




# Using Multiple Minimum Support to Auto-adjust the Threshold of Support in Apriori Algorithm

Azzeddine Dahbi<sup>(✉)</sup> , Youssef Balouki, and Taoufiq Gadi

Laboratory Informatics, Imaging and Modelling of Complex Systems (LIIMSC),  
Faculty of Science and Technology, Hassan 1st University Settat,  
Settat, Morocco  
azzdine.im@gmail.com

**Abstract.** Nowadays, Data mining becomes an important research domain, aiming to extract the interesting knowledge and pattern from the large databases. One of the most well-studied data mining tasks is association rules mining. It discovers and finds interesting relationships or correlations among items in large databases. A great number of algorithms have been proposed to generate the association rules, one of the main problems related to the discovery of these associations (that a decision maker faces) is the choice of the threshold of the minimum support because it influences directly the number and the quality of the discovered patterns. To bypass this challenge, we propose in this paper an approach to determine how to auto-adjust the minimum support threshold according to data by using a multiple minimum support. The experiments performed on benchmark datasets show a significant performance of the proposed approach.

**Keywords:** Data mining · Association rules mining · APRIORI algorithm  
Multiple minimum support

## 1 Introduction

Data mining technologies aim to explore a valuable knowledge in large volumes of data [1]. There are many data mining methods and algorithms, One of the most traditional data mining approaches is finding frequent item-sets in transactional databases, and deduct their corresponding association rules. Currently, there are many proposed algorithms for mining association rules. The most known and the simplest one is the APRIORI Algorithm [2] proposed in 1993 by Agrawal. The use of APRIORI algorithm in DM makes it possible to test the various combinations between the items (Data\_Attributes) to find potential relationships which will be exposed in the form of association rules. However, the rules produced by APRIORI are judged by known measures (support and confidence). But this algorithm suffers from an important defect which cannot determine the minimal value of support and confidence and these parameters are estimated intuitively by the users. Depending on the choice of those thresholds, association rule mining algorithms can generate a huge number of rules

which lead algorithms to suffer from long execution time and large memory consumption, or may generate a small number of rules, and thus may delete valuable information.

This method can also offers several rules in a massive database, millions, which apparently many of them are not useful and helpful; it can be implied that it doesn't have enough efficiency. So we require a method to find the best values of support parameter automatically especially in large databases. The main goal of this paper is to present a method to find proper values of minimum threshold for efficient support.

The outline of our paper is as follows: In Sect. 2, we present the necessary scientific background and an overview of association rules mining, and related works. Part 3 presents our proposed approach based on APRIORI algorithm for mining association rules with auto-adjust the threshold of support with multiple minimum support. In Sect. 4, we discuss the experimental results and its analysis. The conclusion and scope for future work is given in the last section.

## 2 An Overview of AR Mining

### 2.1 Process Association Rules Mining

#### Association Rules Mining

We define  $I = \{i_1, i_2, \dots, i_n\}$  as a set of all items, and  $T = \{t_1, t_2, \dots, t_m\}$  as a set of all transactions, every transaction  $t_i$  is an itemset and meets  $t_i \subseteq I$ . Association rules can be generated from large (frequent/closed/maximal) itemsets. An association rule is an implication expression of the form  $X \rightarrow Y$ ,  $X \subseteq I$ ,  $Y \subseteq I$  where  $X$  and  $Y$  disjoint itemsets (i.e.  $X \cap Y = \emptyset$ ).  $X$  is called the antecedent and  $Y$  is called the consequent of the rule.

The force of an association rule can be measured in terms of its support and confidence. The support of the rule  $X \rightarrow Y$  is the percentage of transactions in database  $D$  that contain  $X \cup Y$  and is represented as:

$$\text{Support}(X \rightarrow Y) = P(XY) = n(XUY)/n \quad (1)$$

The confidence of a rule  $X \rightarrow Y$  describes the percentage of transactions containing  $X$  which also contain  $Y$  and is represented as

$$\text{Confidence}(X \rightarrow Y) = n(XUY)/n(X) = P(XY)/P(X) \quad (2)$$

Where  $n(X \cup Y)$  is the number of transactions that contain items (i.e.  $XUY$ ) of the rule,  $n(X)$  is the number of transactions containing itemset  $X$  and  $n$  is the total number of transactions.

The process of mining association rules is to discover all association rules from the transactional database  $D$  that have support and confidence greater than threshold pre-defined by the user minimum support (minsup) and minimum confidence (minconf).

### APRIORI Algorithm

Now, diverse algorithms for mining association rules are proposed. The most known, and without certainly the simplest one is the APRIORI algorithm [2]. It scans the mesh of the concepts width, such as Charm [3] and Closet [4] algorithms. Other travel the lattice depth is particularly the case for algorithms FP-Growth [5] and Eclat [6].

The APRIORI algorithm works in two steps:

- Find the frequent itemset: The frequent itemset is an itemset that verifies a predefined threshold of minimum support.
- Generate all strong association rules from frequent itemsets: The strong association rule is a rule that verifies a predefined threshold of minimum confidence.

APRIORI algorithm is the most powerful method that candidate  $k + 1$ -itemsets may be generated from frequent  $k$ -itemsets according to the principle of APRIORI algorithm that any subset of frequent itemsets are all frequent itemsets.

Foremost, find the frequent 1-itemsets  $L_1$ . Then  $L_2$  is generated from  $L_1$  and so on, until no more frequent  $k$ -itemsets can be found and then algorithm desists. Every  $L_k$  generated should scan database once. Then  $C_k$  is generated from  $L_k - 1$ .

*Pseudo code of APRIORI:*

```

Ck: candidate itemset of size k.
Lk: frequent itemset of size k.
L1: frequent items.
For (k = 1; Lk ≠ ∅; k++) do begin
Ck+1 = candidate generated from Lk;
For each transaction t in database D do increment the
calculation of all candidates in Ck+1 that are included in
t.
Lk+1 = candidate in Ck+1 with minsup.
End.
Return U(Lk).

```

## 2.2 State of the Arts

The process of association rules mining (ARM) can be categorized into two classes of research: determination of user specified support and confidence thresholds and the post-treatment by using the interestingness measures to evaluate and find the most interesting rules. Most of the algorithms of ARM rely on support and confidence thresholds and they use a uniform threshold at all levels. Therefore a suitable choice of those thresholds directly influences the number and the quality of association rules discovered.

Several works aim to solve this challenge and help the user in the choice of the threshold of support and confidence to be the most adequate to the decision scope.

Fournier-Viger [7, 8] Redefine the problem of association rule mining as mining the top- $k$  association rules by introducing an algorithm to find the top  $k$  rules having the greatest support. With  $k$  is the number of rules to be generated and defined by the users. To generate rules, Top-K-Rules relies on a novel approach called rule expansions, it

finds larger rules by recursively scanning the database for adding a single item at a time to the left or right part of each rule. This has an excellent scalability: execution time linearly increases with  $k$ . Top- $k$  pattern mining algorithm is slower but provides the benefit of permitting the user set the number of patterns to be discovered, which may be more intuitive.

Kuo et al. [9] introduced a new method to determine best values of support and confidence parameters automatically particularly for finding association rule mining using Binary Particle Swarm Optimization and offered a novel approach for mining association rule in order to develop computational performance as well as to automatically define suitable threshold values. The particle swarm optimization algorithm searches firstly for the best fitness value of each particle and then detects corresponding support and confidence as minimal threshold values after the data are converted into binary values and then these minimal support and confidence values are used to extract association rules.

Other approaches [10–12] used a multiple level in the process of ARM, the multiple level association rule mining can run with two kinds of support: Uniform and Reduced. Uniform Support: In this method, same minimum support threshold is used at every level of the hierarchy. There is no necessity to evaluate itemsets including items whose ancestors do not have minimum support. The minimum support threshold has to be suitable. If minimum support threshold is too great then we can lose lower level associations and if it is too low then we can end up in producing too many uninteresting high-level association rules.

Other works [13, 14] proposed an approach based on multi-criteria optimization aiming to select the most interesting association rules Without need to set any parameters at all, The idea is to find the patterns that are not dominated by any other patterns by using a set of interesting measures.

### 3 Proposed Approach

The main problem of The APRIORI algorithm is the choice of the threshold for the support and confidence. APRIORI find the frequent candidate itemset by generating all possible candidate itemset which verifies a minimum threshold defined by users. This choice influences in the number and the qualities of AR. whereas our algorithm uses a threshold of  $\text{minsup}$  defined depending on the transactional dataset which is logical.

In this paper we propose two main contributions, the first one is to compute the minimum support ( $\text{minsup}$ ) automatically according to each datasets instead of using a constant value predefined by the users. The second contribution of our proposed method is making this  $\text{minsup}$  change (updated) dynamically according to each level, most of the existing methods applied a single and uniform minimum support threshold value for all the items or itemsets. But all the items in an itemset do not work in the same process, some appear very frequently and oftentimes, and some infrequent and very rare. Therefore the threshold of  $\text{minsup}$  should change according to different levels of itemset.

Our algorithm can be divided in several steps:

- Input: a set of  $n$  transaction, a transactional dataset.
- Step 1: determine the minimum support for the first level for 1-itemset:  $\text{minsup}_1$  by using the means of support of all itemset with one item.

$$\text{min sup } 1 = \sum_{i=1}^N \frac{\text{sup} - 1\text{itemset}_i}{N}$$

$\text{Minsup}$ : is a minimum support and 1-itemset is a set of items composed of 1item.

- Step 2: Verify whether the support  $\text{sup} - 1\text{itemset}_i$  of each item $_i$  is large than or equal to  $\text{minsup}_1$ . If  $i$  satisfies the above condition put in the set of 1-itemset ( $L_1$ ).  
 $L_1 = \{1\text{-itemset}_i / \text{sup} - 1\text{-itemset}_i \geq \text{minsup}_1 \text{ with } i = 1 \dots N \text{ number of all } 1\text{-itemset}\}$
- Step 3: Generate the candidate  $C_2$  from  $L_1$  with the same way to the APRIORI algorithm. the difference is that the support of all the large  $k$ -itemset.  
 $k\text{-itemset}$ : set of items composed of  $k$  items.
- Step 4: Compute the new  $\text{minsup}$  of the 2 itemset level by using the means of support of the generated  $C_2$  generate the  $L_2$ .  
 $L_2 = \{2\text{-itemset}_i / \text{sup} - 2\text{-itemset}_i \geq \text{minsup}_2 \text{ with } i = 1 \dots N \text{ number of all } 2\text{-itemset}\}$
- Step 5: Check whether the support  $\text{sup} k\text{-itemset}_i$  of each candidat  $k\text{-itemset}_i$  is larger than or equal to  $\text{minsup}_k$  obtained in step 4. If it satisfies the above condition put in the set of large  $k\text{-itemset}(L_k)$ .  
 $L_k = \{k\text{-itemset}_i / \text{sup} - k\text{-itemset}_i \geq \text{minsup}_k \text{ with } i = 1 \dots N \text{ number of all } k\text{-itemset}\}$ .
- Step 6: Repeat steps 3 to 5 until  $L_i$  is null.
- Step 7: Construct the association rules for each large  $k\text{-itemset}_i$  with items:  $\{Ik_1, Ik_2, \dots, Ik_q\}$   $q \geq 2$  which verify the threshold of confidence i.e. the association rule whose confidence values are larger than or equal to the threshold of confidence defined by the mean of support of all large  $q$  itemset  $Ik$ .
- Output: a set of association rules using an automatic threshold of support in multilevel.

## 4 Experiment Study

In this part, we will illustrate and investigate the advantages of our proposed algorithm (Supd), we use different public datasets: (mushroom, flare1, flare2, Zoo, Connect) got from UCI machine learning repository [15]. T10I4D100K (T10I4D) was generated using the generator from the IBM Almaden Quest research group [16] and Foodmart is a dataset of customer transactions from a retail store, obtained and transformed from SQL-Server 2000 [17]. Table 1 summarizes the properties of the used datasets.

Our objectives in this section are multiple, the first, we show through many experiments that our method reduce the huge number of the generated association rules compared to APRIORI algorithm [2]. Second, we conduct an experiment to examine the qualities of the generated association rules. The third, we study the runtime and we compare it to the APRIORI algorithm (APR) and to Topkrule algorithm (Topk) [7].

All the approaches are implemented in Java programming language. At the first, all our experiments are realized through a computer (C1) with the following specifications: Core™ I3, 1,70 GHz, memory capacity: 4 GB.

**Table 1.** Characteristics of the used datasets

Data set	Items	Transactions
Mushroom	22	8124
Flare1	32	323
Flare2	32	1066
Zoo	28	101
Connect	42	67557
Chess	36	3196
Foodmart	1,559	4141
T10I4D	1000	100000

Table 2 shows different values of threshold of confidence chosen and different values of support found by our algorithm in the first level for different dataset.

**Table 2.** Threshold values of support and confidence

Data set	Minsup	Minconf
Mushroom	20	70
Flare1	32	50
Flare2	32	50
Zoo	30	50
Connect	33	80
Chess	76	70
Foodmart	0.02	70
T10I4D	0.2	70

#### 4.1 Reduction of a Number of Rules

In this experiment, we show the ability of our proposed approach to reduce the number of AR generated from the chosen datasets. Our experiment compares our approach to APRIORI based on thresholds.

Table 3 compares the size of AR generated by our method to the APRIORI.

**Table 3.** Number of AR generated for each dataset

	Mushroom	Flare1	Flare2	Zoo	Connect	Chess	Foodmart	T10I4D
Supd	8353	106	127	362	422874	55700	2037	111482
APR	14965927	1430	1298	3716	220224860	1776698	21982	206772

We see through this experiments that our approach can significantly reduce the huge number of rules generated from the data sets, which can facilitate the interpretation and help the users to see the most interesting ones and to take decision.

## 4.2 The Running Time Analysis

We realized an implementation for traditional Apriori from [1] and our proposed algorithm (Supd), and we compare the time wasted of original Apriori (APR) and Topkrule algorithm (Topk), and our proposed algorithm by applying many datasets, various values for the minimum support given in the implementation. The running time analysis may be differ for different machine configuration. For this reason we are realized our experiments through another machine computer (C2) with the following specifications: Core™2 Duo CPU E8400, 3,00 GHz, memory capacity: 4 GB in order to obtain unbiased result comparison. The result is shown in Tables 4 and 5.

As we see in Table 4, that the time-consuming in our proposed algorithm in each dataset is less than it is in the original Apriori, and the difference grows more and more as the number of transactions of datasets increases.

On the other hand we see that this time consuming in our approach is the same as the time consuming in Topkrule in some datasets and it is less than it in other datasets.

We can add another advantage to our algorithm which is the use of memory space. As we see, we did not obtain the result of Topkrule in Connect and T10I4D100K datasets, the Topkrule algorithm can't run on both machines and this is due to the memory problem, while our algorithm runs without any problem.

**Table 4.** The time consuming in different datasets using computer C1 (in ms)

	Muchroom	Flare1	Flare2	Zoo	Connect	Chess	Foodmart	T10I4D
Supd	3746	10	73	24	224335	1852	20660	1283038
APR	86984	100	81	60	888383	9432	62152	340642
Topk	11419	10	20	12	–	41501	18600	–

**Table 5.** The time consuming in different datasets using computer C2 (in ms)

	Muchroom	Flare1	Flare2	Zoo	Connect	Chess	Foodmart	T10I4D
Supd	5728	10	83	24	336146	2037	24309	166599
APR	96935	100	81	60	1557284	10500	84201	689908
Topk	19759	20	43	15	–	52978	4968	–

### 4.3 The Quality of the Extracted Rules

In order to analyze the performance of our proposed algorithm, we have compared the average value of confidence in each dataset of our method to the original method.

The Table 6 shows that the proposed method has found rules with high values of confidence in the majority of the datasets which ensures the benefit of our proposed method.

**Table 6.** The average of confidence for different datasets

	Muchroom	Flare1	Flare2	Zoo	Connect	Chess	Foodmart	T10I4D
Supd	0.89	0.85	0.9	0.86	0.96	0.94	1	0.87
APR	0.80	0.75	0.85	0.84	0.84	0.86	0.98	0.78
Topk	0.89	0.85	0.9	0.88	–	0.94	0.99	–

## 5 Conclusion

In this paper, we proposed a new approach based on APRIORI algorithm for discovering the association rules to auto-adjust the choice of the threshold of support. The main advantage of the proposed method is the automatism of the choice of support in multi-level, we get results containing desired rules with maximum interestingness in a little time. The numbers of rules generated by proposed algorithm are significantly less as compared to APRIORI Algorithm. Hence, we can say our algorithm answer the problem of the choice of the threshold of the support efficiently and effectively. As future works, we plan to ameliorate our approach to be able to select the interesting association rules without using any predefined threshold.

## References

1. Zhou, S., Zhang, S., Karypis, G. (eds.): Advanced Data Mining and Applications: 8th International Conference, ADMA 2012, 15–18 December 2012, Proceedings, vol. 7713. Springer Science & Business Media, Nanjing (2012)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of VLDB 1994 Proceedings of 20th International Conference on Very Large Data Bases, vol. 1215, pp. 487–499 (1994)
3. Zaki, M.J., Hsiao, C.J.: CHARM: an efficient algorithm for closed itemset mining. In: SDM 2002, Arlington, VA, pp. 457–473 (2002)
4. Pei, J., Han, J., Mao, R.: CLOSET: an efficient algorithm for mining frequent closed itemsets. In: Proceeding of the 2000 ACM-SIGMOD International Workshop Data Mining and Knowledge Discovery (DMKD 2000), Dallas, TX, pp. 11–20. ACM (2000)
5. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.-C.: Mining sequential patterns by pattern-growth: the prefixspan approach. *IEEE Trans. Knowl. Data Eng.* **16**, 1424–1440 (2004)



6. Schmidt-Thieme, L.: Algorithmic features of Eclat. In: Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations (FIMI 2004), CEUR Workshop Proceedings, Brighton, UK, vol. 126 (2004)
7. Fournier-Viger, P., Wu, C.-W., Tseng, V. S.: Mining top-k association rules. In: Proceedings of the 25th Canadian Conference on Artificial Intelligence (AI 2012), pp. 61–73. Springer, Canada (2012)
8. Fournier-Viger, P., Tseng, V.S.: Mining top-k non-redundant association rules. In: Proceedings of 20th International Symposium, ISMIS 2012, LNCS, Macau, China, vol. 7661, pp. 31–40. Springer (2012)
9. Kuo, R.J., Chao, C.M., Chiu, Y.T.: Application of particle swarm optimization to association rule mining. In: Proceeding of Applied Soft Computing, pp. 326–336. Elsevier (2011)
10. Liu, B., Hsu, W., Ma, Y.: Mining association rules with multiple minimum supports. In: Knowledge Discovery and Databases, pp. 337–341 (1999)
11. Lee, Y.C., Hong, T.P., Lin, W.Y.: Mining association rules with multiple minimum supports using maximum constraints. *Int. J. Approx. Reason.* **40**(1), 44–54 (2005)
12. Hu, Y.-H., Chen, Y.-L.: Mining association rules with multiple minimum supports: a new algorithm and a support tuning mechanism. *Decis. Support Syst.* **42**(1), 1–24 (2006)
13. Bouker, S., Saidi, R., Ben Yahia, S., Mephu Nguifo, E.: Mining undominated association rules through interestingness measures. *Int. J. Artif. Intell. Tools* **23**(04), 1460011 (2014)
14. Dahbi, A., Jabri, S., Ballouki, Y., Gadi, T.: A new method to select the interesting association rules with multiple criteria. *Int. J. Intell. Eng. Syst.* **10**(5), 191–200 (2017)
15. UCI machine learning repository. <https://archive.ics.uci.edu/ml/index.php>. Accessed 10 Jan 2018
16. Frequent Itemset Mining Implementations Repository. <http://fimi.ua.ac.be/data/>. Accessed 10 Jan 2018
17. An Open-Source Data Mining Library. <http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php>. Accessed 10 Jan 2018