ISSN 2278-3091
Volume 9, No.1.1, 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse1091.12020.pdf
https://doi.org/10.30534/ijatcse/2020/1091.12020

A Modified Apriori Algorithm to Mine Association Rules using Relative Multiple Supports

Miriam P. Pariñas[1], Ariel M. Sison[2], Ruji P. Medina[3]
[1]Technological Institute of the Philippines-Quezon City, Philippines, mpparinas@gmail.com
[2]Emilio Aguinaldo College Manila, Philippines, ariel.sison@eac.edu.ph
[3]Technological Institute of the Philippines-Quezon City, Philippines, ruji.medina@tip.edu.ph

ABSTRACT

Mining frequent itemsets utilizing multiple minimum supports is an essential generalization of the association rule mining problem. Instead of setting a single minimum support for all items, users are allowed to specify different minimum support values to different items. In real applications, using single minimum item support is inadequate since it does not reflect the nature of each item. If single minimum support is set to low, a large number of association rules are generated. On the other hand, if it is set to high, important rules may be lost. In this paper, we proposed an algorithm named Relative Multiple Supports Apriori (RMSApriori) to solve the problem of single multiple support. It is compared with the original Apriori and various experiments are conducted using real datasets. Different values of single minimum support were applied on each dataset for comparison and rules involving frequent and rare items were found. However, the minimum support value has to be set to low resulting in an increase of processing time and space. Experimental results reveal that RMSApriori outperforms Apriori in terms of execution time and memory usage considering the generation of rules not only for frequent items but also for significant rare items.

Key words: Candidate itemsets, Frequent itemsets, Multiple minimum supports, Single minimum support.

1. INTRODUCTION

Mining association rule is considered an essential research method in the field of data mining utilized to obtain useful knowledge and describe the association between different valuable data [1]. Association rules which assist in marketing, advertising, inventory control, and fault prediction in telecommunication network are based on the discovered frequent set of items [2]. These are expressed in consequent. Extracting frequent itemsets is one of the main steps in discovering association rules.

Apriori is an algorithm implemented for mining frequent itemsets and for generating association rules. It is a classic algorithm proposed by R. Agrawal and R. Srikant and known for discovering rules in data mining [3]. Traditionally, this algorithm is known for its usefulness in market basket analysis but it can also be applied into medical data, mobile e-commerce, web usage mining and academic data [4],[5],[6]. It utilizes single minimum support as a measure to identify a set of frequent itemset [7]. It reduces the search space and limits the number of frequent patterns generated. However, using single minimum support makes an assumption that all items in the dataset are of the same nature and have the same frequencies which in contrary, not the case in real life applications [8],[9],[10],[11]. In reality, datasets contain items of varying frequently and knowledge pertaining to frequent items can be discovered in the same manner as that pertaining to rare items [12]. Thus, the main problem in association rule mining is setting the minimum support. It is hard to decide what value of minimum support should be used for many datasets. A low value of minimum support can generate rules involving frequent and rare items but can suffer from generating too many rules including uninteresting rules. On the other hand, a high value of minimum support can control the generation of rules but can suffer from generating rules involving rare items. These problems found in setting single minimum support are known as rare item and rules explosion problems.

To overcome the drawbacks of single minimum support, various modifications have been made [13],[14],[15],[16] where in multiple minimum supports approach was employed to reflect the different natures and frequencies of

items and so that every item in the dataset should be given importance. These algorithms are based on modifying the classical Apriori algorithm. Liu et al. proposed MSApriori to discover frequent patterns with Multiple Supports framework. In this framework, every item in the dataset has its own minimum support value set by the user and itemset can satisfy a different minimum support depending on the items within an itemset. Elahe and Zhang proposed C-MSApriori which aims to mine rules both for occurring frequently and rarely. It uses three parameters namely frequent minimum support, support difference and rare minimum support for mining frequent itemsets. Yun and Hwang proposed RSAA to mine itemsets for frequent and rare items that uses three supports such as first support, second support and relative support.

Since discovering frequent patterns is the primary step in Association Rule Mining (ARM), it can also be regarded as the crucial stage prior to the generation of rules. Almost all researches on ARM demonstrate different techniques on how to improve the process of finding frequent patterns [17]. In this paper, an algorithm named Relative Multiple Supports Apriori (RMSApriori) is proposed. It is an improvement on the traditional Apriori algorithm and designed to discover frequent itemsets under a multiple minimum supports approach. Assigning minimum support for each item enables the proposed algorithm to discovers itemsets involving frequent and rare items. Experimental results show that RMSApriori outperforms Apriori in terms of time and memory.

The rest of the paper is organized as follows. In Section 2, we introduce some terminologies and explain the proposed approach in finding frequent itemset followed by performance analysis through experimental tests in Section 3. Finally, a conclusion is derived in Section 4.

2. METHODS

2.1 Preliminaries

Let $I=\{i_1,i_2,i_3...,i_k\}$ be the set of all items in a transaction *T*. Each transaction in *T* contains set of items *X* in *I*. The support (represented in percentage) of an itemset *X*, denoted as *supp(X)*, is the number of transactions containing *X* in *T*. An itemset *X* is considered frequent if its support count is not less than the user-defined minimum support (*minsupp*). An itemset containing *k* number of items is *k-itemset*.

Let $MIS_{(i)}$ be the minimum support value of an item *i*. Assume itemset $X=\{i_1,i_2,i_3,\ldots,i_k\}$, the minimum support of an itemset X denoted as *minsupp(X),* is the minimum support value among items in an itemset. Thus, itemset X is considered frequent if its support count is >=min $[MIS(i_1),MIS(i_2),MIS(i_3)\ldots MIS(i_k)]$.

2.2 RMSApriori Algorithm

The proposed algorithm is called Relative Multiple Supports Apriori (RMSApriori). It adopts the Apriori algorithm which is based on level-wise search and follows downward closure property wherein all subsets of a frequent itemsets are also frequent. It is similar to Apriori in extracting the set of frequent patterns except for the following dfferences.

1. Apriori uses a single minimum support threshold whereas RMSApriori uses multiple minimum supports to give consideration not only to items with high frequency but also to items wth low frequency. Utilizing different values of minimum support will prevent the immediate pruning of items which can also generate important or interesting rules.
2. In finding frequent patterns in Apriori, items or itemsets need to satisfy only one minimum support value whereas in RMSApriori, items or itemsets need to satisfy the lowest minimum support among items in an itemset.

RMSApriori utilizes two parameters namely $supp_1$ and $supp_2$ in order to compute the minimum support of each item denoted as MIS(i). $supp_1$ is the ratio of item support count $suppcount_{(i)}$ to the total transaction *tn* and $supp_2$ is the ratio of item support count $suppcount_{(i)}$ to total frequency *tf*. Thus, MIS(i) can be obtained by getting the difference between $supp_1$ and $supp_2$.

2.3 Procedure of RMSApriori Algorithm

1. Generate candidate 1-itemset C_1 and its support count;
2. Calculate the minimum support of each item $MIS_{(i)}$ using the following formulas:

$$supp_{1(i)} = suppcount_{(i)} / tn$$
$$supp_{2(i)=} suppcount_{(i)} / tf$$

3. Compare the support count of each candidate 1-itemset in C_1 with its $MIS_{(i)}$ value. Items are considered frequent L_1 if its support count is >= $MIS_{(i)}$.
4. Generate candidate 2-itemset C_2 by combining items in L_1.
5. Determine the support count of each 2-itemset in C_2.
6. Compare the support count of each candidate 2-itemset in C_2 with the lowest minimum support value among items in an itemset. Itemsets are considered frequent L_2, if its support count is >=min$[MIS(i_1),MIS(i_2),MIS(i_3)\ldots MIS(i_k)]$.

Item	A	B	C	D	E	F	G
MIS(i)	0.44	0.59	0.29	0.37	0.66	0.15	0.29

7. Repeat 4 to 6 to generate candidate k-itemsets C_k and frequent k-itemsets L_k until no possible large k-itemsets can be generated.
8. Construct the association rules for each frequent itemset.
9. Output the rules.

3.4 Sample simulation of the proposed RMSApriori Algorithm

An example below will demonstrates the proposed RMSApriori algorithm finds the frequent itemsets. Given are 10 transactions with 7 items as shown in Table 1.

Table 1: Transaction Data

TId	Items
1	ABDG
2	BDE
3	ABCEF
4	BDEG
5	ABCEG
6	BEG
7	ACDE
8	BEG
9	ABEF
10	ACDE

Scan the transaction data to find candidate 1-itemset C_1 and its support count. As shown in Table 2, notice that there are items appearing frequently and infrequently in the transaction.

Table 2: Candidate 1-Itemset C_1

C_1	A	B	C	D	E	F	G
Support Count	6	8	4	5	9	2	4

The next step is to compute the minimum support of each item denoted by MIS(i). This can be done by finding first the values of $supp_{1(i)}$ and $supp_{2(i)}$. using the following formula:

$$supp_{1(i)}=suppcount_{(i)}/tn \quad (1)$$

$$supp_{2\,(i)}=suppcount_{(i)}/tf \quad (2)$$

where i is an item in the dataset, $suppcount_{(i)}$ is the frequency of an item i, tn is the total transaction, and tf is the total frequency. In this case, for item A, the value of $supp_1$ is 6/10=0.6, and $supp_2$=6/38=0.16 respectively. The minimum item support MIS(i) can be obtained by:

$$MIS_{(i)} = supp_{1(i)} - supp_{2(i)} \quad (3)$$

Thus, the $MIS_{(A)}$ = 0.6 - 0.16 = 0.44. All items and its equivalent MIS values are shown in Table 3.

Table 3: Minimum Item Support MIS(i)

The algorithm now finds frequent 1-itemset L_1 by comparing the support count of each candidate 1-itemset C_1 with its equivalent MIS value (e.g., $MIS_{(A)}$ * tn = 0.44 * 10 = 4.4). All items in C_1 are found to be frequent 1-itemset L_1 as shown in Table 4.

Table 4: Frequent 1-Itemset L_1

L_1	A	B	C	D	E	F	G
Support Count	6	8	4	5	9	2	4

Using the join step, candidate 2-itemset C_2 are formed and the next scan is performed to determine the support count of each candidate itemset in C_2, as reflected in Table 5.

Table 5: Candidate 2-Itemset C_2

C_2	Support Count
AB	4
AC	4
AD	3
AE	5
AF	2
AG	2
BC	2
BD	3
BE	7
BF	2
BG	5
CD	2
CE	4
CF	1
CG	1
DE	4
DG	2
EF	2
EG	4

Frequent 2-itemsets L_2, are found by comparing the support count of candidate itemset in C_2 with the lowest minimum support value among items in an itemset. In the case of itemset AB, the equivqlent MIS value of item A and item B are 4.4 and 5.9 respectively and the support count of itemset AB is 4. Thus, itemset AB is not considered frequent since its support count is less than both the MIS value of item A and item B. The complete set of frequent 2-itemset L_2 is shown in Table 6.

Table 6: Frequent 2-Itemsets L_2

L_2	Support Count
AC	4
AE	5
AF	2
BE	7
BF	2
BG	5
CE	4
DE	4
EF	2
EG	4

From the generated L_2, Candidate 3-itemset C_3 are found by combining frequent itemsets in L_2. Then the next scan is performed to determine the support count of each candidate itemset in C_3 as shown in Table 7.

Table 7: Candidate 3-Itemset C_3

C_3	ACE	ACF	AEF	BEF	BEG
Support Count	4	1	2	2	3

As shown in Table 8, frequent 3-itemset L_3 are found by comparing the support count of itemset in C_3 with the lowest minimum support value among items in an itemset.

Table 8: Frequent 3-Itemset L_3

L_3	ACE	AEF	BEF	BEG
Support Count	4	2	2	3

From the generated L_3, no frequent 4-itemset L_4 can be generated. Thus, the complete set of frequent itemsets are {ACE},{AEF},{BEF}, and {BEG}. From the discovered frequent patterns using the proposed algorithm, 24 rules involving frequent and rare items were discovered.

3. RESULTS

The performance of the traditional Apriori and the proposed RMSApriori was evaluated by comparing the run-time and memory space used by both algorithms based on the generated rules. Datasets used in time and space experiments are Census Income with 30,000 instances and 42 items and Demographic Profile of Students with 2,500 instances and 14 items.

In the experiments conducted, each dataset utilized different values of single minimum support for Apriori algorithm whereas the proposed RMSApriori algorithm used the actual frequency of item, the total number of transactions and total frequency of all items as the basis for multiple minimum supports assignments.

To test the performance of both algorithms, we measured the execution time and space required for the discovery of association rules. The unit of time is expressed in seconds, and of memory usage in megabytes. In order to compare the time and space utilized by both algorithms in finding rules for both frequent and rare items, for Apriori, we applied different values of single minimum support. Initial values generate lesser rules, however, rules involving rare items were missing. Then, we gradually reduced the minimum support value until rules for frequent along with rare items are discovered but resulted in a greater number of rules. Thus, a decrease in the value of minimum support means an increase in processing time and space because setting a low minimum support leads to the generation of more candidate itemsets. For RMSApriori, we assigned minimum support for every item instead of using one minimum support. This approach prevents the immediate pruning of items with low support count enabling the generation of rules involving rare items.

The results from comparing the run-time and space of Apriori and RMSApriori algorithms are shown in Figure 1 and Figure 2.

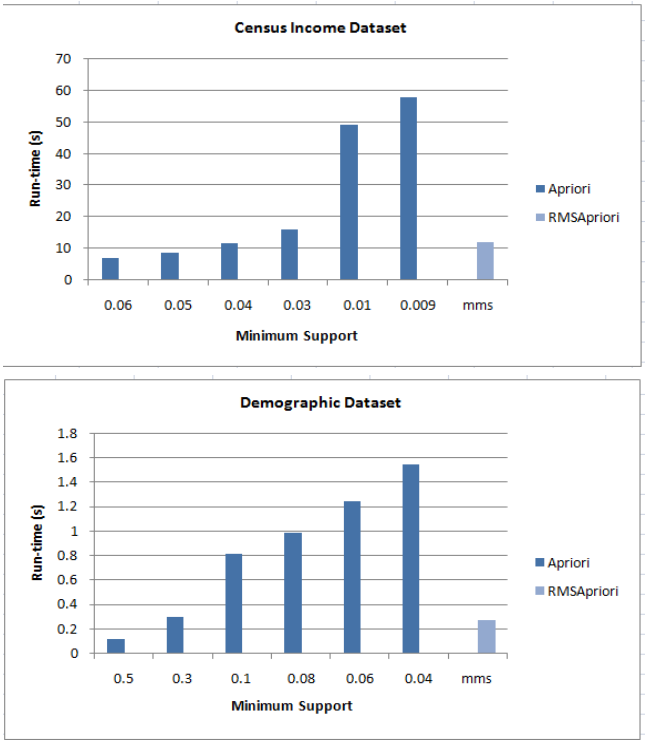

Figure 1: Run-time

Figure 1 shows the processing time required by both algorithms. It is observed that when we applied 0.06, 0.05, and 0.04 single minimum support values to Census data, it required lesser time than that of the proposed approach. However, rules involving rare items were not extracted. Similarly, in the case of demographic data, when a single minimum support value was set to 0.5, the processing time obtained by Apriori is shorter than that obtained by RMSApriori but again, fewer rules were discovered and rules for rare items were not found. Decreasing the values of single minimum support gradually increases the processing time. For an Apriori to discover the target rules, we lowered the value of single minimum support but in effect, it takes more time to process the data. Conversely, the processing time acquired by RMSApriori using multiple minimum supports is much shorter than with Apriori when the discovery of rules for both frequent and rare items is taken into consideration. This is because the proposed approach reduced the number of candidate and frequent itemsets.

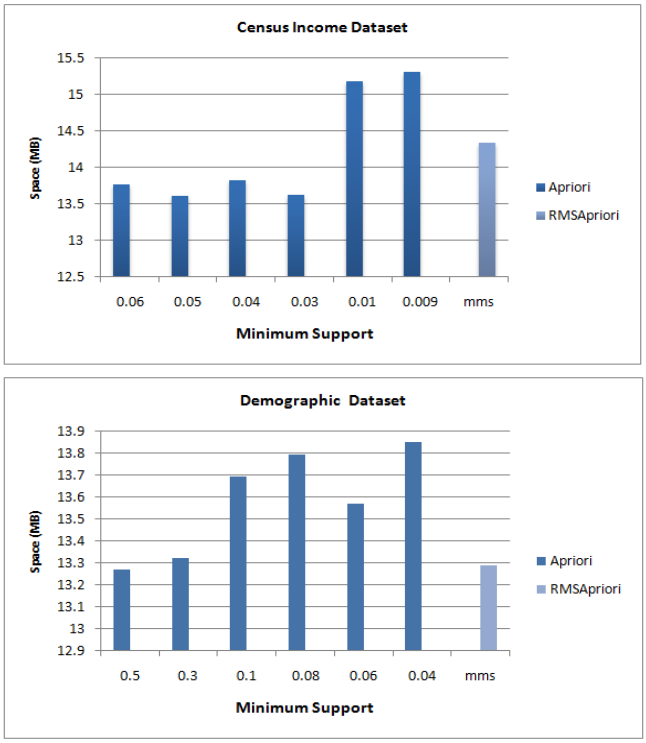

Figure 2: Memory Usage

Figure 2 shows the memory usage of both algorithms. RMSApriori performs better than Apriori in finding rules for frequent and rare items. For Apriori to discover items with low support count, it has to lower the minimum support value which in effect, generates more candidate and frequent itemsets. Though, in every initial scan using RMSApriori, all candidate items C_1 are considered frequent items L_1, but at the end of the process of finding frequent itemsets, it controls the generation of too many candidates itemsets.

4. CONCLUSION

We have proposed RMSApriori through which we can discover association rules that consider not only frequent but also significant rare items. It is based on the Apriori algorithm since Apriori was the first proposed algorithm and has been widely used and studied.

Experimental results show that multiple minimum supports outperforms single minimum support because for Apriori to discover rules for frequent and rare items, the minimum support value must be lowered resulting in an increase in the generation of candidate itemsets as well as processing time and space. RMSApriori discovers rules for both frequent and rare items while reducing the number of candidate itemsets resulting to lesser exeution time and space.

As part of future work, exhaustive experiments should be done to continuously improve the performance of the proposed approach. We also believed that further studies are required to test its suitability to various types of dataset.

REFERENCES

[1] G. Wang, X. Yu, D. Peng, Y. Cui, and Q. Li. **Research of Data Mining Based on Apriori algorithm in Cutting Database**, no. 50805100, pp. 0–3, 2010.

[2] J. Singh and H. Ram. **Improving efficiency of Apriori algorithm using Transaction reduction**, *Int. J. Sci. Res. Publ.*, vol. 3, no. 1, pp. 1–4, 2013.

[3] Z. Chen, S. Cai, Q. Song, and C. Zhu. **Retracted Article: An improved Apriori algorithm based on pruning optimization and transaction reduction**, *2011 2nd Int. Conf. Artif. Intell. Manag. Sci. Electron. Commer. AIMSEC 2011 - Proc.*, pp. 1908–1911, 2011.

[4] Q. Zhang. **The Application of Apriori Algorithm in Analysis on Admitted Students of Colleges and Universities**, *Appl. Mech. Mater.*, vol. 321–324, pp. 2578–2582, 2013.

[5] Y. Guo, M. Wang, and X. Li. **Application of an Improved Apriori algorithm in a Mobile e-commerce Recommendation System**, *Ind. Manag. Data Syst.*, vol. 117, no. 2, pp. 287–303, 2017. https://doi.org/10.1108/IMDS-03-2016-0094

[6] N. Duru. **An Application of Apriori Algorithm on a Diabetic Database**, *Database*, vol. 3681 LNAI, pp. 398–404, 2005.

[7] W. Gan, J. C. W. Lin, P. Fournier-Viger, H. C. Chao, and J. Zhan. **Mining of frequent patterns with multiple minimum supports**, *Eng. Appl. Artif. Intell.*, vol. 60, no. January, pp. 83–96, 2017.

[8] H. Zhang, J. Zhang, X. Wei, X. Zhang, and T. Zou. **A New Frequent Pattern Mining Algorithm with Weighted Multiple Minimum Supports**, *Intell. Autom. Soft Comput.*, vol. 23, no. 4, pp. 605–617, 2017.

[9] T. Xu and X. Dong. **Mining frequent patterns with multiple minimum supports using basic Apriori**, *Proc. - Int. Conf. Nat. Comput.*, pp. 957–961, 2013.

[10] F. Elahe and K. Zhang. **Mining Frequent Itemsets Along with Rare Itemsets Based on Categorical Multiple Minimum Support**, vol. 18, no. 6, pp. 109–114, 2016.

[11] Y. C. Lee, T. P. Hong, and W. Y. Lin. **Mining association rules with multiple minimum supports using maximum constraints**, *Int. J. Approx. Reason.*, vol. 40, no. 1–2, pp. 44–54, 2005. https://doi.org/10.1016/j.ijar.2004.11.006

[12] R. U. Kiran and P. K. Reddy. **An improved multiple minimum support based approach to mine rare association rules**, *2009 IEEE Symp. Comput. Intell. Data Mining, CIDM 2009 - Proc.*, no. January, pp. 340–347, 2009.

[13] B. Liu, W. Hsu, and Y. Ma. **Mining Association Rules with Multiple Minimum Supports**, 1999.

[14] H. Yun and B. Hwang. **Mining association rules on significant rare data using relative support**, vol. 67, pp. 181–191, 2003.

[15] Y. Lee, T. Hong, and W. Lin. **Mining association rules with multiple minimum supports using maximum constraints**, vol. 40, pp. 44–54, 2005.

[16] M. Tseng and W. Lin. **Efficient mining of generalized association rules with non-uniform minimum support**, vol. 62, pp. 41–64, 2007.

[17] S. Darrab and B. Ergenç. **Frequent Pattern Mining under Multiple Support Thresholds**, vol. 4, 2016. https://doi.org/10.1016/j.procs.2017.08.051