Volume 3, No. 10, October 2012
Journal of Global Research in Computer Science
RESEARCH PAPER
Available Online at www.jgrcs.info
ISSN-2229-371X

AN IMPROVED ASSOCIATION RULE MINING WITH FP TREE USING POSITIVE AND NEGATIVE INTEGRATION

Rashmi Shikhariya*1 and Prof. Nitin Shukla2
*1Computer Science & Engineering Shri Ram Institute of Technology
Jabalpur, India
rashmishikhariya86@gmail.com
2Computer Science & Engineering Shri Ram Institute of Technology
Jabalpur, India
nitnin.sritmca123@gmail.com

Abstract- Construction and development of classifier that work with more accuracy and perform efficiently for large database is one of the key task of data mining techniques [17] [18]. Training dataset repeatedly produces massive amount of rules. It's very tough to store, retrieve, prune, and sort a huge number of rules proficiently before applying to a classifier[1]. In such situation FP is the best choice but problem with this approach is that it generates redundant FP Tree. A Frequent Pattern Tree (FP-Tree) is a type of prefix tree [3] that allows the detection of recurrent (frequent) item set exclusive of the candidate item set generation [14]. It is anticipated to recuperate the flaw of existing mining methods. FP –Trees pursues the divide and conquers tactic. In this paper we have adopt the same idea of author [17] to deal with large database. For this we have integrated a positive and negative rule mining concept with frequent pattern (FP) of classification. Our method performs well and produces unique rules without ambiguity.

Keywords- Association, FP, FP-Tree, Nagtive, Positive.

INTRODUCTION

Mining using Association rules discover appealing links or relationship among the data items sets from huge amount of data [4]. For this, association uses various techniques like *Apriori* and *FP* rules. *Apriori* employ *generations-and-testing* methodology. While generating item sets, it examines the whole database of transactions multiple times to generate the frequent itemsets. This scanning process gradually decreases the frequent itemset generation rate as the data size grows [4].

Second well-known algorithm is *Frequent Pattern (FP) growth* algorithm. It takes up *divide-and-conquer* approach. *FP* computes the frequent items and forms a tree of frequent-pattern.

In comparison with *Apriori* algorithm, *FP* is much superior in case of efficiency [13]. But problem with traditional *FP* is that it produces a huge number of conditional *FP-Trees* [3].
Construction and development of classifier that work with more accuracy and perform efficiently for large database is one of the key task of data mining techniques [17] [18]. Training dataset repeatedly produces massive amount of rules. It's very tough to store, retrieve, prune and sort a huge number of rules proficiently before applying to a classifier [1]. To eliminate such problems, Author of [17] proposed a new method based on positive and negative concept of association rule mining. Author suggests customary techniques of classification based on the positive and negative association rules and ignores the value of negative association rules.

In this paper, we have adopted the same idea of Author [17] to deal with large database. For this, we have integrated a positive and negative rule mining concept with frequent pattern (*FP*) of classification. Our method performs well and produces unique rules without ambiguity.
Rest of paper is organized as follows:
Section 2 insights the background details of the association data mining technique and also explores the idea of *FP* and positive and negative theory.
Section 3 discusses the previous works in same field.
Section 4 discusses about the proposed method and algorithm adopted.
Section 5 presents the results obtained by the proposed method and finally
Section 6 concludes the paper.

BACKGROUNDS & RELATED TERMINOLOGY

Association

Association rule was proposed by Rakesh Agrawal [1]; its uses the 'if-then' rules to generate extracted information into the form transaction statements [3]. Such rules have been created from the dataset and it obtains with the help of support and confidence of rule that illustrate the rate (frequency) of occurrence of a given rule.

According to the Author of [2] Association mining may be can he stated as follows: Let $I = (i_1, i_2 \ldots i_n)$ be a set of items. Let $D = (T_1, T_2 \ldots T_j, \ldots T_m)$ the task-relevant data, be a set of transactions in a database, where each transaction $T_j (j=1, 2, \cdots, m)$ such that $T_j \subseteq I$. Each transaction is assigned an identifier, called TID (Transaction id). Let A be a set of items, a transaction T is said to contain A if and only if $A \subseteq I$. An association rule is an implication of the form $A \rightarrow B$ where $A \subseteq I$, $B \subseteq I$ and $A \cap B = \emptyset$. The rule $A \rightarrow B$ holds in the transaction set D with support s, where s is the percentage of transactions in D that contain $A \cup B$ (i.e., both A and B). This is taken to be the probability $P(A \cup B)$. The rule has confidence c in the transaction set D if c is the percentage of transactions in D containing A that also contain B. This is taken to be the conditional probability, P(B|A). That is,

$$\text{confidence}(A \rightarrow B) = P(B|A) =$$

$$\text{support}(A \cup B) \ / \ \text{support}(A) = c,$$

$$\text{support}(A \rightarrow B) = P(A \cup B) = s.$$

The popular association rules Mining is to mine strong association rules that satisfy the user specified both minimum support threshold and confidence threshold. That is, minconfidence and minsupport.

If support(X) ≥ minsupport, X is frequent item sets. Frequent k-itemsets is always marked as L_K.

If support (A→B) ≥ minsupport and confidence (A→B) ≥ minconfidence , A→B is strong correlation.

Several Theorems are introduced as follows:

(i) If A⊆B , support(A) ≥ support(B).

(ii) If A⊆B and A is non-frequent itemset, then B is non-frequent itemset.

(iii)If A⊆B and B is frequent itemset, then A is frequent itemset.

Frequent Pattern (FP) Tree:

A Frequent pattern tree (*FP-Tree*) is a type of prefix tree [3] that allows the detection of recurrent (frequent) item set exclusive of the candidate item set generation [14]. It is anticipated to recuperate the flaw of existing mining methods. *FP-Trees* pursue the divide and conquers tactic. The root of the *FP-Tree* is tag as 'NULL' value. Childs of the roots are the set of item of data. Conventionally a *FP-Tree* contains three fields- Item name, node link and count.

To avoid numerous conditional *FP-Trees* during mining of data author of [3] has proposed a new association rule mining technique using improved frequent pattern tree (*FP-Tree*) using table concept conjunction with a mining frequent item set (*MFI*) method to eliminate the redundant conditional *FP-Tree*.

Positive and Negative FP Rule Mining:

Author of [15] cleverly explain the concept of positive and negative association rules. According to the [15] two indicators are used to decide the positive and negative of the measure:

i. Firstly find out the correlation according to the value of

$$corr(P,Q) = \sup(P \cup Q) \,/\, \sup(P)\sup(Q)$$

which is used to delete the contradictory association rules emerged in mining process.

There are three measurements possible of *corr(P,Q)* [16]:

a. If $corr(P,Q) > 1$, Then *P* and *Q* are related;
b. If $corr(P,Q) = 1$, Then *P* and *Q* are independent of each other;
c. If $corr(P,Q) < 1$, Then *P* and *Q* negative correlation;

ii. Support and confidence is the positive and negative association rules in two important indicators of the measure.

The support given by the user to meet the minimum support (minsupport) a collection of item sets called frequent item sets, association rules mining to find frequent item sets is concentrating on the needs of the user to set the minimum confidence level (minconf) association rules.

LITERATURE SURVEY

Data mining is used to deal with size of data stored in the database, to extract the desired information and knowledge [3]. Data mining has various techniques to perform data extraction; association technique is the most effective data mining technique among them. It discover hidden or desired pattern among the large amount of data. It is responsible to find correlation relationships among different data attributes in a large set of items in a database. Since its introduction, this method has gained a lot of attention. Author of [3] has analyzed that an association analysis [1] [5] [6] [7] is the discovery of hidden pattern or clause that occur repeatedly mutually in a supplied data set. Association rule finds relations and connection among data and data sets given.

An association rule [1] [5] [8] [9] is a law which necessitate certain relationship with the objects or items. Such association's rules are calculated from the data with help of the concept of probability.

Association mining using *Apriori* algorithm perform better but in case of large database it performs slow because it has to scan the full database each time while scanning the transaction as author of [4] surveyed.

Apriori [6] employ a bottom-up breadth-first approach to discover the huge item set. The problem with this algorithm is that it cannot be applied directly to mine complex data [3]. Second well-known algorithm is *Frequent Pattern (FP) growth* algorithm. It takes up divide-and-conquer approach. *FP* computes the frequent items and forms in a tree of frequent-pattern.

In comparison with *Apriori* algorithm *FP* is much superior in case of efficiency [13]. But problem with traditional *FP* is that it produces a huge number of conditional *FP-Trees* [3].

IMPROVED ASSOCIATION RULE MINING WITH POSITIVE AND NEGATIVE INTEGRATION

Existing work based on *Apriori* algorithm uses candidate sets for finding frequent pattern to generate association rules, then apply class label association rules where this work uses *FP-Tree* with *growth* for finding frequent pattern to generate association rules. *Apriori* algorithm takes more time for large data set where *FP growth* is time efficient to find frequent pattern in transaction.

In this paper, we have proposed a new dimension into the data mining technique. For this we have integrated the concept of positive and negative association rules into the frequent pattern (*FP*) method. Negative and positive rules works better than traditional association rule mining and *FP* cleverly works in large database. Our proposed algorithm has two stages:

a. Rule Generation and,
b. Classification.

In the first stage, the algorithm calculates the whole set of positive and negative class association rules such that sup(R) support and conf(R) confidence given thresholds. Furthermore, the algorithm prunes some contradictory rules and only selects a subset of high quality rules for classification.

In the second stage i.e. classification, for a given data object, the algorithm extracts a subset of rules found in the first stage matching the data object and predicts the class label of the data object by analyzing this subset of rules.

Rule Generation:

To find rules for classification, the algorithm first mines the training dataset to find the complete set of rules passing certain support and confidence thresholds. This is a typical frequent pattern or association rule mining task. The algorithm adopts *FP Growth* method to find frequent itemset. *FP Growth* method is a frequent itemset mining algorithm which is fast. The algorithm also uses the correlation between itemsets to find positive and negative class association rules. The correlation between itemsets can be defined as:

$$Corr(X,Y) = \sup(X \cup Y) / \sup(X)\sup(Y)$$

Where X and *Y* are itemsets.

When *corr(X, Y)>1, X* and *Y* have positive correlation.
When *corr(X, Y)=1, X* and *Y* are independent.
When *corr(X, Y)<1, X* and *Y* have negative correlation.
Also when *corr(X, Y)>1*, we can deduce that *corr(X, -Y)<1* and *corr(-X,Y)<1.*

So, we can use the correlation between itemset *X* and class label c_i to judge the class association rules.

When *corr(X, ci)> 1*, we can deduce that there exists the positive class association rule $X \rightarrow ci$

When *corr(X, ci)> 1*, we can deduce that there exists the negative class association rule $X \rightarrow -ci$

The Rule Generation algorithm works as follow:

Definition FP-tree: A frequent-pattern tree (or FP-tree) is a tree structure defined below:

a. It consists of one root labeled as "null", a set of item-prefix subtrees as the children of the root, and a frequent-item-header table.
b. Each node in the item-prefix subtree consists of three fields: item-name, count, and node-link, where item-name registers which item this node represents, count registers the number of transactions represented by the portion of the path reaching this node, and node-link links to the next node in the FP-tree carrying the same item-name, or null if there is none.
c. Each entry in the frequent-item-header table consists of two fields - (1) item-name and (2) head of node-link (a pointer pointing to the first node in the FP-tree carrying the item-name).

Based on this definition, we have the following FP-tree construction algorithm.

Algorithm for FP-tree construction:

Input: A transaction database DB and a minimum support threshold ξ.
Output: FP-tree, the frequent-pattern tree of DB.
Method: The FP-tree is constructed as follows:

a. Scan the transaction database DB once. Collect F, the set of frequent items, and the support of each frequent item. Sort F in support-descending order as FList, the list of frequent items.
b. Create the root of an *FP-Tree*, T, and label it as "null". For each transaction Trans in DB do the following-

(a) Select the frequent items in Trans and sort them according to the order of FList. Let the sorted frequent-item list in Trans be [p | P], where p is the first element and P is the remaining list. Call insert tree ([p | P], T).
(b) The function insert tree([p | P], T) is performed as follows. If T has a child N such that N.item-name = p.item-name, then increment N's count by 1; else create a new node N, with its count initialized to 1, its parent link linked to T , and its node-link linked to the nodes with the same item-name via the node-link structure. If P is nonempty, call insert tree (P, N) recursively.

Algorithm to find Frequent Item sets using FP-Growth algorithm:

The *FP- Growth* algorithm for mining frequent patterns using *FP-Tree* by pattern fragment growth is:
Input: a *FP-Tree* constructed with the algorithm mentioned in Algorithm for FP-tree construction
D - Transaction database
ξ - Minimum support Threshold.
Output: The complete set of frequent patterns.
Method:
Call FP-growth (*FP-tree*, *null*)
Procedure FP-growth (*Tree, A*)
{
if (*Tree* contains a single path *P)*
then for each **(**combination (denoted as *B*) of the nodes in the path *P*) *do*
generate pattern $B \cup A$ with support = minimum support of nodes in *B*;
else (for each a_i in the header of *Tree*) do
{
generate pattern $B = a_i \cup A$ with *support* = a_i*.support*;
construct *B*'s conditional pattern base and then *B*'s conditional *FP-Tree TreeB*;
if ($TreeB \neq \emptyset$)
{
call *FP-growth* (*TreeB, B*)
}
}
}

The next step is to generate positive and negative class association rules. It firstly finds the rules contained in F which satisfy min_sup and min_conf threshold. Then, it will determined the rules whether belong to the set of positve class correlation rules P_AR or the set of negative class correlation rules N_AR.

The algorithm of generating positive and negative class association rules is shown as follow:

Algorithm for generating positive and negative class association rules:

Input: training dataset T, min_sup, min_conf
Output: P_AR, N_AR
Method:

```
a.  P_AR=NULL, N_AR=NULL;
b.  for (any frequent itemset X in F and ci in C)
      {
    if (sup(X→ci)>min_sup and conf(X→ ci)> min_conf)
            if corr(X, ci > 1)
            {
                P_AR= P_AR U {X→ ci};
            }
            else if corr(X, ci <1)
            {
                N_AR= N_AR U {X→ - ci};
            }
      }
c.  return P_AR and N_AR;
```

In this algorithm, correlation between itemsets and class labels is used as an important criterion to judge whether or not the correlation rule is positive. Lastly, P_AR and N_AR are returned.

Classification:

After P_AR and N_AR are selected for classification, the algorithm is ready to classify new objects. Given a new data object, the algorithm collects the subset of rules matching the new object.

In this section, we discuss how to determine the class label based on the subset of rules.

First, the algorithm finds all the rules matching the new object, generates PL set which includes all the positive rules from P _ AR and sorts the itemset by descending support values. The algorithm also generates NL set which includes all the negative rules from N_AR and sort the itemset by descending support values.

Second, the algorithm will compare the positive rules in PL with the negative rules in NL and decides the class label of the data object.

The algorithm of classification is shown as follow:

Algorithm for classification:

Input: data object, P_AR, N_AR
Output: the class label of data object C_d
Method:

```
a.  PL=Sort(P_AR); NL=Sort(N_AR); i=j=1;
b.  p_rule=GetElem(PL, i); n_rule=GetElem(NL, j);
c.  while(i≤PL_Length and j≤NL_Length
        {
            if(RuleCompare(p_rule, n_rule))
            {
                if(p_rule>n_rule)
                {
                    Cd = the label of p_rule;
                    Break;
                }
                if(p_rule=n_rule)
                {
                    Cd = the label of p_rule;
                    break;
                }
                if(p_rule<n_rule)
                {
                    j++;
                }
            }
            if(!RuleCompare(p_rule, n_rule))
            {
                if(p_rule>n_rule)
                {
                    Cd = the label of p_rule;
                    break;
                }
                if(p_rule=n_rule)
                {
                    i++;
                    j++;
                }
                if(p_rule<n_rule)
                {
                    i++;
                }
            }
        }
d.  Return Cd;
```

In the algorithm of classification, the function Sort(P_AR) returns PL and the itemsets in PL are sorted by descending support values, the function GetElem(PL, i) returns first I rule in the set of PL. Also, we can deduce the returns of the function of Sort(N_AR) and GetElem(NL, j).

RESULTS AND PERFORMANCE MEASUREMENT

Proposed enhanced FP with positive and negative system has been implement using java technologies.
Following results have been measured by the system.

Settings:

```
File name                 = data.num
Support (default 20%)     = 20.0
Confidence (default 80%) = 80.0
Reading input file: data.num
Number of records = 95
Number of columns = 38
Min support       = 19.0 (records)
Generation time = 0.0 seconds (0.0 mins)
FP tree storage = 2192 (bytes)
FP tree updates = 694
FP tree nodes   = 97
```

FP Tree:

```
(1) 9:90 (ref to null)
(1.1.1.1.1) 1:72 (ref to 1:4)
(1.1.1.1.1.1) 32:65 (ref to 32:3)
And so on…………
```

Generating ARs:

Generation time = 0.17 seconds (0.0 mins)
T-tree Storage = 8824 (Bytes)
Number of frequent sets = 626

```
[1] {9} = 90
[2] {19} = 90
[3] {19 9} = 85
[4] {23} = 90
And so on…….(Approximate 624 generated)
```

Association Rules:

```
(1) {1 32 5} -> {19} 100.0%
(2) {9 1 32 5} -> {19} 100.0%
.
.
.
(102) {9 23 32 14 37} -> {27} 100.0%
(103) {9 27 32 14 37} -> {23} 100.0%
(104) {9 32 14 37} -> {23 27} 100.0%
And so on……( Approximate 7855 generated)
```

Possitive Class Itemsets Rules:

```
{9 27 1 32 14} -> {19}
{9 27 1 32 14} -> {23}
{19 27 1 32 14} -> {23}
{9 19 27 1 32 14} -> {23}
{19 23 27 1 32 14} -> {9}
And so on…….
```

Negative Class Itemsets Rules:

```
{9 14 37} -> ~ {23 27}
{9 19 23 14 37} -> ~ {27}
{9 19 14 37} -> ~ {23 27}
{9 1 14 37} -> ~ {23}
{9 19 1 14 37} -> ~ {23}
And so on……
```

The result shows that the proposed system works more efficiently than exiting techniques. We have evaluated that it handles very large data set and is able to mine efficiently. A current experiment shows that it can handle data 129941 KB of data. This statistics is chosen by us.

CONCLUSION

In this paper, we have proposed a new hybrid approach for data mining process. Data mining is the current focus of research since last decade due to enormous amount of data and information in modern day. Association is the hot topic among various data mining technique. In this article we have proposed a hybrid approach to deal with large size data. Proposed system is the enhancement of *Frequent pattern (FP)* technique of association with positive and negative integration on it. *Traditional FP* method performs well but generates redundant trees resulting efficiency degrades. To achieve better efficiency in association mining, positive and negative rules generation helps out. Same concept has been applied in the proposed method. Results shows that proposed method perform well and handles very large size of data set.

REFERENCES

[1] Agrawal R,Imielinski T,Swami A, "Mining Association Rules between Sets of Items in Large Databases, "In:Proc of the ACM SIGMOD International conference on Management of Data,Washington DC,1993,pp,207-216.

[2] QI Zhenyu, XU Jing, GONG Dawei and TIAN He "Traversing Model Design Based on Strong-association Rule for Web Application Vulnerability Detection", IEEE, International Conference on Computer Engineering and Technology, 2009.

[3] A.B.M.Rezbaul Islam and Tae-Sun Chung "An Improved Frequent Pattern Tree Based Association Rule Mining Technique", IEEE, International Conference on Information Science and Applications (ICISA), 2011

[4] LUO XianWen and WANG WeiQing "Improved Algorithms Research for Association Rule Based on Matrix", IEEE, International Conference on Intelligent Computing and Cognitive Informatics, 2010.

[5] R Srikant, Qouc Vu and R Agrawal "Mining Association Rules with Item Constrains". IBM Research Centre, San Jose, CA 95120, USA.

[6] R Agrawal and R Srikant "Fast Algorithm for Mining Association Rules". Proceedings of VLDB conference pp 487 – 449, Santigo, Chile, 1994.

[7] J. Han and M. Kamber.Data Mining: Concepts and Techniques. Morgan Kaufman, San Francisco, CA, 2001.

[8] Ashok Savasere, E. Omiecinski and ShamkantNavathe"An Efficient Algorithm for Mining Association Rules in Large Databases", Proceedings of the 21st VLDB conference Zurich, Swizerland, 1995.

[9] Arun K Pujai "Data Mining techniques", UniversityPress (India) Pvt. Ltd., 2001.

[10] J. S. Park, M.-S. Chen and P. S. Yu, "An effective Hash-Based Algorithm for Mining Association Rules", Proceedings of the ACM SIGMOD, San Jose, CA, May1995, pp. 175-186.

[11] S. Brin, R. Motwani, J. D. Ullman and S. Tsur, "Dynamic Item set Counting and Implication Rules for Market Basket Data", Proceedings of the ACM SIGMOD, Tucson, AZ, May 1997, pp. 255-264.

[12] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation", Proceedings of the ACM SIGMOD, Dallas, TX, May 2000, pp. 1-12.

[13] Qin Ding and gnanasekaran Sundaraj "Association rule mining from XML data", Proceedings of the conference on data mining.DMIN'06.

[14] C. Silverstein, S. Brin, and R. Motwani, "Beyond Market Baskets: Generalizing Association Rules to Dependence Rules," Data Mining and Knowledge Discovery, 2(1), 1998, pp 39–68.

[15] Yanguang Shen, Jie Liu and Jing Shen "The Further Development of Weka Base on Positive and Negative Association Rules", IEEE, International Conference on Intelligent Computation Technology and Automation (ICICTA), 2010.

[16] Yanguang Shen,Jie Liu,Fangping Li.Application Research on Positive and Negative Association Rules Oriented Software Defects,2009 International Conference on Computational Intelligence and Software Engineering(CISE 2009)[C].Wuhan,China,December 11-13,2009.

[17] LuoJunwei and Luo Huimin "Algorithm for Classification Based on Positive and Negative Class Association Rules", 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), 2010.

[18] Wenmin Li, Jiawei Han and Jian Pei, "CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules," Proceedings of the 2001 IEEE International Conference on Data Mining, IEEE Press, Dec. 2001, pp. 123-131.

