International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 5, May - 2013

Generating Effective Pattern From Skewed Data-Set

[1]Manish K. Gupta
M.Tech Scholar
AFSET
Faridabad, India.

[2]Rupak Kumar
Sr. Lecturer
AFSET
Faridabad, India.

[3]Nitin K. Singh
IFTM University
Moradabad, India

Abstract

Most of the existing algorithms rely mainly on the spatial position of data items, thus generating patterns which are least required. Thus these algorithms prove to be not very effective for the purpose of mining interest patterns. Such problems persist in the conventional algorithms because they do not have predefined knowledge of the affinity among the data items. In this work a solution to this problem is proposed which is entirely based on the strong affinity between the data items. An algorithm has been proposed which help in finding the optimized pattern which are interested patterns for extracting value defined by the user and uses this threshold value to perform the extraction of optimized pattern from the item sets.

Keywords:- Support, Candidate pattern, O- Conf, maximal_ Optimized_ Pattern, Data items.

1. Introduction

In Skewed dataset, means those datasets which consist of large number of data items, belonging to different levels of support, if we use conventional clustering algorithms for mining associated patterns then they will not be effective. Example of skewed dataset could be the large range of commodities in any shopping mall. These large ranges of commodities may belong to entirely different price level or may also belong to the same price level. Most of the conventional clustering algorithms rely entirely on support based pruning strategy and this strategy when used on highly skewed data as defined previously proves to be in effective because of the following two reasons.

1. If minimum threshold value is lowered, then the number of extracted patterns increases. Such spurious patterns contain data items belonging to different support level. These spurious patterns are called cross- support patterns and the data items contained in such spurious patterns are weakly correlated which each other. More ever, using a lower value for minimum threshold also increase the computational and memory requirement substantially.
2. If we use a large value for minimum threshold, just opposite of the first case, then there are high chance that interested patterns having support less than the minimum threshold value may be missed.

Such problem lead to the need for searching measure which perform mining by using substantially lower values of support level and able to remove spurious associations among items with substantially different support levels.

Generally the clustering algorithms do not have knowledge regarding the affinity between the data items. They simply follow their notion and consequently prune the patterns to form clusters having data items many of which having least affinity between them.

The proposed solution will be to have a measure that can efficiently identify useful patterns even at low levels of support and can be used to automatically remove spurious patterns during the association mining process. For achieving this goal cross-support property is being used for eliminating the spurious patterns which contains data items having substantially different levels.

2. Technique for maximal optimized patterns

As we already know that, clustering is a process of assigning various objects to various clusters, keeping in mind that objects or data items belonging to a certain cluster has higher similarity or higher affinity with each other. Thus, provides patterns for the purpose of discovering knowledge. If some of the data items belonging to that particular cluster moves from that particular cluster to some other cluster because of the clustering algorithm being used then it would become very difficult to gather knowledge from such clusters. Even more it is much easy to gather knowledge from the well understood patterns rather than interpreting the data items directly.

Thus to solve this problem optimized-clustering is an approach. In this approach, patterns are preserved such that the data items belonging to a particular pattern always belongs to a particular cluster. Optimized-patternsare patterns which contains data items which have high affinity with each other. By high affinity in optimized patterns means that the presence of each and every data item in that optimized-pattern highly implies the presence of each and every other data item belonging to that same optimized-pattern.

The optimized-confidence or (o-*confidence)* of an itemset I={$i_{1,}i_{2,}i_{3,.........,}i_m$}, is denoted as o-conf(I), is a measure that reflects the overall affinity among items within the itemset. This measure is defined as $min\{conf\{i_1 \rightarrow i_2,\ldots\ldots,i_m\}$, $conf\{i_2 \rightarrow i1,i_3,\ldots\ldots i_m\}$, ………., $conf\{i_m \rightarrow i_1,i_2,\ldots\ldots i_m-1\}\}$, where conf is the conventional definition of association rule confidence.

The scope of optimized-confidence could be understood properly with the help of following example. Consider an itemset I= {Bread, Butter, Milk}. Assume that supp({Bread})=0.1, supp({Butter})=0.1, supp({Milk})=0.06, and supp({Bread, Butter, Milk })=0.06, where supp is the support of an itemset. Then

conf{Bread$\rightarrow$Butter,Milk}=supp({Bread,Butter,Milk})/ supp({Bread})=0.6

conf{Butter$\rightarrow$Bread,milk}=supp({Bread,Butter,Milk})/ supp({Butter})=0.6

conf{Milk$\rightarrow$Bread,Butter}=supp({Bread,Butter,Milk})/ supp({Milk})=1

Hence, o-conf(I)=min{conf{Butter$\rightarrow$Bread,Milk}, conf{Bread$\rightarrow$Butter,Milk}, conf{Milk$\rightarrow$Bread,Butter}}=0.6.

The collection of candidate patterns from the itemset(I) is a optimized-pattern if and only if, the value of o-conf(I)>= Tc, Tc is the minimum threshold confidence which is provided by the user. Further if for any optimized-pattern there exist some subset of this optimized-pattern, then this subset pattern should be removed from the set of all optimized-pattern. The reason for this is due to the property of all-confidence **[10].**

2.1 Properties of o- confidence measure

The O-confidence measure has four important properties, namely the anti-monotone property, the cross-support property, the strong affinity property and the all-confidence property.

2.1.1 Anti-Monotone

The O-confidence measure posses anti-monotone property. This property states that if for all the data items belonging to P, the value of O-confidence is greater than the threshold value Tc, then for all the subsets of P, the value of O-confidence will remain greater than the threshold value Tc. How O-confidence measure uses this property of anti-monotone? This could be easily explained with the help of the following example: Suppose the supp({milk}) = 0.2, supp({sugar}) = 0.6 and the supp({milk,sugar})= 0.3 and the value of minimum o-confidence threshold is 0.6, then the o-confidence of the candidate pattern {milk,sugar} is given by supp({milk,sugar})/ max{supp({milk}),supp({sugar})}= 0.3/.6 = 0.5 which is less than the minimum o-confidence of 0.6. Thus the candidate pattern {milk,sugar} is not a optimized pattern. Moreover, all the candidate patterns having {milk,sugar} as their subset are pruned, like {milk,sugar,biscuit} is not a optimized pattern. One thing should be noted down here, the pruning here is done on the basis of O-confidence threshold. If the value of O-confidence threshold is reduced to .45, then {milk,sugar} will be a optimized pattern.

2.1.2 Strong Affinity

The O-confidence measure also posses the property of Strong Affinity. The O-confidence measure take cares that all the data items contained in a data set have strong affinity with each other. By strong affinity we means to say strong association between each other. This could be easily understood with the help of following consideration. Suppose the value of O-confidence is 90% for any itemset (D). Then if any of the data item belonging to the itemset (D) occurs in any transaction, then there are 90% chances that the remaining data items belonging to the same itemset (D) will also occur in the same transaction.

2.1.3 Cross-support patterns

The O-conf helps in minimizing the cross-support patterns which are actually the spurious patterns. It is always very difficult to choose the right threshold value for the purpose of mining the large collection of data. If we set a very high value of threshold then there are chances that we may miss many interesting patterns. Conversely, if we set a very low value for the threshold then also it may not be easy to find the interested associated patterns because of the following two reasons. The first reason is that the computational and memory requirements of existing analysis algorithm increases considerably and secondly, the number of extracted patterns also increases substantially. The O-conf helps us in eliminating patterns which consists of data items which are not of interest. Also, O-conf does not involve extra computational cost as it simply depends on the support values of the individual data items or their various combinations. This could be easily understood with the help of following consideration. Suppose Tc is the given value of threshold and P is a pattern such that P= {I1, I2,….,,In}. We could say P as a cross-support pattern with respect to Tc,if for any two data items suppose I1 and I2 belonging to P, the value of supp({I1})/supp({I2}) < Tc, where 0<Tc<1.

2.1.4 All Confidence

Omiecinski proposed the concept of all confidence**[10]** as an alternative to the support. All confidence represents the minimum confidence of all the association rules extracted from the itemset. Omiecinski's all-confidence posses the desirable property of anti-monotone.

The all-confidence measure for an itemset P = {i1, i2,…….., im} is given by min({conf(A→B | for all A, B is subset of P, AUB = P, A∩B = Ø }) and is equal to :-

$$\text{Supp}(\{i1,i2,\ldots\ldots,im\}) / \max_{1<=k<=m}\{\text{supp}(\{ik\})\} \qquad (5.1)$$

2.2 Algorithm for maximal optimized pattern

Input

I: *Item Set stored in database containing list of transactions with their items and corresponding support*

Min_threshold: *Minimum Threshold value of o-confidence*

*** Note the value of Min_threshold will be provided by the user.*

Variable

Optimized : *Optimized Pattern Set*

Maximal_Optimized: *Maximal Optimized Pattern Set*

Optimized_Pattern_Evaluation() : *Function for evaluating Optimized Pattern Set*

Maximal_ Optimized_Pattern_Evaluation() : *Function for evaluating Maximal Optimized Pattern Set*

Method

I: Extracting Maximal Optimized Pattern

```
Optimized= Optimized_Pattern_Evaluation(I, Min_threshold)
    {
        1. for each element in (I) access the support value
        2. Create candidate patterns with items belonging to different level of support
        3. Perform pruning based on Anti-monotone property
        4. Perform pruning based on cross-support property
        5. Optimized patterns (i.e. Optimized) with O-confidence > Min_threshold
    }
Maximal_Optimized=
Maximal_Optimized_Pattern_Evaluation(Optimized)
    {
```

```
    1. Find an optimized patterns (X') such
       that X' is a subset of Y and both X',Y
       Є Optimized
    2. Optimized = Optimized – X'
    3. Reapeat until steps 1 -2 until there
       exist no X', X' subset of Y and both
       belonging to the Optimized Patterns.
    4. Set Maximal_Optimized = Optimized
}
```

II: Performing Clustering

3. Explanation of the Algorithm

1. "I" provides the complete set of data items stored in the database on which the mining has to be performed. And "Min_threshold" defines the value of minimum threshold confidence as explained by the user.

2. "Optimized" and "Optimized_Pattern_Evaluation" is the variable for storing optimized pattern and the function for calculating the optimized patterns respectively. Similarly "Maximal_Optimized" and "Maximal_Optimized_Pattern_Evaluation" is the variable for storing maximal optimized pattern and the function for calculating the maximal optimized patterns respectively.

3.Optimized_Pattern_Evaluation() is a function used for calculating the Optimized patterns. It takes input item set and min_threshold value as the input parameter. And using them produces the optimized patterns with different level of support are created. After this, pruning is performed on these optimized patterns satisfying the property of Anti-Monotone and cross-support. These set of optimized patterns may consist of redundant patterns i.e. one pattern could be subset of other pattern. These patterns providing the redundant information need to be removed. These are removed by using another function Maximal_Optimized_Pattern_Evaluation(). This function takes optimized as its input parameter and thus helps in removing this problem.

4. Conclusion

This algorithm can even be used on highly skewed item set and proves to be beneficial by removing spurious patterns and generating strong affinity patterns. While on the other hand if we use conventional clustering algorithms for mining associated patterns then they will not be effective. Most of the clustering algorithms defined so far rely purely on support-based pruning strategy and this strategy when used on highly skewed data sets proves to be ineffective.

Reference

[1] J. Han, and M. Kamber, 2000. Data Mining Concepts and Techniques. Morgan Kanufmann.

[2] J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In: Proc. Conf. on the Management of Data (SIGMOD'00, Dallas, TX). ACM Press, New York, NY, USA 2000.

[3] J. Han, J. Pei, Y. Yin, and R. Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery, 2003.

[4] C. Borgelt. An Implementation of the FP-growth Algorithm. Proc. Workshop Open

Software for Data Mining (OSDM’05 at KDD’05, Chicago,IL),1–5.ACMPress, New York, NY, USA 2005.

[5] C. Borgelt. SaM: Simple Algorithms for Frequent Item Set Mining. IFSA/EUSFLAT 2009 conference- 2009.

[6] Berglund, E., Sitte, J.: The parameterless self-organizing map algorithm. IEEE Transactions on Neural Networks, vol. 17, no. 2, 2006, pp. 305-316.

[7] Van Hulle, M.: Faithful Representations and Topographic Maps: From Distortion- to Information-Based Self-Organization, John Wiley, New York, 2000.

[8] Kohonen, T.: Self-Organizing Maps. 3rd ed., Springer, Berlin, 2001.

[9] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys., vol. 21, no. 6, 1953, pp. 1087-1092.

[10] Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems, Annals of Eugenics, vol. 7, 1936, pp. 179188.