Market Basket Analysis through Frequent Itemset Mining

Zenan Bai(白泽楠)
1024040801
Nanjing University of Posts and Telecommunications
Nanjing, China

***Abstract*—Analyzing customers' shopping records helps product sellers discover customer needs and preferences, and develop appropriate strategies to gain more profits. This paper adopts two frequent itemset mining methods, namely Apriori algorithm and FP growth algorithm, for market basket analysis. Using support degree, confidence degree, and lift degree as evaluation criteria, the frequent patterns and association rules in the given dataset are mined. At the same time, the performance of the two algorithms is compared, and the results show that for the same dataset, The running time of FP growth algorithm is shorter than that of Apriori algorithm, and FP growth algorithm has better performance.**

***Index Terms*—market basket analysis; Apriori; FP-growth; frequent pattern mining**

I. Introduction

As an indispensable part of modern society, supermarkets provide people with daily necessities. Every day, a massive number of customers go to supermarkets to purchase the products they need, generating a massive amount of shopping data. For supermarket managers, if they can conduct detailed analysis of customers' market basket data, discover the actual needs and preferences of customers, they will be able to develop appropriate sales strategies, which is conducive to increasing product sales and obtaining more profits. Therefore, mining information from shopping data is both meaningful and rewarding and deserves to be investigated.

Relevant research on market basket analysis has been ongoing for many years, and currently a common practice is to utilize data mining techniques for data analysis. Through data mining, we can extract frequent patterns hidden in the data, such as itemsets, subsequences and so on. The collection of goods that appear simultaneously in the trading dataset is called a frequent itemset. On the basis of frequent patterns, we can further explore the associations, correlations, and many other interesting connections between data to obtain more valuable knowledge. Therefore, frequent pattern mining is an important task in data mining. This paper will focus on the correlation analysis between products in the shopping basket dataset, with frequent pattern mining techniques employed.

II. Related Works

Researchers have proposed many methods for frequent itemset mining so far. In 1994, Agrawal and Srikant proposed the Apriori algorithm [2] and introduced a method for generating association rules from frequent itemsets in [3].

Subsequently, some variants of the Aprori algorithm were proposed one after another. In 1995, Park, Chen, and Yu [4] used hash tables to improve the efficiency of association rule mining. Savacere, Omicinski, and Navath [5] proposed partitioning techniques. In 1997, Brin, Motwani, Ullman, and Tsur [6] proposed a dynamic itemset counting method. Under the Apriori framework, parallel and distributed association rule mining was studied by Park, Chen, and Yu [7], Agrawal and Shafer [8], Cheung, Han, Ng et al. [9].

In addition, some researchers have proposed frequent itemset mining methods that are different from the Apriori method. FP growth is a pattern growth method that mines frequent patterns without generating candidates, proposed by Han, Pie, and Yin [10] in 2000. Eclat is a method proposed by Zaki [11] for mining frequent itemsets by exploring vertical data formats.

III. Problem Statement

Given a dataset stored in CSV file format, which contains 1892 user shopping data records, each record contains the corresponding item number purchased by the customer. Some of the shopping records in the CSV file are shown in the table below.

TABLE I
Some Records in the Dataset

Record ID	Items purchased
1	1332,971,167,60,9,75,700,349
2	972,35,1843,1844,973,117,1566,583,71
3	262,101,41,1250,15,744,29,1176,89,405,76
4	29,262,406,9,61,863,675,1033,10,1103

For instance, in the first shopping record in the table above, the customer bought 8 items in total, including item1332, item 971, item167, item60, item9, item75, item700 and item349.

Our task is to search for frequently purchased product combinations, i.e. frequent itemsets in the dataset, and analyze the correlations between products to discover purchasing patterns where products are purchased simultaneously. We formalize these purchasing patterns through association rules.

IV. Algorithms

The mining of association rules from the given dataset is divided into two steps: First, we adopt two different methods, namely Apriori Algorithm and FP-growth, to find frequent itemsets first. Second, association rules can be generated by making use of the frequent itemsets discovered. Before introducing the two methods, we need to clarify some basic concepts and terminology.

A. Basic Concepts

Association rule is a rule in the form of "$A \Rightarrow B$", where A and B are two nonempty itemsets, and A and B are disjoint itemsets, i.e. $A \cap B = \varnothing$. It indicates that in the dataset, when itemset A appears, itemset B also tends to appear.

The evaluation criteria for association rules are support degree, confidence degree, and lift degree.

Support degree refers to the frequency of a pattern appearing in a dataset, usually expressed as the ratio of the number of times the pattern appears to the total number of records in the dataset. The support degree for rule "$A \Rightarrow B$" is the percentage of records in the dataset that contain $A \cup B$, which is the probability $P(A \cup B)$, i.e:

$$\mathit{Support}(A \Rightarrow B) = P(A \cup B) \tag{1}$$

Confidence degree can be used to measure the reliability of association rules. The confidence degree of rule "$A \Rightarrow B$" refers to the proportion of records containing both itemset A and itemset B. Actually, confidence degree represents the conditional probability $P(B \mid A)$, and its calculation formula is:

$$Confidence(A \Rightarrow B) = \frac{\mathit{Support}(A \cup B)}{\mathit{Support}(A)} \tag{2}$$

In the above equation, $\mathit{Support}(A \cup B)$ represents the proportion of records that appear simultaneously in itemsets A and B to the total number of records, and $\mathit{Support}(A)$ represents the proportion of records that appear in itemset A to the total number of records.

Lift degree is used to measure the correlation between itemsets in association rules, and it can evaluate the practicality of the rules. The lift degree of rule "$A \Rightarrow B$" refers to the ratio of the likelihood of including itemset A and the likelihood of itemset B appearing without this condition. Specifically, the formula for calculating the lift degree is:

$$Lift(A \Rightarrow B) = \frac{P(A \cup B)}{P(A)P(B)} \tag{3}$$

In (3), if its lift degree is less than 1, the occurrence of A is negatively correlated with the occurrence of B, meaning that one occurrence may cause the other not to occur. If the value of equation (3) is greater than 1, then A and B are positively correlated, meaning that each occurrence implies another occurrence. If the result value is equal to 1, then A and B are independent and there is no correlation between them.

Usually, when mining association rules, we need to predefine a minimum support threshold $\mathit{min\_sup}$ and a minimum confidence threshold min_conf. If a rule satisfies both the minimum support threshold and the minimum confidence threshold, and its lift degree is greater than 1, such a rule is called a strong rule, which is also our goal in data mining in the user shopping record dataset.

B. Apriori Algorithm

The Apriori algorithm is a classic algorithm for frequent itemset mining and association rule learning in data mining. It is widely used to discover interesting relationships between variables in large databases.

The core idea of the Apriori algorithm is based on the property of frequent itemsets: "All subsets of a frequent itemset must also be frequent." This property is used to reduce the number of candidate itemsets that need to be generated and tested, thereby improving the efficiency of the algorithm.

The steps of Apriori Algorithm are as follows:

- *Generate 1-Frequent Itemsets* L_1: First, scan the dataset to count the frequency of each individual item. And then identify the items whose frequency is greater than or equal to the minimum support threshold. These items form the 1-frequent itemsets L_1.
- *Generate k-Candidate Itemsets* C_k: This step includes two substeps, namely join step and prune step. In join substep, we generate k-candidate itemsets C_k by joining $(k-1)$-frequent itemsets L_{k-1} with themselves. Specifically, take two $(k-1)$-frequent itemsets and combine them to form a k-itemset. In prune step, we reduce the number of candidate itemsets by checking if all $(k-1)$-subsets of the generated k-itemset are in $(k-1)$-frequent itemsets. If any $(k-1)$-subset is not in $(k-1)$-frequent itemsets, the k-itemset is pruned.
- *Calculate the Support of k-Candidate Itemsets*: Scan the dataset again to count the frequency of each k-candidate itemset. Identify the itemsets whose frequency is greater than or equal to the minimum support threshold. These itemsets form the k-frequent itemsets L_k.
- *Repeat the second step and the third step*: Continue generating k-candidate itemsets and calculating their support until no new frequent itemsets can be generated.

The Apriori algorithm is straightforward and easy to understand and implement. And it is applicable to vari-

Algorithm 1 Apriori Algorithm

```
Input: Transaction database $D$, Minimum support threshold $min\_sup$
Output: Frequent itemsets $L$
$L_1 \leftarrow$ {frequent 1-itemsets} {Initial pass}
$k \leftarrow 2$
while $L_{k-1} \neq \varnothing$ do
    $C_k \leftarrow$ apriori_gen($L_{k-1}$) {Generate candidate itemsets}
    for all transactions $t \in D$ do
        $C_t \leftarrow \{c \in C_k \mid c \subseteq t\}$ {Find candidates contained in $t$}
        for all candidate itemsets $c \in C_t$ do
            $c.count \leftarrow c.count + 1$ {Increment support count}
        end for
    end for
    $L_k \leftarrow \{c \in C_k \mid c.count \geq min\_sup\}$ {Filter frequent itemsets}
    $k \leftarrow k + 1$
end while
return $L \leftarrow \bigcup_k L_k$
```

ous types of datasets, especially medium and small-scale datasets.

However, Apriori algorithm also has some disadvantages due to its high computation complexity and multiple scans of the dataset. For large datasets, generating and testing candidate itemsets can be computationally expensive, leading to performance bottlenecks. What's more, the algorithm requires multiple scans of the dataset, which can be inefficient, especially for very large datasets.

C. FP-growth

Similar to Apriori algorithm, the FP-growth algorithm is also a popular and efficient method for frequent itemset mining, which is used to discover frequent patterns in large datasets. It is particularly useful for market basket analysis and other applications where understanding the relationships between items is important.

The principle of the FP-growth algorithm is to avoid the expensive process of generating candidate itemsets, which is a major drawback of the Apriori algorithm. Instead, FP-growth uses a compact data structure called the FP-tree to store the transactions and extract frequent itemsets directly from it.

The algorithm can be summarized in two main steps:

- *FP-Tree Construction*: In the first pass, scan the dataset to count the frequency of each item and discard infrequent items. And then sort the frequent items in decreasing order of their support. In the second pass, construct the FP-tree by inserting each transaction into the tree. The tree is built in such a way that common prefixes are shared, and each node in the tree represents an item and has a counter to keep track of the number of times the item appears in the transactions. Pointers are maintained between nodes containing the same item, creating singly linked lists.
- *Frequent Itemset Generation*: In this step, the algorithm starts from the leaves of the FP-tree and moves towards the root. It uses a divide-and-conquer strategy to find frequent itemsets. For each item, extract the prefix path sub-trees ending in that item. These sub-trees are called conditional pattern bases. Recursively mine each conditional pattern base to find frequent itemsets. The solutions are then merged to form the final set of frequent itemsets. For each frequent item, construct a new FP-tree (conditional FP-tree) using the transactions that contain the item. This process is repeated until no more frequent itemsets can be found.

The FP-growth algorithm is more efficient than Apriori because it avoids the generation of candidate itemsets and reduces the number of database scans. With regard to memory usage, the FP-tree is a compact representation of the dataset, which can fit into memory, making it suitable for large datasets. In addition, the FP-growth has good scalability and the algorithm can handle large and high-dimensional datasets effectively.

D. Generation of Association Rules

Once we identify frequent itemsets from the dataset, we can directly generate strong association rules from them. The generation of association rules is as follows:

- For each frequent itemset l, generate all nonempty subsets of l.
- For each nonempty subset s of l, if $Confidence(s \Rightarrow l - s)$ is greater than or equal to the minimum confidence threshold min_conf, and $Lift(s \Rightarrow l - s)$ is greater than 1, then output association rule "$s \Rightarrow l - s$".

Since the rules are generated by frequent itemsets, each rule automatically satisfies the minimum support threshold min_sup.

V. Evaluation

A. Frequent Itemsets

Setting the minimum support threshold min_sup to 0.01, we can find a total of 335 frequent itemsets by using Apriori algorithm or FP-growth. We selected the top 10 frequent itemsets with the highest frequency of occurrence in the frequent itemsets, as shown in the Fig. 1.

From the Fig. 1, it can be seen that the itemsets with the highest frequency of occurrence in the dataset are all 1-frequent itemsets, namely {1}, {2}, {3}, {6}, {7}, {8}, {5}, {9}, {19}, {11}. Among them, the itemset with the highest frequency of occurrence is {1}, with a support count of 189.

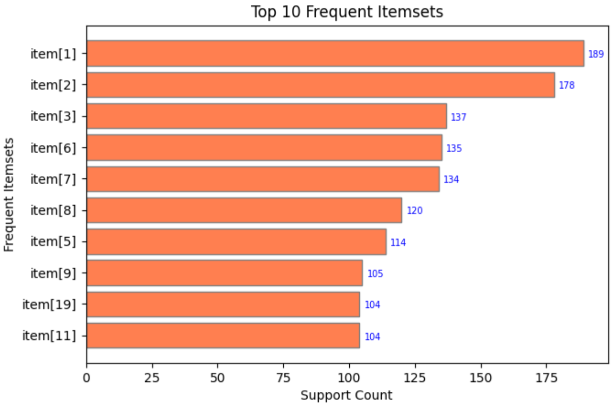

Fig. 1. Some frequent itemsets with the highest frequency

B. Association Rules

When we set the minimum support threshold min_sup to 0.01 and the minimum confidence threshold min_conf to 0.7, a total of 15 association rules can be mined using Apriori algorithm or FP-growth. These association rules and their corresponding support, confidence, and improvement are shown in the table II.

TABLE II
ASSOCIATION RULES GENERATED WHEN $min\_sup = 0.01$ AND $min\_conf = 0.7$

Association Rules	Support Degree	Confidence Degree	Lift Degree
{131}⇒{7}	0.018	0.773	10.910
{147}⇒{6}	0.013	0.735	10.305
{202}⇒{2}	0.010	0.704	7.480
{55}⇒{44}	0.015	0.725	29.820
{43}⇒{2}	0.015	0.707	7.518
{50}⇒{1}	0.012	0.742	7.427
{185}⇒{57}	0.011	0.700	33.959
{86}⇒{27}	0.012	0.920	41.444
{2,71}⇒{3}	0.016	0.750	10.358
{3,71}⇒{2}	0.016	0.750	7.972
{2,69}⇒{3}	0.014	0.788	10.881
{3,69}⇒{2}	0.014	0.788	8.375
{3,43}⇒{2}	0.011	0.800	8.503
{2,139}⇒{3}	0.010	0.760	10.496
{3,139}⇒{2}	0.010	0.864	9.179

From the table, it can be seen that rules "$\{2, 71\} \Rightarrow \{3\}$" and "$\{3, 71\} \Rightarrow \{2\}$" have the highest support degree, both with a support degree of 0.016. Rule "$\{86\} \Rightarrow \{27\}$" has the highest confidence degree, with a confidence degree of 0.92. The rule "$\{185\} \Rightarrow \{57\}$" has the highest lift degree, with a lift degree of 33.959, indicating that item185 and item57 have the greatest correlation. Managers can consider these two products together when selling goods.

In addition, if we want to discover more interesting rules, we can appropriately lower the minimum support threshold min_sup or minimum confidence threshold min_conf to obtain more detailed information from the dataset.

C. Comparison of Two Algorithms

In order to explore the performance of Apriori algorithm and FP-growth algorithm in mining frequent itemsets and association rules, we set the minimum confidence threshold min_conf to a constant value of 0.7, and set different minimum support thresholds min_sup in the interval $[0.005, 0.02]$. We tested the running time of the two algorithms separately and plotted the test results in a line graph, as shown in Fig. 2.

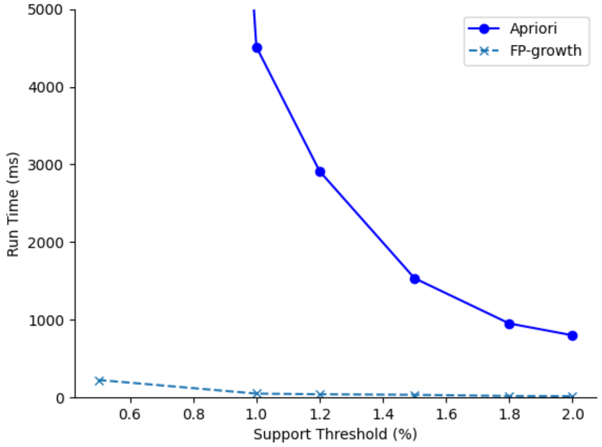

Fig. 2. Comparison of running time between two algorithms

From Fig. 2, it can be seen that the running time of the FP-growth algorithm is much shorter than that of the Apriori algorithm, indicating that compared with the Apriori algorithm, the FP-growth algorithm has less time overhead, higher execution efficiency, and better performance. The smaller the minimum support threshold min_sup, the more obvious the advantages of FP-growth algorithm. If you want to mine a large number of frequent patterns, FP-growth algorithm will be a better choice.

VI. CONCLUSION

This paper investigates the issues related to market basket analysis from the perspective of frequent itemset mining. By using two algorithms, namely Apriori algorithm and FP-growth algorithm, the frequent patterns and association rules of a given dataset are mined, and the performance of the two algorithms is compared. The results show that FP-growth algorithm has better performance. Some shortcomings are that, due to limited practicality, this article failed to introduce more frequent pattern mining algorithms for relevant analysis and comparison. In the future, we will consider using more frequent pattern mining algorithms for shopping basket analysis to make the performance comparison of algorithms more comprehensive and objective, and try to improve existing algorithms.

VII. Acknowledgment

Finally, I would like to express my heartfelt gratitude to Professor Zou Zhiqiang. Prof. Zou's big data course covers rich and profound content, and his attentive teaching in class has benefited me greatly. After a semester of course learning, I have gained a lot of knowledge and significantly broadened my academic horizons. Prof. Zou's serious scientific attitude, rigorous academic spirit, and work style of striving for excellence in class left a deep impression on me and gave me great encouragement.

References

[1] J. Han, M. Kamber, and J. Pei, "Data Mining: Concepts and Techniques," China Machine Press, Third Edition, 2012.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," In Proc. 1994 Int. Conf. Very Large Data Bases (VLDB'94), pp. 487–499, Santiago, Chile, Sept. 1994.

[3] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," In Research Report RJ 9839, IBM Almaden Research Center, San Jose, CA, June 1994.

[4] J. S. Park, M. S. Chen, and P. S. Yu, "An effective hash-based algorithm for mining association rules," In Proc. 1995 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'95), pp. 175–186, San Jose, CA, May 1995.

[5] A. Savasere, E. Omiecinski, and S. Navathe, "An effective algorithm for mining association rules in large databases," In Proc. 1995 Int. Conf. Very Large Data Bases (VLDB'95), pp432–443, Zurich, Switzerland, Sept. 1995.

[6] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur, "Dynamtic itemset counting and implication rules for market basket analysis," In Proc. 1997 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'97), pp. 265–276, Tucson, AZ, May 1997.

[7] J. S. Park, M. S. Chen, and P. S. Yu, "Efficient parallel mining for association rules," In Proc. 4th Int. Conf. Information and Knowledge Management, pp. 31–36, Baltimore, MD, Nov. 1995.

[8] R. Agrawal and J. C. Shafer. "Parallel mining of association rules: Design, implementation, and experience," IEEE Trans. Knowledge and Data Engineering, 8:962–969, 1996.

[9] D. W. Cheung, J. Han, V. Ng, A. Fu, and Y. Fu, "A fast distributed algorithm for mining association rules," In Proc. 1996 Int. Conf. Parallel and Distributed Information Systems, pp 31–44, Miami Beach, FL, Dec. 1996.

[10] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," In Proc. 2000 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'00), pp. 1–12, Dallas, TX, May 2000.

[11] M. J. Zaki, "Scalable algorithms for association mining," IEEE Trans. Knowledge and Data Engineering, 12:372–390, 2000.