CAARMSAD : Combinatorial Approach of Association Rule Mining for Sparsely Associated Databases

P. R. Pal[1], R.C. Jain[2]

Abstract

Classification based on association rule mining has now become a very powerful technique of data mining for extraction of accurate knowledge from very large databases. First it finds all association rules in a training database then uses these rules to classify an unknown object of test database. Support_threshold and confidence_threshold are used to specify the user's interests. This paper proposes an algorithm CAARMSAD to mine the association rules in sparsely associated databases by applying minimum efforts. It uses the concepts of combinatorial mathematics to generate only those candidate itemsets whose sizes are up to the probable size of maximal frequent itemsets. Since CAARMSAD reduces the large number of candidate itemsets generations, it is very efficient for association rule mining in sparsely associated databases.

Keywords : combinatorial mathematics, classification, association rule mining, sparsely associated databases.

1. Introduction

Data mining [6, 8] is a popular area of research and development in computer science nowadays, and attracting more and more attention of peoples from different disciplines. Data mining aims to extract various models of hidden and interesting knowledge (i.e. patterns, rules, trends, etc.) from the databases, where the volume of a collected database can be measured in gigabytes. Association rule mining [1, 4, 6, 8] is a data mining technique that aims to identify all association rules in a given database. An association rule is a typical pattern that describes co-occurring relationships among database attributes. One application of association rule mining is to define classification rules that will classify database records. These kinds of association rules are called classification association rules or class association rules or CARs. The process to build a classifier using class association rules is called associative classification or classification based on association rules or classification association rule mining or simply classification rule mining [2, 3, 5, 7, 9, 10]. Sometime it is also referred by integration of classification and association rule mining [2] or simply integration of data mining techniques.

Now classification of association rule mining has become a well-known data mining technique for extraction of hidden classification association rules. Classification [6] actually involves two phases; training and test. In training phase the rule set is generated from the training data, where each rule associates a pattern to a class. In the test phase the generated rule set is used to decide the class of a test data record to which it belongs.

In the past decade, a number of classification based on association rules algorithms have been developed. Some most popular algorithms are CBA (Classification Based on Association)[2], CAEP (Classification by Aggregating

[1]Assistant Professor, MCA Department, Lakshmi Narain College of Technology, Bhopal, M.P. India. e-mail: prpal@rediffmail.com
[2]Professor and Head, Department of Computer Applications, Samrat Ashok Technological Institute Vidisha, M.P. India. e-mail: dr.jain.rc@gmail.com

Emerging Patterns)[3], CMAR (Classification based on Multiple Association Rules)[5], CPAR (Classification based on Predictive Association Rules)[7], Corrclass (correlated association rule mining for classification)[9] etc. Each algorithm followed by other performs more efficiently than previous one. But still there is possibility to design new algorithms that may perform even with more efficiency.

Association rule mining algorithm that was designed by Pal and Jain [13], mines all association rules in a most efficient manner. The algorithm first finds all candidate itemsets using a combinatorial method, and then finds all frequent itemsets from these candidate itemsets. Subsequently, it uses these frequent itemsets to generate all association rules in a systematic way. We feel that this algorithm will work in very efficient manner for those databases in which there is dense associations among the items. But it may perform large number of unnecessary and lengthy computations for those databases that are having sparse associations among the items. Large number of unnecessary and lengthy computations may be in terms of generating all candidate itemsets and finding their frequency in the database, subsequently finding frequent itemsets.

We can overcome this problem if we do not generate those candidate itemsets whose size is greater than the size of transactions having support less than min_sup. Also there is no need to find the frequency of these itemsets in the database. It is leaving a large number of computations in finding candidate itemsets and also reducing large number of comparisons while finding frequency of these itemsets through physical scan of the database. The fundamental logic behind this method is that an itemset can never be frequent if its size is greater than the size of transactions having support less than min_sup. In this paper we have designed an algorithm based on this logic.

2. Association Rule Mining

Association rule mining is a well-established Data Mining technique. The objective of association rule mining is to extract association rules, typically defined according to the co-occurrences of binary valued attributes, from a transaction database (D_T). Let I = {a1, a2, ..., a_n} be a set of items (database attributes), and T = {T1, T2, ..., Tm} be a set of transactions, D_T is described by T, where each $T_i \in T$ contains a set of items I' and $I' \subseteq I$. In association rule mining, two threshold values are usually used to determine the significance of an association rule.

- Support: The frequency that the items occur or co-occur in T. A support threshold min_sup, defined by the users, is used to distinguish frequent itemsets from the infrequent ones. A set of items S is called an itemset, where $S \subseteq I$, and $V_{ai} \in S$ co-occur in T. If the occurrence of some S in T exceeds min_sup, we say that S is a frequent itemset.
- Confidence: represents how "strongly" an itemset X implies another itemset Y, where X, $Y \subseteq I$; and $X \cap Y = \{\phi\}$. A confidence threshold min_conf, supplied by the user, is used to distinguish high confidence association rules from low confidence association rules.

An association rule X => Y is valid when the support for the co-occurrence of X and Y exceeds min_sup, and the confidence of this association rule exceeds min_conf. The computation of support is: $(X \cup Y)$ / (total number of transactions in D_T). The computation of confidence is: support $(X \cup Y)$ / support (X). Informally, X => Y can be interpreted as "if X exists, it is likely that Y also exists".

3. Combinatorial Mathematics

A combination is an unordered collection of unique sizes. (An ordered collection is called permutations.) Given S, the set of all possible unique elements, a combination is a subset of the elements of S. The order of elements is not considered in combinations. (Two or more subsets with same elements in different orders are considered as one combination.) For example ab, and ba represents two different permutations but only one combination i.e. ab or ba. Also elements cannot be repeated in a combination. Every element appears uniquely once; they because the combinations are defined by the set of elements contain this in unordered manner. For example, aba is not a legal combination; the legal combination is ab or ba.

A k_combination is a subset with k elements. The number of k_combinations (each of size k) from a set S with n elements (size n) is the binomial coefficient and represented as follows:

$$^{n}C_{k} = \frac{n!}{k!\,(n-k)!}$$

k_combination is also defined as the k elements taken at a time out of n elements.

The sum of all the possible combinations of a set S with n elements can be calculated by adding all the 0_combinations, 1_combinations, 2_combinations, up to n_combinations. Sum of all the combinations is equal to 2^n. It can be represented as follows:

$$^{n}C_{0} + {}^{n}C_{1} + {}^{n}C_{2} + \ldots\ldots + {}^{n}C_{n} = 2^{n}$$

For example, set S has 3 elements i.e. S = (a, b, c). The set of all possible combinations of S is C = (ϕ, a, b, c, ab, bc, ac, abc), i.e. there are total 8 combinations, which is equal to 2^3.

The above discussion finds the number of combinations taking k elements at a time out of n elements. It also finds the total number of all the possible combinations for which k will vary from 0 to n that is 2^n.

The elements in each combination of a set S with n unique elements can be found as follows: Generate 2^n unique binary patterns. Each binary pattern will consist an n digits binary string of 0 and 1. Here each digit of the binary pattern corresponds to a unique element of the set S i.e. 1st digit of binary pattern corresponds to 1st element of S, 2nd digit of binary pattern corresponds to 2nd element of S and so on up to nth digit of the binary pattern. Here, a binary pattern represents a combination and each 0 in a binary pattern shows the absence of corresponding element and each 1 shows the presence of the corresponding elements in that combination. Therefore, in each binary pattern, the elements having corresponding binary digits 1's are combined to form the subset of elements in that combination because each 1 in the binary pattern represents that corresponding element to be included in the combination. In such a way we will find a subset of elements in each combination. It will produce total 2^n subsets (each subset will represent a combination) that will represent the set of all combinations.

For example, let S = (a, b, c) as n = 3, the total numbers of combinations are $2^3 = 8$. The unique binary patterns for n = 3 can be represented as:

B = (000, 001, 010, 011, 100, 101, 110, 111)

It gives $C_0 = ϕ$, $C_1 = c$, $C_2 = b$, $C_3 = bc$, $C_4 = c$, $C_5 = ac$, $C_6 = ab$, $C_7 = abc$.

Now C = (ϕ, a, b, c, ab, bc, ac, abc).

Here C is containing all the possible subsets of combinations for set S.

Actually, the combinatorial study tells about the number of combinations (^nC_k) to be generated, but it doesn't tells any thing that how the subsets of these combinations will be generated. In this section we have explored the systematic method that generates the subsets of these combinations.

4. Association Rule Mining For Sparsely Associated Database

Our algorithm is based on the combinatorial approach, which is briefly discussed in the section 3. Let there are n items in the itemset i.e. $I = (I_1, I_2, I_3, \dots I_n)$, support threshold is min_sup and confidence threshold is min_conf. The algorithm CAARMSAD is given in figure 1.

This algorithm, first finds frequency (F_k) of all transactions according to the number of items they contain in reverse order (line 1-3). Then it calculates its cumulative frequency (CF_k) to find the probable size of maximal frequent itemset (line 4-7). Now the algorithm generates the candidate itemsets up to this probable size (line 8-9). Next it scans the training database to find the frequency of only these candidate itemsets (line 10-11). It gives all the frequent itemsets (line 12-17), subsequently all association rules also (line 18-31).

The process can better be understood by following example.

Example : Let we have a transactional database as depicted in table 1.

Table 1

Tid↓	A1	A2	A3	A4	A5
1.	1	0	0	0	1
2.	0	1	0	1	0
3.	0	0	0	1	1
4.	0	1	1	0	0
5.	0	0	0	0	1
6.	1	0	0	0	0
7.	0	1	0	0	0
8.	0	0	0	1	0
9.	1	0	0	0	1
10.	0	0	1	0	1
11.	0	0	1	0	1
12.	0	0	0	1	1
13.	0	1	0	1	0
14.	1	0	1	0	1
15.	1	1	1	0	1

In above transactional database, there are total 5 unique items in item set i.e. I = (A1, A2, A3, A4, A5). Let we have the min_sup = 3 and the min_conf = 70%.

Step 1. Frequency table will be as follows:

No of items k	Frequency F_k
5	0
4	1
3	1
2	9
1	4

Figure 1: The ARMSAD ALgorithm

```
ALGORITHM:
Step 1. Find frequency table $F_k$, i.e. frequency $F_k$ of k-items in the training database ($n \geq k \geq 1$).
        1.  Arrange the frequency table in descending order according to number of items in itemset.
        2.  Set $F_k = 0$, for all k such as ($n \geq k \geq 1$).
        3.  Scan database one by one record and if a record contains k-items increment corresponding $F_k$.
Step 2. Find cumulative frequency table $CF_k$.
        4.  Calculate cumulative frequency $CF_k$.
Step 3. Find the size of the itemset such as any super-itemset cannot be frequent
        5.  $k = n$;
        6.  while ($CF_k$ < min_sup)        $k = k - 1$;
        7.  $m = k$;
Step 4. Find candidate k-itemsets $C_k$ ($0 \leq k \leq m$); i.e. $C_0, C_1, C_2, \dots C_m$.
        8.  Generate $2^n$ unique binary patterns.
        9.  In each binary pattern, there are k numbers of 1 and if ($k \leq m$) then combine these
            corresponding k items to form k-itemset and add it to $C_k$.
Step 5. Scan database D to find the frequency of each itemset of $C_k$ ($0 \leq k \leq m$).
        10. Read a record from database D.
        11. If an itemset $C_{ki}$ (ith k-itemset of $C_k$) is contained by the record, increment its corresponding
            frequency counter $F_{ki}$ ($1 \leq k \leq m$ and $1 \leq i \leq {}^nC_k$).
Step 6. Find all frequent itemsets i.e. $L_1, L_2, L_3, \dots L_k$ from $C_1, C_2, C_3, \dots C_m$ ($k \leq m$).
        12. $k = 1$;
        13. Do {
        14.     For each $C_{ki}$, if ($F_{ki} \geq$ min_sup) then add this $C_{ki}$ and its $F_{ki}$ to $L_k$ ($1 \leq k \leq m$ & $1 \leq i \leq {}^nC_k$)
        15.     $k = k + 1$;
        16.     } While ($L_{k-1} \neq$ null)
        17.     $k = k-2$;
Step 7. Generate association rules and prune also.
        18. $i = 2$;
        19. while ($i \leq k$) {
        20.         $j = 1$;
        21.         do {
        22.             generate $2^i$ unique binary patterns for jth itemset of $L_i$, i.e. $L_{ij}$;
        23.             for each binary pattern (2 to $2^i - 1$)
        24.                 { form rule of the form (all items having value 0 with $\Lambda$ operator)[say
                              A] $\Rightarrow$ (all items having value 1 with $\Lambda$ operator)[say B];
        25.                 find confidence of the rule i.e. confi ($A \Rightarrow B$) = $F(A \cup B) / F(A)$;
        26.                 if (confi ($A \Rightarrow B$) $\geq$ min_conf) then add this rule to association rule
                            set ARS with their confidence;
        27.                 }
        28.             $j = j + 1$;
        29.             } while (Lij $\neq$ null)
        30.         $i = i + 1$;
        31.         }
```

Figure 1: The ARMSAD Algorithm

Step 2. The cumulative frequency table can be calculated as under:

No of items k	Frequency F_k	Cum Freq CF_k
5	0	0
4	1	1
3	1	2
2	9	11
1	4	15

Step 3. Size of the itemset (such as any super-itemset can't be frequent) is:

$m = 2$.

Step 4. There will be 6 candidate itemsets i.e. C_0, C_1, C_2, C_3, C_4, C_5. The 2^5 unique binary combinations are as follows:

00000, 00001, 00010, 00011, 00100, 00101, 00110, 00111, 01000, 01001, 01010, 01011, 01100, 01101, 01110, 01111, 10000, 10001, 10010, 10011, 10100, 10101, 10110, 10111, 11000, 11001, 11010, 11011, 11100, 11101, 11110, 11111.

The above pattern will generate the following candidate sets.

$C_0 = (\phi)$,
$C_1 = (A1, A2, A3, A4, A5)$,
$C_2 = (A_1A_2, A1A3, A1A4, A1A5, A2A3, A2A4, A2A5, A3A4, A3A5, A4A5)$.

Step 5. Scan of database D will produce the frequency of itemsets as follows:

$C_0 = (\phi)$,
$C_1 = (A1 / 5, A2 / 5, A3 / 5, A4 / 5, A5 / 9)$,
$C_2 = (A1A2 / 1, A1A3 / 2, A1A4 / 0, A1A5 / 4, A2A3 / 2, A2A4 / 2, A2A5 / 1, A3A4 / 0, A3A5 / 4, A4A5 / 2)$.

Step 6. The above candidate itemsets will generate the following frequent itemsets:

$L_1 = (A1 / 5, A2 / 5, A3 / 5, A4 / 5, A5 / 9)$,
$L_2 = (A1A5 / 4, A3A5 / 4)$.

Step 7. Finally, the association rule set will be as follows:

ARS = {(A1 => A5, 80%),
(A3 => A5, 80%)}.

5. Evaluation Of CAARM Algorithm

Association rule mining is a complex method of knowledge discovery process. The most complex and time-consuming part of the association rule mining is finding of frequent patterns or itemsets, because it requires the physical scan of database. Various algorithms have been discovered in last several years to find all the frequent itemsets (and association rules also) in very fast manner. Some of the most popular algorithms are Apriori [1] and FPTree [4] etc. Apriori [1] needs at least k number of database scans and also complex join and prune operation after each scan to find all frequent 1_itemsets up to maximal frequent k_itemsets. FPTree [4] needs two number of database scans, and it also creates a complex tree structure in memory to store items and their frequency. Each time, a complex tree traversal is needed to find a frequent itemset. For very large databases, some time it is difficult to store the tree in the memory.

CAARM [13] overcomes with these problems and mines the association rules in the fastest manner. The most beautiful feature of CAARM is that it scans the database only once to find all association rules. It also doesn't needs any new structure to be designed or other data structure is used to store the items in the memory during the process. It also generates the pruned set of rules.

In sparsely associated databases CAARMSAD works better than CAARM, as it generates only those candidate itemsets whose size is up to the probable size of maximal frequent itemsets. Since any super itemset of maximal frequent itemset can never be frequent, therefore they are removed in the candidate itemset generation step. Large databases will have large number of items (let n) in its itemset, subsequently very large number of candidate itemsets (combinations i.e. 2^n). Out of these 2^n candidate itemsets only several itemsets will be frequent.

Let m be probable size of frequent itemset, the CAARMSAD generates only 2^m candidate itemsets and removes $2^n - 2^m$ candidate itemsets in candidate itemset generation step. Since m is small quantity in compare to n for sparsely associated databases, subsequently 2^m will be very small to the 2^n. Therefore CAARMSAD reduces $2^n - 2^m$ number of candidate itemset generation, subsequently finding the frequency of these itemsets in the training database. It shows that CAARMSAD saves $(2^n - 2^m)$, which is a very big amount and are involved in CAARM. It means our algorithm will be faster with this difference to the CAARM.

6. Conclusion

This paper proposes a new and efficient approach of association rule mining for sparsely associated databases. It uses combinatorial mathematics to generate only those candidate item sets whose size is up to probable size of maximum frequent item set. In this way it saves a lot of efforts in generation of frequent itemsets and subsequently in generation of association rules also. Generation, therefore it will be very efficient in compared to. Since a lot of efforts are reduced in association rule set other approaches available so far, especially for the sparsely associated databases.

References

[1] Agrawal. R. and Srikant. R, *"Fast algorithms for mining association rules"*, In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), PP. 487-499, Santiago, Chile, September 1994.

[2] Liu B, Hsu W and Ma. Y, *"Integrating classification and association rule mining"*, In Proceedings of the Fourth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, PP. 80-86, New York, NY, August 1998.

[3] Dong. G, Zhang. X, Wong. L and Li. J, *"CAEP: classification by aggregating emerging patterns"*, In Proceedings of The Second International Conference on Discovery Science (DS'99), PP. 43-55, Tokyo, Japan, December 1999.

[4] Han. J, Pei J and Yin. Y, *"Mining frequent patterns without candidate generation"*, In proceedings of international conference on management of data (ACM SIGMOD'00), PP. 1-12, Dallas, TX, May 2000.

[5] Li. W, Han. J and Pei. J, *"CMAR: Accurate and efficient classification based on multiple class-association rules"*, In Proceedings of The 2001 IEEE International Conference on Data Mining (ICDM'01), PP. 369-376, San Jose, CA, November 2001.

[6] Han. J. and Kamber. M, *"Data mining, Concepts and techniques"*, Academic Press, 2003.

[7] Yin X and Han. J, *"CPAR: Classification based on predictive association rules"*, In Proceedings of

SIAM International Conference on Data Mining (SDM'03), PP. 331-335, San Francisco, CA, May 2003.

[8] Hand. D., Mannila. H and Smyth, *"P. Principles of Data Mining"*, Prentice Hall of India, 2004.

[9] Zimmerman. A and De Raedt L, *"Corclass: correlated association rule mining for classification"*, In proceedings of 7th international conference on discovery science (DS'04), PP. 60-72, Padova, Italy, Oct. 2004.

[10] Veloso. A, Meira. W, *"Rule Generation and Selection Techniques for Cost Sensitive Associative Classification"*, 19th Brazilian Symposium on Software Engineering, 2005.

[11] Leng. P, Coenen. F, *"The effect of threshold values on association rule based classification accuracy"*, Data and Knowledge Engineering, 2006.

[12] Pal. P. R, Jain. R. C, *"Framework for Classification Based on Association Rules"*, in the proceedings of the International Conference on Soft-computing and Intelligent Systems (ICSCIS-07), PP. 64-67, Jabalpur India, December 2007.

[13] Pal. P. R, Jain. R. C, *"CAARM: "Combinatorial Approach of Association Rule Mining"* under communication.

Authors Biography

R. C. Jain, M.Sc., M. Tech., Ph. D., is a Professor and Head, Department of Computer Applications at S.A.T.I. (Engg. College) Vidisha (M. P.) India. He is Dean, Faculty of Computer and Information Technology, Rajeev Gandhi Technical University Bhopal (M.P.). He has 35 years of teaching experience. He is actively involved in Research with area of interest as Fuzzy Systems, DIP, Mobile Computing, Data Mining and Adhoc Networks. He has published more than 125 research papers, produced 7 Ph. Ds. And 10 Ph. Ds are under progress.

P. R. Pal, M.C.A., Ph.D. Scholar, is an Assistant Professor in MCA Department of L.N.C.T. Bhopal (M. P.) India. He is working on Data Mining Algorithms to improve their efficiency under guidance of Dr. R. C. Jain. His area of interest is DBMS, Data Mining, Computer Graphics and Computer Architecture.