An Improved Data Mining Method Applied to the Search of Relationship between Metabolic Syndrome and Lifestyles

Yi Chao Huang, Yu Ling Liao*, Chiu Shuang Lin

***Abstract*—**A data cutting and sorting method (DCSM) is proposed to optimize the performance of data mining. DCSM reduces the calculation time by getting rid of redundant data during the data mining process. In addition, DCSM minimizes the computational units by splitting the database and by sorting data with support counts. In the process of searching for the relationship between metabolic syndrome and lifestyles with the health examination database of an electronics manufacturing company, DCSM demonstrates higher search efficiency than the traditional Apriori algorithm in tests with different support counts.

***Keywords*—**Data mining, Data cutting and sorting method, Apriori algorithm, Metabolic syndrome

I. Introduction

Due to the rapid development of hospital information, medical records that used to be exclusively paper-based are now replaced by electronic files stored in databases. The databases generally grow in size over time, so much so that data analysis and interpretation can be an extremely labor-intensive and time-consuming process, so are they more susceptible to human error and negligence. This study, therefore, adopts an improved data mining method to extract valuable information automatically from huge amounts of data. The Apriori algorithm has been one of the most popular data mining techniques since it was proposed in 1994 [1]; nevertheless, its efficiency is seriously undermined by the need to repeatedly scan the entire database. This study proposes to replace Apriori with the Data Cutting and Sorting Method (DCSM), which significantly enhances the performance of the former by reducing both the need to scan irrelevant data and the number of scans. Aside from improving algorithm efficiency, this research presents a case study that investigates the correlation between the lifestyle of employees in Taiwan's electronics industry and metabolic syndrome. The health examination records of an electronics company in Taiwan provide the main basis for this empirical inquiry. A questionnaire is also used to collect the employees' basic information, workplace conditions, lifestyles, diets and eating patterns, workout habits and so on. Data mining techniques are applied to the questionnaire data, as well as to the health examination records of the same year, in order to search for relationship between lifestyles and metabolic syndrome. The research results will have immense implications for employee health management staff in assessing the specific needs and interests of the electronics industry. DCSM and Apriori are applied to the same database to compare their execution time. The result shows that both are capable of obtaining the same association rules, but DCSM has better search efficiency than the traditional Apriori algorithm.

II. Literature Review

Data mining can help unearth hidden relations, specific association rules, or previously unknown facts in large amounts of data [2], [3]. In general, data mining involves first extracting appropriate data from massive datasets, and then, after performing some data conversion and data processing tasks, reaching the ultimate goal of uncovering useful knowledge through data analysis [4]. Currently, a number of algorithms and computer technologies have been widely adopted in data mining. Cluster Detection methods, for example, are often employed to build medical database systems that organize vast amounts of medical literature into different categories. In this way, the data are transformed into useful information to assist in making medical decisions. Proposed by Agrawal and Srikant in 1994, the Apriori algorithm has been extensively applied in a wide range of data mining projects [1]. However, thanks to the rapid advances in technology today, database size may grow exponentially to the extent that it seriously impairs the efficiency of Apriori as the algorithm requires iterative database scans. Scholars have since proposed an impressive array of modified algorithms with heightened efficiency, for example, the Direct Hashing and Pruning (DHP) algorithm which improves the deletion process of Apriori by creating hash tables with hash functions [5]. Dynamic Itemset Counting (DIC) saves itemsets in the form of a lattice. The algorithm identifies frequent itemsets at an early stage—the moment an individual segment is scanned, instead of until after a full scan of the database—thereby significantly reducing the computational cost that Apriori's iterative search requires. The reduced number of database scans naturally leads to higher efficiency. The Linear time Closed itemset Miner (LCM) algorithm uses array lists to compress data, and then enumerates frequent itemsets recursively to improve efficiency [6]. To achieve the

Y.C. Huang is with the Department of Industrial Management, National Pingtung University of Science and Technology, 1, Hseuh-Fu Road, Nei-Pu Hsiang, Pingtung 91201, Taiwan, (phone: +886-8-7703202; e-mail: tyh1332@mail.npust.edu.tw).

Y.L. Liao is with the Nursing Department, Antai Medical Care Cooperation Antai Tian-Sheng Memorial Hospital, Pingtung, Taiwan, (e-mail: ling@ms1.tsmh.com.tw).

C.S. Lin is with Department of Medical teaching and research, Antai Medical Care Cooperation Antai Tian-Sheng Memorial Hospital, Pingtung, Taiwan. (e-mail: a098123@mail.tsmh.org.tw)

same goal, Transaction Identifiers are used to reduce the number of database scans and shorten scan time [7]. This study develops DCSM in a similar attempt to enhance the performance of Apriori. Most significantly, DCSM modifies the data processing technique to remove the problem of repetition that is chiefly responsible for slowing down the calculations.

III. Data Cutting And Sorting Method

Notations:

DB : the original trade database

N_{db}: the quantity of items of the original trade database

O_i:the items of the original trade database , $i = 1 \ldots N_{db}$

N_R:the quantity of the trade of the original trade database

R_i: the set consisted of the items of the i^{th} trade

$I_{i,j}$: the j^{th} item of the i^{th} Boolen Matrix

$H_{i,j}$: the j^{th} item of the i^{th} large itemset

L_i : the i^{th} large itemset

NT_i : the quantity of items of the i^{th} Boolen Matrix

NI_i : the quantity of trades of the i^{th} Boolen Matrix

S_{min} : the minimum support

$T_{i,j}$: the j^{th} trade of the i^{th} Boolean matrix

E_{ijk} : the Boolean value of the k^{th} item of the j^{th} trade of the i^{th} Boolean matrix

$$E_{ijk} = \begin{cases} 0, \text{ the } k^{th} \text{ item of the } j^{th} \text{ trade of the } i^{th} \text{ Boolean matrix does not exist} \\ 1, \text{ the } k^{th} \text{ item of the } j^{th} \text{ trade of the } i^{th} \text{ Boolean matrix exists} \end{cases}$$

$SI_{i,j}$: the sum of the j^{th} item of the i^{th} Boolean matrix

$SP_{i,j}$: the support of the j^{th} item of the i^{th} Boolean matrix

BM_0 : the original Boolean matrix

BM_1 : the Boolean matrix of the *first* large itemset, L_1

BM_2 : the Boolean matrix of the *second* large itemset, L_2

A. The Procedures of DCSM

To improve the efficiency of the Apriori Algorithm, this paper proposes the data cutting and sorting method, DCSM, to verify the relationship between Metabolic Syndrome and Lifestyles. DCSM reduces the calculation time by getting rid of redundant data during the data mining process. In addition, DCSM minimizes the computational units by splitting the database and sorting data with support counts. The procedures of DCSM are illustrated as follow:

Step1 Transfer the original database into the original Boolean matrix, BM_0

```
/ * $DB \rightarrow BM_0$
for $i$ = 1 to $N_R$
  for $j$= 1 to $N_{db}$
    if $O_j \in R_i$
      $E_{0ij} = 1$
    else
      $E_{0ij} = 0$
    endif
  endfor
endfor
```

Step2 Establish the first large itemsets, L_1

```
$NT_1=NT_0$
for $x = 1$ to $NT_0$
  if $SP_{0,x} < S_{min}$
    delete $I_{0,x}$
    $NT_1= NT_1 - 1$
  endif
endfor
  /* the remained items become the elements of the first
     large itemsets, $H_{1,x}$ ,
  /* where $x$= 1, 2, …., $NT_1$
$NI_1=NI_0$
for $x$ =1 to $NI_0$
  if $SI_{0,x} = 1$
    delete $T_{0,x}$
    $NI_1= NI_1 - 1$
  endif
endfor
$I_{1,j} = sort( H_{1,x})$ /* recalculate the support of each item,
  /* the items of $BM_1$ are arranged in the order of their
     supports
```

Step3 Establish the second large itemsets, L_2

```
for $x=1$ to $NT_1$ - 1
/* extract the item which has the minimum support
  $NT_2=NT_1$
      delete $T_{1,i}$ where $E_{1ix} = 0$
  for $y=1$ to $NT_1 - x$
    if $SP_{1,y} \geq S_{min}$
  / * ($I_{1,x}$, $I_{1,y}$) become the element of the second large
      itemsets, $H_{2,x}$
```

Step4 Establish the third large itemsets, L_3

```
      for z=1 to $NT_2$ - $x$-$y$
        if $SP_{2z} \geq S_{min}$
        / *($I_{1,x}$, $I_{1,y}$, $I_{1,z}$) become the element of the third large
          itemsets, $H_{3,x}$
        endfor
      endif
    endfor
```

B. An Illustration of the Data Cutting and Sorting Method

The following example, as shown in Table I, illustrates the deduction procedure via DCSM. The minimum support count, S_{min}, is set at 2 in this example.

Step 1 Data conversion

Firstly, the items are converted into a Boolean matrix, as shown in Table II. The right column represents the item number of each event. The bottom row, SP_i, is the support count of each item.

TABLE I
Original Database

TID	Items
1	A B D H
2	A B C D F G H
3	A B C G
4	A B E G
5	A

TABLE II
Original Boolean Matrix

TID	A	B	C	D	E	F	G	H	Count
1	1	1	0	1	0	0	0	1	4
2	1	1	1	1	0	1	1	1	7
3	1	1	1	0	0	0	1	0	4
4	1	1	0	0	1	0	1	0	3
5	1	0	0	0	0	0	0	0	1
Support	5	4	2	2	1	1	3	2	

Step 2 Establish the first large itemsets, L_1

Items E and F are deleted from Table II because their support counts are less than the minimum support, S_{min}. Sequentially the remaining items are arranged by support count in descending order, as shown in Table III.

TABLE III
The First Largfe Itemsets

TID	A	B	G	C	D	H	Count
1	1	1	0	0	1	1	4
2	1	1	1	1	1	1	6
3	1	1	1	1	0	0	4
4	1	1	1	0	0	0	3
5	1	0	0	0	0	0	1
Support	5	4	3	2	2	2	

Step 3 Establish the first Reduction Matrix

There are no pairs when the item number of each event is less than 2. Therefore event 5 is deleted from Table III because its item number is 1. Item H, which has the lowest support, is selected to be the conjunctive item. Events 1 and 2, which contain H, are selected to establish a new reduction matrix, as shown in Table IV.

Step 4 Establish the second large itemsets, L_2

Items G and C are deleted for their support counts are less than S_{min}. Therefore, the second large itemsets, L_2, are composed of itemsets (H, A), (H, B), (H, D), as shown in Table V.

TABLE IV
The First Reduction Matrix For H

H						
TID	A	B	D	G	C	Count
1	1	1	1	0	0	3
2	1	1	1	1	1	5
Support	2	2	2	1	1	

TABLE V
The Second Large Itemsets For H

H				
TID	A	B	D	Coun t
1	1	1	1	3
2	1	1	1	5
Suppor	2	2	2	

Step 5 Establish the second Reduction Matrix, BM_2

Item D, which has the lowest support, is selected to be the second conjunctive item. Events 1 and 2, which contain D, are selected to establish a new reduction matrix, as shown in Table VI. The support counts of items A and B are not less than the minimum support. Therefore, the third large itemsets, L_3, are composed of itemsets (H, D, A) and (H, D, B).

TABLE VI
The Second Reduction Matrix For H,D

H, D			
TID	A	B	Coun t
1	1	1	2
2	1	1	2
Suppor	2	2	

Step 6 Search recursively the items of the third large itemsets

The second large itemsets are scanned recursively to look for items of the third large itemsets. In Table V, the mining operation moves from H to B as the former has already been processed, thus forming the second reduction matrix for HB, as shown in Table VII. The support count of A is not lower than the minimum support. Therefore, the third large itemsets, L_3, are composed of itemset (H, B, A).

Step 7 Return to the first large itemsets

After the calculation of item H is completed in Step 3, item D, which is to the left of H, is selected as the next conjunctive item and the first reduction matrix for D is established, as Table VIII shown. Steps 3-7 are subsequently repeated to search for the second and the third large itemsets centering on D. After the search for D is completed, items C and G are selected—in accordance with the order established in Table III —to initiate new searches for the third large itemsets centering on them.

TABLE VII
THE SECOND REDUCTION MATRIX FOR H,B

H, B		
TID	A	Count
1	1	1
2	1	1
Support	2	

TABLE VIII
THE FIRST REDUCTION MATRIX FOR D

D					
TID	A	B	G	C	Count
1	1	1	0	0	2
2	1	1	1	1	4
Support	2	2	1	1	

IV. PERFORMANCE ANALYSIS

This empirical study is based on the health examination records of 227 employees in an electronics company and on a questionnaire survey on their lifestyles. The questionnaire includes questions about 18 attributes, such as smoking, sleep, diet, shifts and so forth. The data are analyzed with DCSM and Apriori respectively. It turns out that both algorithms discover the same association rules between lifestyles and metabolic syndrome (MS), as shown in Table IX (only the top five supports are listed here).

Figure 1 shows that, with different support counts, DCSM consistently uses up far less calculation time than Apriori, thus proving that the efficiency of DCSM is superior to that of the traditional Apriori algorithm.

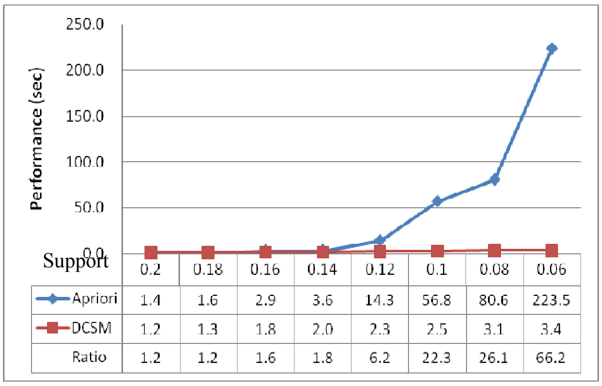
V.CONCLUSION

DCSM speeds up the search in the mining process by partitioning the database (Ye and Chiang, 2006) and by deleting items whose support counts are lower than the minimum support (such as items E and F in Step 2) and transactions whose item number is lower than 2 (such as event 5 in Step 3). Besides, DCSM arranged items in the order of their support value (such as Step 2). In such a way, the items to be processed are reduced considerably, so is the search time due to deletion of repetition.

The association rules derived by DCSM and by Apriori respectively are the same, but it is evident that DCSM has better efficiency than Apriori.

TABLE IX
ASSOCIATION RULES

Association Rules (L_3)		Support(%)	Confidence(%)
BMI>27, Exercise≦ 1,	MS	11.89	61.36
BMI>27, Shift-Work,	MS	10.57	55.81
BMI>27, Male,	MS	10.13	58.97
Male, No Coffee,	MS	8.81	57.14
BMI>27, with family history of MS,	MS	7.05	100.00



Support	0.2	0.18	0.16	0.14	0.12	0.1	0.08	0.06
Apriori	1.4	1.6	2.9	3.6	14.3	56.8	80.6	223.5
DCSM	1.2	1.3	1.8	2.0	2.3	2.5	3.1	3.4
Ratio	1.2	1.2	1.6	1.8	6.2	22.3	26.1	66.2

Fig. 1 Calculation time of DCSM and Apriori

ACKNOWLEDGMENT

This research was partially supported by National Science Council, Taiwan, ROC. (NSC 98-2221-E-020-006 ; NSC 99-2221-E-020-021).

REFERENCES

[1] R. Agrawal, R. Srikant, “Fast Algorithms for Mining Association Rule,” in Proceedings of 20th International Conference on Very Large Data Bases, pp. 487-499, 1994.

[2] W. J. Frawley, G. Piatetsky-Shapiro, C. J. Matheus, “Knowledge discovery databases: An overview,” AI Magazine. vol. 13, no. 3, pp. 57-70, 1992.

[3] F. H. Grupe,; M. M. Owrang, “ data base mining discovering new knowledge and competitive advantage,” Information systems management, vol. 12, no. 4, pp. 26-31, 1995.

[4] M.H. Smith, and W. Pedrycz, “Expanding the meaning of and applications for data mining,” International Conference on Systems, Man, and Cybernetics, vol. 3, 2000, pp. 1874.

[5] J. S. Park, M. Chen and P. S. Yu, “An effective hash-based algorithm for mining association rules,” in Proceedings of the 1995 ACM SIGMOD international conference on Management of data.

[6] U. Takeaki, A. Taisuya , U. Yuzo, A. Hiroki “ LCM : AN Efficient Algorithm for Enumerating Frequent Closed Item Sets,” In Proc. IEEE ICDM99 Workshop FIMI’ 03,2003.

[7] K. M. Yu, J. Zhou, T. P. Hong, J. L. Zhou, “A Load-Balanced Distributed Parallel Mining Algorithm,” Expert Systems with Applications, vol. 37, issue 3, pp.2459-2464, 2010.