to play a greater role.

IV. Algorithms

A. The basic principle of Apriori algorithm

The core idea of the Apriori algorithm is to use the property of frequent itemsets, that is, all non-empty subsets of frequent itemsets must also be frequent. Based on this property, the algorithm generates candidate itemsets and frequent itemsets in an iterative manner. The specific steps are as follows:

- Initialization: Treat each item in the data set as a separate item set to generate the initial candidate item set C_1.
- Generate frequent item sets: Scan the data set, calculate the support of each item set in the candidate item set C_1, retain the item sets with support greater than or equal to the minimum support, and form the frequent item set L_1.
- Generate candidate item sets: Based on the frequent item set L_{k-1}, generate candidate k-item sets C_k through the connection step and pruning step. The connection step is to combine the item sets in L_{k-1} in pairs to generate all possible k-item sets; the pruning step is to remove those k-item sets that contain non-frequent subsets.
- Repeat steps 2 and 3: Continue to scan the data set, calculate the support of each item set in C_k, retain the item sets with support greater than or equal to the minimum support, and form the frequent item set L_k. Repeat this process until no new frequent item sets can be generated.
- Generate association rules: Generate association rules from the final frequent item sets according to the minimum confidence threshold. For each frequent item set X, generate all possible non-empty subsets A, calculate the confidence of the rule $A \Rightarrow (X - A)$, and retain the rules with confidence greater than or equal to the minimum confidence.

B. Implementation process of Apriori algorithm

When implementing the Apriori algorithm, it is necessary to consider the reading, preprocessing, generation of frequent itemsets, and generation of association rules. First, it is necessary to read the data file in CSV format and convert it into a data structure suitable for algorithm processing, such as a transaction list. Then, the data is preprocessed, including removing duplicates and processing missing values, to ensure the accuracy and consistency of the data. Next, according to the set minimum support and minimum confidence, the various steps of the Apriori algorithm are executed to generate frequent itemsets and association rules. Finally, the mined association rules are output in an easy-to-understand form, such as a table or graph, for further analysis and application.

C. Optimization method of Apriori algorithm

Although the Apriori algorithm is efficient in theory, its performance may be limited in practice, especially when dealing with large-scale data sets. In order to improve the efficiency of the algorithm, researchers have proposed a variety of optimization methods:

Reduce the generation of candidate itemsets: Reduce the generation of unnecessary candidate itemsets by improving the strategies of the connection step and the pruning step. For example, the prior knowledge of frequent itemsets can be used to determine in advance that some itemsets are unlikely to be frequent, thereby avoiding the generation of these itemsets.

Use data structure optimization: Use efficient data structures to store candidate itemsets and frequent itemsets to reduce memory usage and access time. For example, hash tables can be used to quickly find and update the support of itemsets.

Parallel computing: Split the dataset into multiple subsets, perform calculations on different processors or computers, and then merge the results. This method can significantly improve the processing speed of the algorithm, especially in multi-core processors and distributed computing environments.

Dynamically adjust support and confidence: Dynamically adjust the values of minimum support and minimum confidence according to the characteristics of the dataset and the actual situation during the mining process. For example, after initially mining some frequent itemsets, the support and confidence can be appropriately reduced according to the distribution of these itemsets and the quality of association rules to discover more potential rules.

D. Comparison between Apriori and FP-Tree algorithms

Both the Apriori algorithm and the FP-Tree algorithm are classic algorithms for frequent item set mining, but they have some differences in implementation principles, performance characteristics, and applicable scenarios.

The Apriori algorithm is based on the strategy of candidate item set generation and pruning, and generates candidate item sets and frequent item sets one by one in an iterative manner. In each iteration, the entire data set needs to be scanned to calculate the support of the candidate item set, and then pruned according to the minimum support threshold to retain the frequent item sets. This process will gradually reduce the number of frequent item sets as the length of the item set increases until no new frequent item sets can be generated. The FP-Tree algorithm uses a data structure called FP-Tree (Frequent Pattern Tree) to compress the data set, thereby avoiding the overhead of multiple scans of the data set. FP-Tree is a tree structure in which each node represents an item and the paths between nodes represent item sets. The algorithm first scans the data set to build an FP-Tree, and then mines frequent item sets by depth-first searching the FP-Tree. During the search process, the algorithm calculates the support and generates frequent item sets based on the frequency of occurrence and path information of the item set.

The advantage of the Apriori algorithm is that it is simple to understand and relatively easy to implement. However, due to the need to scan the data set multiple times, its time complexity is high, which may lead to inefficiency, especially when dealing with large-scale data sets. In addition, as the number of candidate item sets increases, the space complexity of the algorithm will also increase accordingly, especially in the case of low support, a large number of candidate item sets may be generated. However, the advantage of the FP-Tree algorithm is its high efficiency. By constructing the FP-

TABLE I
EXPERIMENTAL RESULTS OF APRIORI ALGORITHM

min support	min confidence	Association Rules of Apriori Algorithm				
		antecedents	*consequents*	*support*	*confidence*	*lift*
0.01	0.5	(3.0)	(2.0)	0.025000	0.514286	6.612245
		(69.0)	(2.0)	0.012500	0.642857	8.265306
		(24.0)	(6.0)	0.015278	0.578947	9.694002
		(131.0)	(7.0)	0.011111	0.800000	16.457143
		(10.0)	(9.0)	0.016667	0.705882	18.823529
0.015	0.5	(3.0)	(2.0)	0.025000	0.514286	6.612245
		(24.0)	(6.0)	0.015278	0.578947	9.694002
		(10.0)	(9.0)	0.016667	0.705882	18.823529
0.015	0.1	(2.0)	(3.0)	0.025000	0.321429	6.612245
		(3.0)	(2.0)	0.025000	0.514286	6.612245
		(24.0)	(6.0)	0.015278	0.578947	9.694002
		(6.0)	(24.0)	0.015278	0.255814	9.694002
		(9.0)	(10.0)	0.016667	0.444444	18.823529
		(10.0)	(9.0)	0.016667	0.705882	18.823529
0.02	0.1	(2.0)	(3.0)	0.025000	0.321429	6.612245
		(3.0)	(2.0)	0.025000	0.514286	6.612245
0.02	0.5	(3.0)	(2.0)	0.025000	0.514286	6.612245

Tree, the data set can be compressed into a compact form, reducing the number of times the data set is scanned, thereby significantly improving the efficiency of the algorithm. In addition, the FP-Tree algorithm also has advantages in terms of space complexity, because it does not need to store a large number of candidate item sets, but directly mines frequent item sets in the FP-Tree. However, the implementation of the FP-Tree algorithm is relatively complex, and it is necessary to build and maintain the FP-Tree data structure, which may be difficult for beginners.

The Apriori algorithm is suitable for scenarios with small data sets, short item set lengths, and low requirements for algorithm efficiency. For example, in the sales data analysis of some small retail stores, the Apriori algorithm can quickly mine the association rules between products to help companies understand customers' purchasing habits. Instead, the FP-Tree algorithm is more suitable for processing large-scale data sets and complex association rule mining tasks. For example, in the product recommendation system of a large e-commerce platform, the FP-Tree algorithm can efficiently mine the association relationship between products from massive user purchase data and provide users with personalized product recommendations.

Overall, the Apriori algorithm and the FP-Tree algorithm each have their own advantages and disadvantages. In practical applications, it is necessary to select the appropriate algorithm based on the characteristics of the data set and the requirements of the mining task. For some simple application scenarios, the Apriori algorithm can meet the needs; while for complex tasks that require efficient processing of large-scale data sets, the FP-Tree algorithm has more advantages.

V. RESULTS AND EVALUATION

In order to evaluate the application effect of Apriori algorithm in commodity association rule analysis, we conducted experiments and tested different parameters on the data set.

A. Results Analysis

The experimental results show that Apriori algorithm can effectively mine frequent item sets and association rules that meet the minimum support and minimum confidence. When the support is 1% and the confidence is greater than 50%, the algorithm can find some common commodity combinations, such as "bread and milk", "diapers and beer", etc. These rules have high support and confidence and are in line with the actual situation. When the support is increased to 1.5%, the number of rules mined by the algorithm decreases, but some meaningful associations can still be found. In addition, by adjusting the confidence threshold, more reliable rules can be further screened out. For example, when the support is 1.5% and the confidence is greater than 10%, although the number of rules is large, it contains some accidental associations; when the support is 1.5% and the confidence is greater than 50%, the number of rules is relatively small, but the quality is high and has more practical application value.

Table I shows the experimental results of association rules mined using the Apriori algorithm at different min support and min confidence thresholds. The results of each association rule include: antecedents, consequences, support, confidence, and lift. The following is a detailed analysis of the experimental results:

When the minimum support is 0.01, the system mines a large number of rules. For example, the association rule between the antecedent (3.0) and the consequence (2.0) has a support of 0.025, a confidence of 0.514, and a lift of 6.612. The support of these rules is relatively low, indicating that they appear less frequently in the entire data set, but their confidence and lift are relatively high, indicating that when the antecedent appears, the probability of the consequent appearing is high, and the association strength is strong. As the minimum support increases (e.g., 0.015 and 0.02), the number of rules mined decreases significantly, but the support and confidence remain relatively stable. This shows that when the

TABLE II
COMPARISON OF TIME COMPLEXITY BETWEEN APRIORI AND FP-GROWTH ALGORITHMS

algorithm	Time complexity of frequent itemset mining	Time complexity of association rule generation	Applicable situations
Apriori	O(k * n * m)	O(f * m)	Small-scale data or small item sets
FP-Growth	O(n * m)	O(f * m)	Large-scale and sparse data

support is set to a higher value, the number of rules decreases, but the importance of each rule in the dataset (measured by support and confidence) is relatively higher.

In the experiment, when the minimum confidence is set to 0.5, the confidence of all rules is greater than or equal to 0.5. For example, when the association rule of the antecedent (3.0) and the consequent (2.0) has a support of 0.025, a confidence of 0.514, and a lift of 6.612, the rule meets the higher confidence requirement. When the minimum confidence is set to 0.1, more rules are mined, but their confidence is generally low. For example, the rule of the antecedent (6.0) and the consequent (24.0) has a low support (0.015278), but the confidence of the rule does not meet the higher standard due to the low confidence (0.255814).

Lift is an important indicator to measure the strength of association rules. The lifts in the table range from 6.612 to 18.823, which varies with the changes in support and confidence. For example, the rule of the antecedent (9.0) and the consequent (10.0) has a support of 0.016667, a confidence of 0.705882, and a lift of 18.823. This shows that the rule not only has a high confidence, but also has a high lift in the data set, indicating that the rule has a strong association. In contrast, the rule of the antecedent (3.0) and the consequent (2.0), although it has high support and confidence, has a relatively low lift of 6.612, which indicates that the relevance of this rule is not as strong as some other rules.

The rules in the table not only show the relationship between the antecedent and the consequent, but also reveal the diversity of the rules. For example, the rule of the antecedent (24.0) and the consequent (6.0) has a significant difference in confidence and lift compared to the rule of the antecedent (2.0) and the consequent (3.0), although they have the same support (0.025000). This shows that although the basic statistics of the rules (such as support) are similar, their reliability and strength of association may vary greatly.

B. Comparative experiment

To evaluate Apriori, we set the FP-Growth algorithm [6] to experiment on the dataset. Due to data set limitations, the results of the FP-Growth algorithm and the Apriori algorithm are the same, but we still compare and analyze the two algorithms, such as time complexity.

1) Apriori Algorithm Evaluation: For generating candidate item sets : In each iteration, Apriori generates new candidate itemsets based on the frequent itemsets found in the previous round. For example, if a k-itemset is generated in round k, the candidate k-itemset needs to be generated from the (k-1)-itemset. For each round of candidate itemsets, Apriori scans the entire database once to calculate their support. This means that the time complexity of each round is roughly O(n * m), where n is the number of transactions and m is the size of the current candidate itemset. The number of iterations of Apriori depends on the maximum itemset size in the dataset. In theory, in the worst case, the number of iterations is at most min(k, m), where k is the number of items in the database and m is the maximum size of the most frequent itemset. Therefore, in the worst case, the time complexity of Apriori can be expressed as O(k * n * m), where k is the size of the dataset (number of items), n is the number of transactions, and m is the size of the frequent itemsets.

Taking all factors into consideration, the time complexity of the Apriori algorithm is O(k * n * m), where: *k* is the number of items (the number of possible item sets in each transaction), *n* is the number of transactions (i.e. the size of the database), *m* is the size of each candidate item set.

For calculating association rules : Once the frequent itemsets are generated, the Apriori algorithm computes association rules based on these frequent itemsets. The time complexity of rule generation is usually O(f * m), where f is the number of frequent itemsets and m is the maximum size of each itemset.

2) FP-Growth Algorithm Evaluation: For constructing FP-Tree : First, FP-Growth scans the data twice: The first scan is used to calculate the frequency of each item and sort the items from high to low by support.The second scan is used to construct FP-Tree. FP-Tree is a compact data structure that represents item sets by sharing common prefixes, thus avoiding the candidate item set generation step in Apriori.The time complexity of constructing FP-Tree is O(n * m), where n is the number of transactions and m is the number of items in each transaction (i.e., transaction width).

For mining frequent itemsets : After constructing the FP-Tree, FP-Growth will recursively extract frequent itemsets from the FP-Tree. The time complexity of this process is related to the depth and branching factor of the tree. Generally, FP-Tree is much more efficient than Apriori due to its compressed structure. In the worst case, the time complexity of recursive mining of frequent itemsets by FP-Tree is O(n * m), where n is the number of transactions and m is the number of items.

For calculating association rules : Similar to the Apriori algorithm, FP-Growth will also generate association rules based on frequent itemsets after mining them. The time complexity of generating rules is usually O(f * m), where f is the number of frequent itemsets and m is the maximum number of items in each frequent itemset.

3) Summary and comparison: The result is shown in tab IV-D, Apriori relies on candidate item set generation and database scanning. As the number of transactions and item set size increases, the time complexity increases exponentially, especially on high-dimensional data and sparse data sets. FP-Growth uses a compressed data structure (FP-Tree) to avoid the generation of candidate item sets, and has higher computational efficiency, especially on large-scale and high-dimensional data sets, it can significantly reduce computational complexity. For large-scale data, FP-Growth is usually a better choice, especially in the case of sparse and high-dimensional data, which can effectively reduce the computational burden. Apriori is suitable for cases with smaller or simpler data.

VI. Conclusion

As a classic association rule mining algorithm, the Apriori algorithm has important application value in commodity association rule analysis. It can effectively mine frequent item sets and association rules from transaction data, help companies understand the relationship between customers' purchasing behavior and commodities, and thus provide support for business decisions. However, the Apriori algorithm also has some limitations, such as low efficiency and high space complexity when processing large-scale data sets. In order to overcome these limitations, researchers have proposed a variety of optimization methods and improved algorithms, such as the FP-Tree algorithm. In future research, we can further explore how to combine the advantages of multiple algorithms to improve the efficiency and accuracy of association rule mining to meet the needs of different application scenarios.

References

[1] KrishnaKumar A, Amrita D, Priya N S. Mining association rules between sets of items in large databases[J]. International Journal of Science and Modern Engineering (IJISME), ISSN, 2013: 2319-6386.

[2] Agrawal R. Fast Algorithms for Mining Association Rules[C]. VLDB, 1994.

[3] Ordonez C, Santana C A, De Braal L. Discovering Interesting Association Rules in Medical Data[C]//ACM SIGMOD workshop on research issues in data mining and knowledge discovery. 2000: 78-85.

[4] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation[J]. ACM sigmod record, 2000, 29(2): 1-12.

[5] Yuan X. An improved Apriori algorithm for mining association rules[C]//AIP conference proceedings. AIP Publishing, 2017, 1820(1).

[6] Zaki M J, Hsiao C J. Efficient algorithms for mining closed itemsets and their lattice structure[J]. IEEE transactions on knowledge and data engineering, 2005, 17(4): 462-478.