ISSN:0975-9646

A Generalized Association Rule Mining Framework for Pattern Discovery

[1] Ravindra Changala, [2]Dr. D Rajeswara Rao, [3]Annapurna Gummadi

[1, 3] *PhD (CSE) Research Scholars, Dept of CSE, KL University, Guntur,*
[2]*Professor & RPAC Chairman, CSE Department, KL University, Guntur,*
[1]*Assistant Professor, Department of IT, GNITC, Hyderabad, India*

Abstract: The strength of data mining is association rule mining techniques which describe the associations among items in a database and are useful to identify domain knowledge hidden in large volume of data efficiently. It is difficult to discovery association rules without providing support and confidence framework where a minimum support must be supplied to start the discovery process. In this paper, we propose a novel framework; in this associations are discovered based on logical implications. The principle of the approach considers that an association rule should only be reported when there is enough logical evidence in the data. To do this, we consider both presence and absence of items during the mining. The proposed algorithm discovers the natural threshold based on observation of data set. The different intelligence models can be used in conjunction with the proposed algorithm in determining the target item(s) to be considered during the mining process. It provides a logical underpinning to the discovery process of patterns. Currently, the illustration of the mapping of constraints to the discovery process in this paper is based on support value.

Key Words: Data mining, association rules, minimum support, patterns, logical implications.

1. INTRODIUCTION

The use of association rule mining technique is to describe the associations among items in a database. Thus, association rule mining is useful to identify domain knowledge hidden in large volume of data efficiently. The discovery of association rules is typically based on the support and confidence framework where a minimum support (min sup) must be supplied to start the discovery process.

A priori is a representational algorithm based on this framework and many other algorithms are a priori-like. Without this threshold specified, typically, no association rules can be discovered because the procedure to discover the rules will quickly exhaust the available resources. In this paper, we propose a novel framework to address the above issues by removing the need for a minimum support threshold. Associations are discovered based on logical implications. The principle of the approach considers that an association rule should only be reported when there is enough logical evidence in the data. To do this, we consider both presence and absence of items during the mining.

By considering this new approach in finding data pattern, a solution toward fulfilling domain-driven data mining requirements can be made. The proposed algorithm suggests a solution in two areas: The proposed algorithm discovers the natural threshold based on observation of data set. The different intelligence models can be used in conjunction with the proposed algorithm in determining the target item(s) to be considered during the mining process. It provides a logical underpinning to the discovery process of patterns. Currently, the illustration of the mapping of constraints to the discovery process in this paper is based on support value. The use of association rule mining technique is to describe the associations among items in a database. These associations represent the domain knowledge encapsulated in databases. Identifying domain knowledge is important because these knowledge rules usually are known only by the domain experts over years of experience. Thus, association rule mining is useful to identify domain knowledge hidden in large volume of data efficiently. The discovery of association rules is typically based on the support and confidence framework where a minimum support (min sup) must be supplied to start the discovery process [1]. A priori is a representational algorithm based on this framework and many other algorithms are a priori-like. Without this threshold specified, typically, no association rules can be discovered because the procedure to discover the rules will quickly exhaust the available resources.

2. RELATED WORK

Rule Mining Framework:

We propose a novel association rule mining framework that can discover association rules without the need for a minimum support threshold. This enables the user, in theory, to discover knowledge from any transactional record without the background knowledge of an application domain usually necessary to establish a threshold prior to mining. To introduce our framework, this section starts with the distinction between an association rule and the different modes of an implication as defined in propositional logic. The topic of implication from logic is raised because our proposed mining model is based on an association rule's ability to be mapped to a mode of implication. If an association can be mapped to an implication, then there is reason to report this relation as an association rule. Otherwise, without a priori such as the minimum support threshold, many association rules would be found, and we would need to report all of them. An implication having a rule where the left-hand side is connected to the right-hand side correlates two item sets together. This implication exists because it is true according to logical grounds, follows a specific truth table value, and does not need to be judged to be true by a user.

The rule is reported as an interesting association rule if its corresponding implication is true.

Association rules are like classification rules. You could find them in the same way, by executing a divide-and-conquer rule-induction procedure for each possible expression that could occur on the right-hand side of the rule. But not only might any attribute occur on the right-hand side with any possible value; a single association rule often predicts the value of more than one attribute. To find such rules, you would have to execute the rule-induction procedure once for every possible *combination* of attributes, with every possible combination of values, on the right-hand side.

3. ASSOCIATION RULES

Shortly we will explain how to generate these item sets efficiently. But first let us finish the story. Once all item sets with the required coverage have been generated, the next step is to turn each into a rule, or set of rules, with at least the specified minimum accuracy. Some item sets will produce more than one rule; others will produce none. For example, there is one three-item set with coverage of 4.

humidity = normal, windy = false, play = yes

This set leads to seven potential rules:

One-item sets	Two-item sets	Three-item sets	Four-item sets
outlook = sunny (5)	outlook = sunny temperature = mild (2)	outlook = sunny temperature = hot	outlook = sunny temperature = hot
outlook = overcast (4)	outlook =sunny temperature = hot (2) play = no (2)	outlook=sunny temperature = hot windy = false	outlook = sunny humidity = high windy = false play = no (2)
outlook = rainy (5)	outlook = sunny humidity = normal (2)	outlook = sunny humidity = normal play = yes (2)	outlook = sunny temperature = hot windy = false play = yes (2)
temperature = cool (4)	outlook = sunny humidity = high (3)	outlook = sunny humidity = high windy = false (2)	outlook = rainy temperature = mild windy = false play = yes (2)
temperature = mild (6)	outlook = sunny windy = true (2)	outlook = sunny humidity = high play = no (3)	outlook = rainy windy = false play = yes (2)

If humidity = normal and windy = false then play = yes 4/4
If humidity = normal and play = yes then windy = false 4/6
If windy = false and play = yes then humidity = normal 4/6
If humidity = normal then windy = false and play = yes 4/7
If windy = false then humidity = normal and play = yes 4/8
If play = yes then humidity = normal and windy = false 4/9
If – then humidity = normal and windy = false and play = yes 4/12

The figures show the number of instances for which all three conditions are true that is, the coverage—divided by the number of instances for which the conditions in the antecedent are true. Interpreted as a fraction, they represent the proportion of instances on which the rule is correct—that is, its accuracy. Assuming that the minimum specified accuracy is 100%, only the first of these rules will make it into the final rule set.

	Association rule			Coverage	Accuracy
1	humidity = normal windy = false	⇒	play = yes	4	100%
2	temperature = cool	⇒	humidity = normal	4	100%
3	outlook = overcast	⇒	play = yes	4	100%
4	temperature = cool play = yes	⇒	humidity = normal	3	100%
5	outlook = rainy windy = false	⇒	play = yes	3	100%
6	outlook = rainy play = yes	⇒	windy = false	3	100%
7	outlook = sunny humidity = high	⇒	play = no	3	100%
8	outlook = sunny play = no	⇒	humidity = high	3	100%
9	temperature = cool windy = false	⇒	humidity = normal play = yes	2	100%
10	temperature = cool humidity = normal windy = false	⇒	play = yes	2	100%
11	temperature = cool windy = false play = yes	⇒	humidity = normal	2	100%
12	outlook = rainy humidity = normal windy = false	⇒	play = yes	2	100%
13	outlook = rainy humidity = normal play = yes	⇒	windy = false	2	100%
14	outlook = rainy temperature = mild windy = false	⇒	play = yes	2	100%
15	outlook = rainy temperature = mild play = yes	⇒	windy = false	2	100%
16	temperature = mild windy = false play = yes	⇒	outlook = rainy	2	100%
17	outlook = overcast temperature = hot	⇒	windy = false play = yes	2	100%
18	outlook = overcast windy = false	⇒	temperature = hot play = yes	2	100%
19	temperature = hot play = yes	⇒	outlook = overcast windy = false	2	100%
20	outlook = overcast temperature = hot windy = false	⇒	play = yes	2	100%
21	outlook = overcast temperature = hot play = yes	⇒	windy = false	2	100%
22	outlook = overcast windy = false play = yes	⇒	temperature = hot	2	100%
23	temperature = hot windy = false play = yes	⇒	outlook = overcast	2	100%
24	windy = false play = no	⇒	outlook = sunny humidity = high	2	100%

Association rules for the weather data

Generating rules efficiently

We now consider in more detail an algorithm for producing association rules with specified minimum coverage and accuracy. There are two stages: generating item sets with the specified minimum coverage, and from each item set determining the rules that have the specified minimum accuracy. The first stage proceeds by generating all one-item sets with the given minimum coverage and then using this to generate the two-item sets (second column), three-item sets (third column), and so on. Each operation involves a pass through the dataset to count the items in each set, and after the pass the surviving item sets are stored in a hash table a standard data structure that allows elements stored in it to be found very quickly. From the one-item sets, candidate two-item sets are generated, and then a pass is made through the dataset, counting the coverage of each two-item set; at the end the candidate sets

with less than minimum coverage are removed from the table. The candidate two-item sets are simply all of the one-item sets taken in pairs, because a two-item set cannot have the minimum coverage unless both its constituent one-item sets have minimum coverage, too. This applies in general: a three-item set can only have the minimum coverage if all three of its two-item subsets have minimum coverage as well, and similarly for four-item sets.

An example will help to explain how candidate item sets are generated. Suppose there are five three-item sets (A B C), (A B D), (A C D), (A C E), and (B C D) where, for example, A is a feature such as *outlook = sunny*. The union of the first two, (A B C D), is a candidate four-item set because its other three item subsets (A C D) and (B C D) have greater than minimum coverage. If the three-item sets are sorted into lexical order, as they are in this list, then we need only consider pairs whose first two members are the same. For example, we do not consider (A C D) and (B C D) because (A B C D) can also be generated from (A B C) and (A B D), and if these two are not candidate three-item sets then (A B C D) cannot be a candidate four-item set. This leaves the pairs (A B C) and (A B D), which we have already explained, and (A C D) and (A C E). This second pair leads to the set (A C D E) whose three-item subsets not all have the minimum coverage, so it is discarded. The hash table assists with this check: we simply remove each item from the set in turn and check that the remaining three-item set is indeed present in the hash table. Thus in this example there is only one candidate four-item set, (A B C D). Whether or not it actually has minimum coverage can only be determined by checking the instances in the dataset. The second stage of the procedure takes each item set and generates rules from it, checking that they have the specified minimum accuracy. If only rules with a single test on the right-hand side were sought, it would be simply a matter of considering each condition in turn as the consequent of the rule, deleting it from the item set, and dividing the coverage of the entire item set by the coverage of the resulting subset—obtained from the hash table—to yield the accuracy of the corresponding rule. Given that we are also interested in association rules with multiple tests in the consequent, it looks like we have to evaluate the effect of placing each *subset* of the item set on the right-hand side, leaving the remainder of the set as the antecedent.

We observed when describing association rules in Section 3.4 that if the double-consequent rule

If windy = false and play = no then outlook = sunny and humidity = high holds with a given minimum coverage and accuracy, then both single consequent rules formed from the same item set must also hold:

If humidity = high and windy = false and play = no then outlook = sunny

If outlook = sunny and windy = false and play = no then humidity = high

Conversely, if one or other of the single-consequent rules does not hold, there is no point in considering the double-consequent one. This gives a way of building up from single-consequent rules to candidate double-consequent ones, from double-consequent rules to candidate triple-consequent ones, and so on. Of course, each candidate rule must be checked against the hash table to see if it really does have more than the specified minimum accuracy. But this generally involves checking far fewer rules than the brute force method. It is interesting that this way of building up candidate (n + 1)-consequent rules from actual n consequent ones is really just the same as building up candidate (n + 1)-item sets from actual n-item sets, described earlier.

4. WORKING EXAMPLE

Association analysis. Suppose, as a marketing manager of All Electronics, you would like to determine which items are frequently purchased together within the same transactions. An example of such a rule, mined from the All Electronics transactional database, is

buys(*X*; "*computer*"))*buys*(*X*; "*software*") [*support* = 1%; *confidence* = 50%]

where *X* is a variable representing a customer. A confidence, or certainty, of 50% means that if a customer buys a computer, there is a 50% chance that she will buy software as well. A 1% support means that 1% of all of the transactions under analysis showed that computer and software were purchased together. This association rule involves a single attribute or predicate (i.e., *buys*) that repeats. Association rules that contain a single predicate are referred to as single-dimensional association rules. Dropping the predicate notation, the above rule can be written simply as "*computer*) *software* [1%, 50%]".

Suppose, instead, that we are given the All Electronics relational database relating to purchases. A data mining system may find association rules like

age(*X*, "20:::29")^*income*(*X*, "20K:::29K"))*buys*(*X*, "*CD player*") [*support* = 2%, *confidence* = 60%]

The rule indicates that of the All Electronics customers under study, 2% are 20 to 29 years of age with an income of 20,000 to 29,000 and have purchased a CD player at All Electronics. There is a 60% probability that a customer in this age and income group will purchase a CD player. Note that this is an association between more than one attribute, or predicate (i.e., *age*, *income*, and *buys*). Adopting the terminology used in multidimensional databases, where each attribute is referred to as a dimension, the above rule can be referred to as a multidimensional association rule. Typically, association rules are discarded as uninteresting if they do not satisfy both a minimum support threshold and a minimum confidence threshold. Additional analysis can be performed to uncover interesting statistical correlations between associated attribute-value pairs.

5. CONCLUSION

Association rules are often sought for very large datasets, and efficient algorithms are highly valued. The method described previously makes one pass through the dataset for each different size of item set. Sometimes the dataset is too large to read in to main memory and must be kept on disk; then it may be worth reducing the number of passes by checking item sets of two consecutive sizes in one go. For example, once sets with two items have been generated, all sets of three items could be generated from

them before going through the instance set to count the actual number of items in the sets. More three-item sets than necessary would be considered, but the number of passes through the entire dataset would be reduced. In practice, the amount of computation needed to generate association rules depends critically on the minimum coverage specified. The accuracy has less influence because it does not affect the number of passes that we must make through the dataset. In many situations we will want to obtain a certain number of rules say 50 with the greatest possible coverage at a pre-specified minimum accuracy level. One way to do this is to begin by specifying the coverage to be rather high and to then successively reduce it, re-executing the entire rule-finding algorithm for each coverage value and repeating this until the desired number of rules has been generated. The tabular input format that we use throughout this book, and in particular a standard ARFF file based on it, is very inefficient for many association-rule problems. Association rules are often used when attributes are binary either present or absent and most of the attribute values associated with a given instance are absent.

REFERENCES

[1] Ian H. Witten, Eibe Frank, Department of Computer Science, University of Waikato "Data Mining Practical Machine Learning Tools and Techniques", Second Edition, Morgan Kaufmann Publishers is an Imprint of Elsevier.

[2] Alex Tze Hiang Sim, Maria Indrawan, Samar Zutshi, Member, IEEE, and Bala Srinivasan "Logic-Based Pattern Discovery" , IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 6, June 2010.