Enhancing Association Rule Mining with a Profit Based Approach

Ming-Lai Li[1], Heung-Num Kim[1], Jason J.Jung[1] and Geun-Sik Jo[2]
Intelligent E-commerce Systems Laboratory, School of Computer Engineering,
[1]Inha University, 253 Yonghyun-Dong, Nam-Gu, Incheon Korean 402-751
Liminglai2002@hotmail.com, nami@eslab.inha.ac.kr, j2jung@intelligent.pe.kr
[2]School of Computer Engineering, Inha University,
253 Yonghyun-Dong, Nam-Gu, Incheon, Korean 402-751
gsjo@inha.ac.kr

Abstract

With the continuous growth of e-commerce there is a huge amount of products information available online. Shop managers expect to apply information techniques to increase profit and perfect service. Hence many e-commerce systems use association rule mining to further refine their management. However previous association rule algorithms have two limitations. Firstly, they only use the number to weight item's essentiality and ignore essentiality of item profit. Secondly, they did not consider the relationship between number and profit of item when they do mining. We address a novel algorithm, profit-based association rule algorithm that uses profit-based technique to generate 1-itemsets and the multiple minimum supports mining technique to generate N-items large itemsets.

1. Introduction

Nowadays many online shop managers are willing to apply association rule mining to their business to increase profits and perfect customer service. However, unfortunately traditional association rule algorithms are amount-based approach which is difficult to mine the items with high profits but low amount. A plenty of different algorithms for association rule mining have been proposed [4,5,7]. However, it is generally true that the association rules in themselves do not serve the ultimate purpose of profit but mine the rules based on amount of items. [6,9]. In this paper, our algorithm is based on profit for the best benefits and giving a better recommendation to buyers. Here we present the profit-based association rule, which uses a unique profit to generate a minimum support for every 1-itemset and uses multiple minimum supports to mine association rules.

2. Previous Work on Association Rules

Association rule analyzes how the items purchased by customers are associated. Association rule mining has been studied extensively in the past [2,3,4,5].
Previous algorithms only use single **minimum support** (minsup) that implicitly assumes all items in the data are of the same nature. Every item has the same profit and similar frequencies in the database. These are not the case in real life applications. Firstly, from the profit point of view though some items have a few amounts which are usually less than the minsup, they take more weight than those which have large amounts. If using previous association rules, like the Apriori algorithm [2], we will encounter two problems: one is we always mine out some rules made few profits. Second is in the first iteration of the Apriori algorithm to generate 1-itemset some items are deleted, which can make higher profits but have lower support. Therefore we should consider profit when we do association rule not only concern the amount of items. Former researchers[10] have found some methods to solve it but they are not efficient. In this paper our algorithm will based on one of existent approach "mining association rules with multiple minimum supports" to mining association rules with N-itemsets (N≥2)[4].

3. Profit-based Association Rule Algorithm (PAR)

This algorithm derived from the Apriori algorithm. The differentia between this paper's and previous approaches is that we use the profit as a criterion to set minimum support for each item [7]. In our algorithm we first set minimum support for each item using unique profit criterion. Then we mine large itemsets with multiple minimum supports.

3.1 Generate 1-itemset Using Minimum Total Profit

We use minimum total profit to generate 1-itemset. In this step we only compare the total profit of each item with the minimum total profit. We delete all the 1-items which have total profits less than minimum total profit. For example, in table 1 we delete itme3 and item4.

Table 1. Example of setting minimum support for item

	Item 1	Item 2	Item 3	Item 4
Amount of Item	4	40	150	150
Unit Profit	100$	2.5$	0.5$	0.5$
Total Profit	400$	100$	75$	75$
Minimum Total Profit	100$			

3.2 Using Profit to set Minsup for Each Item

In this step we use Unique Profit to reevaluate and set Profit-Based Amount, the minimum support, for each item and generate N-itemset ($N \geq 2$).

Profit is the most clear and effective character. So profit should be the criterion. But the profits of items as the amount of items float in a large range. So we use profit as a unique criterion to reevaluate the amount of items. In other words, here the profit margin is looked on as a power of amount of items. After re-evaluating by profit, the amount of the item which has large amount but lower profit is decreased corresponding to the profits. And the amount of the item which has little amount but higher profit is increase. Another benefit getting from reevaluate the amount is that we can still use the level-wise Apriori algorithm to do association rule mining which is based on the amount of items as threshold. Before giving details we first introduce one concept:

Specific Profit: It is a unique profit value given by user and is used to calculate profit-based amount of each item. (Table 1). It is set as an integer and same as or times of the highest unit item profit. Because setting by this way most of the final profit-based amount can be a integer thereby predigest the computation.

Minimum Profit-based Support (MPS): The minimum support of every item. Though it is still the amount of item but it considers the profit affections to amount. And it satisfies the following formula:

$$\text{MPS} = \frac{\text{Specific Profit}}{\text{Unit Profit}}$$

For example, minimum profit-based support of item2 is calculated as follow, suppose specific profit is 100:

$$\text{MPS (item2)} = \frac{\text{Specific Profit}}{\text{Unit Profit}} = \frac{100}{2.5} = 40$$

It means to get 100$ profit by selling item2 we should sell 40 item2. And using the same way we calculate *minimum profit-based support of item1. The result is* equal to 1. It means to get 100$ profit by selling item1 we should sell one item1.

3.3 Mining Large Itemsets with MPS

For each item we can mine large itemsets with MPS as describe in [5]. In the first pass, it counts the supports of individual items and determines whether they are large. Each subsequent pass starts with the seed set of itemsets found to be large in the previous pass. It uses this seed set to generate new possibly large itemsets, called candidate itemsets. The actual supports for these candidate itemsets are computed during the pass over the data. At the end of the pass, it determines which of the candidate itemsets are actually large. A key operation in the proposed algorithm is the sorting of the items in the set of items (I) in ascending order of their MPS values. This ordering is used in all subsequent operations of the algorithm. The items in each item-set also follow this order. Let L_k, denote the set of large k-itemsets. Each itemset c is of the following form, <c[l], c[2], ... c[k]>, which consists of items, c[l], c[2], ... c[k], where MIS(c[l])≤l MIS(c[2])≤ . . . ≤MIS(c[k]). The algorithm is given below: **Algorithm MPSApriori**

```
M=sort(I,MS)     /*according to MPS(i) in MS*/
F=init-pass(M,T)/*make the first pass over T*/
$L_1=\{<f> \mid f \in F, f.count \geq MIS(f)\}$;
for($k=2; L_{k-1} \neq \phi$; k++) do
    if k=2 then $C_2$=level2-candidate-gen(F)
    else $C_k$=candidate-gen($L_{k-1}$)
    end
    for each candidate $t \in T$ do
        $C_t$=subs$_k$,tet(C);
        for each candidate $c \in C_t$ do c.count++;
    end
    $L_k=\{c \in C_k \mid c.count \geq MIS(c[1])\}$
end
Answer = $\cup_k L_k$;
```

For each subsequent pass, say pass k, the algorithm performs 3 operations. Both candidate generation functions level2-candidate-gen and candidate-gen are described below.

3.4 Candidate Generation

Level2-candidate-gen takes as argument F (not L_1,), and returns a superset of the set of all large 2-itemsets. The algorithm is as follows:

```
for each item f in F in the same order do
  if f.count ≥ MPS(f)then
    for each item h in F that is after f do
      if h.count ≥ MPS(f) then
          insert<f,h>into $c_2$
```

The function candidate-gen performs a similar task as apriori-gen in Apriori algorithm [3]. The prune step is, *however, different. The join step is given below. It* joins L_{k-1}, with L_{k-1}:

```
insert into $C_k$
select p.item$_1$, p.item$_2$,..., p.item$_{k-1}$,q.item$_{k-1}$
from $L_{k-1}$p,$L_{k-1}$q
where p.item$_1$=q.item$_1$,...,p.item$_{k-2}$=q.item$_{k-2}$
      p.item$_{k-1}$<q.item$_{k-1}$
```

Basically, it joins any two itemsets in L_k, whose first k-2 items are the same, but the last items are different. After the join step, there may still be candidate itemsets in C, that are impossible to be large. The prune step removes these itemsets. This step is given below:

```
for each itemset $c \in$ Ck do
for each(k-1)-subset s of c do
    if(c[1]$\in$ s)or (MIS(c[2])=MIS(c[1]))then
        if(s$\notin L_{k-1}$) then delete c form $C_k$;
```

It checks each itemset c in C_k, (line 1) to see whether it can be deleted by finding its (k-1)-subsets in L_{k-1}, For each (k-l)-subset s in c, if s is not in L_{k-1}, c can be deleted.

4 Experiments

In our experiment, we use the IBM synthetic data generator in [8] to generate the data set with the following parameters (same as [7]): 1,000 items, 10,000 transactions, 10 items per transaction on average, and 4 items per frequent itemset on average. We generated the profit of items for a single quantity as follows: 80% of items have a medium profit ranging from $1 to $5, 10% of items have a high profit ranging from $5to $10, 10% of items have a low profit ranging from $0.1 to $1. This is a simplified version of the normal distribution.

4.1 Experiment Results and Discussion

In our experiments we try to run this algorithm and another amount-based association rules mining with multiple minimum support, MSapriori [5].

MSapriori is amount-based and our approach is profit-based. However ours is generated from MSapriori. That means the ways to calculate N-itemsets (N≥2) with multiple minimum support are same. In other words, the only difference is to generate 1-itemset. The results are shown in Table 2 and Table 3 as follows(LS: lowest support a:):

Table 2. Number of large itemsets found

	MSapriori (traditional)/ PAR (novel)		
LS	a=2	a=10	a=20
0.1%	5140/5070	25030/24000	29570/28470
0.2%	4920/4750	12850/11250	13100/12020
0.3%	2400/2300	5040/4980	5910/5900

From table 2, we found ours PAR algorithm result always lower then traditional algorithm MSapriori. It is because that PAR deletes most 1-items which take less profit at the first time to pass over the database. But the same time MSapriori does not do this work.

Table 3 Number of candidate itemsets (LS: lowest support)

	MSapriori (traditional)/ PAR (novel)		
LS	a=2	a=10	a=20
0.1%	326520/345450	356840/369510	402560/422100
0.2%	269540/285640	275100/285000	290110/310050
0.3%	225460/235010	235420/250550	241050/259840

From table 3, we found though the MSapriori numbers of large itemsets are mostly bigger than our algorithm, but PAR generates more candidate itemsets. The reason is in our approach using minimum total profit value generate 1-itemset at the beginning of the process can delete most of the useless items. Secondly, we use profit-based support to generate N-itemsets.

5 Conclusions

Our algorithm enhances the efficiency of user operation and effectively helps shop managers to get more profits to increase the quantity of sale. In this study, we first introduce unique profit criterion to generate unique minsup for each item and combine this technique with the algorithm "mining association rules with multiple minimum supports". Finally, we apply this profit-based association rule mining algorithm for e-commerce environment. Our algorithm still has some problems. Such as the minimum total profit is difficult to be set value, the time-consuming is very long and how we improve the efficiency. We should modify this approach in the future.

Acknowledgement

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

References

1. Kohavi, R.: Mining e-commerce data: the good, the bad, and the ugly, ACM SIGKDD conference on Knowledge discovery and data mining, 2001, 8-13.
2. Agrawal, R., Imielinski: Mining association rules between sets of items in large databases, international conference on Management of data, 1993, 207-216.
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules, Proceedings of the 20th International Conference on Very Large Data Bases, 1994, 487-499.
4. Han, J., Fu, Y.: Discovery of multiple-level association rules from large databases, 1995 International Conference on Very Large Data Bases, 1995, 420-431.
5. Liu, B., Hsu, W., Ma, Y.: Mining association rules with multiple minimum supports, Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, 1999, 337-341.
6. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of Recommendation Algorithms for E-Commerce, Proceedings of the 2nd ACM conference on Electronic commerce, 2000, 158-167.
7. Wang, K., Su, M.-Y.T.: Item selection by "hub-authority" profit ranking, Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002, 652-657.
8. www.almaden.ibm.com/cs/quest/syndata.html
9. Brin, S., Motwani, R., Silverstein, C.: Beyond market baskets: generalizing association rules to correlations, Proceedings of the 1997 ACM SIGMOD international conference on Management of data, 1997, 265-276.
10. Mannila, H.: Database methods for data mining, Proceedings of the KDD-98, Tutorial, 1998.