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 frequent pattern rules, even though Apriori employ cut-technology while generating item sets, it examine the whole database during scanning of the the transaction database every time. This resulting scanning speed is gradually decreased as the data size is growing [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 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].

Construction and development of classifier that works with more accuracy and perform efficiently for large database is one of the key task of data mining techniques [17] [18]. Secondly 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 integrate a positive and negative rule mining concept with frequent pattern (FP) of classification. Our method performs well and produces unique rules without ambiguity.

Backgrounds & Related terminology

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 a piece rule that illustrate the rate (frequency) of occurrence of a given rule.

According to the Author of [2] Association mining may be can be he stated as follows: Let \( I = (i_1, i_2, \ldots, i_n) \) be a set of items. Let \( D = (T_1, T_2, \ldots, T_m) \) be a set of transactions in a database, where each transaction \( T_j \) is a set of items, a transaction \( T \) is said to contain \( A \) if \( A \subseteq T \) 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 \( A \) and \( B \) (i.e., both A and B). This is taken to be the probability \( P(A \rightarrow B) \). 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, \( P(B|A) \).

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 \( support(X) \geq minsupport \) and \( support(A \rightarrow B) \geq minconfidence \) then \( A \rightarrow B \) is strong correlation. Several Theorems are introduced as follows:

(i) If \( A \subseteq B \), then \( support(A) \leq support(B) \).

(ii) If \( A \not\subseteq B \) and \( A \) is non-frequent itemset then \( B \) is non-frequent itemset.

Keyword

Association, FP, FP-Tree, Negative, Positive.
(iii) If \( A \subseteq B \) 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-Tree pursues the divide and conquer tactic. The root of the FP-tree is tagged 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:

1) Firstly find out the correlation according to the value of \( \text{corrP, Q}=s(P)\cdot s(Q) \), which is used to delete the contradictory association rules emerged in mining process. There are three measurements possible of \( \text{corrP, Q} \) [16]:
   - If \( \text{corrP}, \text{Q}>1 \), Then P and Q are related;
   - If \( \text{corrP}, \text{Q}=1 \), Then P and Q are independent of each other;
   - If \( \text{corrP}, \text{Q}<1 \), Then P and Q negative correlation;

2) 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 items called frequent itemsets, association rules mining to find frequent itemsets is concentrating on the needs of the user to set the minimum confidence level (minconf) association rules.

Negative association rules contains itemset does not exist (non-existing-items, for example \( \overline{P}, \overline{Q} \)), Direct calculation of their support and confidence level more difficult.

**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 technique 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 hidden pattern or clause that occur repeatedly mutually in a 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.

Author of [3] has surveyed and conclude with the help of previous research in data mining using association rules has found that all the previously proposed algorithm like - Apriori [10], DHP [11], and FP growth [12].

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].

**Improving association rule mining with correlation technique**

Existing work based on Apriori algorithm 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 propose 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. Positive and negative rules works better for than traditional association rule mining and FP cleverly works in large database. Our proposed method work as follows-

**Positive and Negative class association rules based on fp tree**

This algorithm has two stages: rule generation and classification. In the first stage: the algorithm calculate the whole set of positive and negative class association rules such that \( \text{sup}(R) \) support and \( \text{conf}(R) \) 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: classification, for a given data object, the algorithm extracts a subset of rules fund in the first stage matching the data object and predicts the class label of the data object by analyzing this subset of rules.

**Generating Rules**

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:

\[
\text{corr}(X, Y) = \frac{\text{sup}(X \cup Y)}{\text{sup}(X) \cdot \text{sup}(Y)}
\]

Among X and Y are itemsets.

When \( \text{corr}(X, Y)>1 \), X and Y have positive correlation.

When \( \text{corr}(X, Y)=1 \), X and Y are independent.

When \( \text{corr}(X, Y)<1 \), X and Y have negative correlation.

Also when \( \text{corr}(X, Y)>1 \), we can deduce that \( \text{corr}(X, -Y)<1 \) and \( \text{corr}(-X, Y)<1 \).

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

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

When \( \text{corr}(X, ci)> 1 \), we can deduce that there exists the negative class association rule \( X \rightarrow \neg ci \).

So, the first step is to generate all the frequent itemsets by making multiple passes over the data. In the first pass, it counts the support of individual itemsets and determines whether it is frequent. In each subsequent pass, it starts with the seed set of itemsets found to be frequent in the previous pass. It uses this seed set to generate new possibly frequent itemsets, called candidate itemsets. The actual supports for these candidate
The algorithm of generating frequent itemsets is shown as follow:

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

- 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.
- 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.
- 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 \( \xi \).

**Output**: FP-tree, the frequent-pattern tree of DB.

**Method**: The FP-tree is constructed as follows.

1. 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.
2. Create the root of an FP-tree, T, and label it as “null”. For each transaction Trans in DB do the following:
   - Select the frequent items in Trans and sort them according to the order of FList. Let the sorted frequent-item list in Trans be \([p \mid P]\), where \( p \) is the first element and \( P \) is the remaining list. Call insert tree \([p \mid P], T\).
     - The function insert tree \([p \mid P], T\) is performed as follows.
       - If \( T \) has a child \( N \) such that \( N . i t e m - n a m e = p . i t e m - n a m e \), then increment \( N \)'s count by 1; else create a new node \( N \) with its item-name via the node-link structure. If \( P \) is nonempty, call insert tree \((P, N)\) recursively.
   - Then, the next step is to generate positive and negative class association rules. It firstly finds the rules contained in \( F \) which satisfy \( \text{min}_\text{sup} \) and \( \text{min}_\text{conf} \) threshold. Then, it will determined the rules whether belong to the set of positive class correlation rules \( P \_ \text{AR} \) or the set of negative class correlation rules \( N \_ \text{AR} \).

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 \( T \), \( \text{min}_\text{sup} \), \( \text{min}_\text{conf} \)

**Output**: \( P \_ \text{AR}, N \_ \text{AR} \)

(1) \( P \_ \text{AR}=\text{NULL}; N \_ \text{AR}=\text{NULL}; \)

(2) for (any frequent itemset \( X \) in \( F \) and \( Ci \) in \( C \))
   
   if \( \{ \text{sup}(X \rightarrow ci) > \text{min}_\text{sup} \text{ and conf}(X \rightarrow ci) > \text{min}_\text{conf} \} \)
   
   if \( \text{corr}(X, ci > 1) \)
   
   \( P \_ \text{AR}= P \_ \text{AR} \cup \{X \rightarrow ci;\} \)
   
   else if \( \text{corr}(X, ci < 1) \)
   
   \( N \_ \text{AR}= N \_ \text{AR} \cup \{X \rightarrow - ci;\} \)

(3) return \( P \_ \text{AR} \) and \( N \_ \text{AR} \);

In this algorithm, we use FP Growth method generates the set of frequent itemsets \( F \). In \( F \), there are some itemsets passing certain support and confidence thresholds. And the correlation between itemsets and class labels is used as an important criterion to judge whether or not the correlation rule is positive. Lastly, \( P \_ \text{AR} \) and \( N \_ \text{AR} \) are returned.

**Classification**

After \( P \_ \text{AR} \) and \( N \_ \text{AR} \) 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 \( P \_ \text{AR} \) and sorts the itemset by descending support values. The algorithm also generates \( NL \) set which includes all the negative rules from \( N \_ \text{AR} \) 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, \( P \_ \text{AR}, N \_ \text{AR} \)

**Output**: the class label of data object \( Cd \)

(1) \( PL=\text{Sort}(P \_ \text{AR}); NL=\text{Sort}(N \_ \text{AR}); i=j=1; \)

(2) \( pJule=\text{GetElem}(PL, i); nJule=\text{GetElem}(NL, j); \)

(3) while \( Ci <= PL\_ \text{Length} \text{ and } j <= NL\_ \text{Length} \)
   
   if (RuleCompare\((p\_role, n\_role)\) \}
   
   if (\( p\_role > n\_role \)) \}
   
   \( Cd = \text{the label of } p\_role; \)
   
   if (\( p\_role = n\_role \)) \}
   
   \( Cd = \text{the label of } n\_role; \)
   
   if (\( p\_role < n\_role \)) \}
   
   \( Cd = \text{the label of } n\_role; \)
   
   if (RuleCompare\((p\_role, n\_role)\) \}
   
   if (\( p\_role > n\_role \) and \( ni \)) \}
   
   \( Cd = \text{the label of } p\_role; \)
   
   \( Cd = \text{the label of } n\_rule; \)
   
   \( Cd = \text{the label of } n\_role; \)
   
   if (RuleCompare\((p\_rule, n\_rule)\) \}
   
   if (\( p\_rule > n\_rule \)) \}
   
   \( Cd = \text{the label of } p\_role; \)
   
   \( Cd = \text{the label of } n\_rule; \)
   
   \( Cd = \text{the label of } n\_role; \)
   
   if (RuleCompare\((p\_rule, n\_rule)\) \}
   
   if (\( p\_rule < n\_rule \)) \}
   
   \( Cd = \text{the label of } p\_role; \)
   
   \( Cd = \text{the label of } n\_rule; \)
   
   \( Cd = \text{the label of } n\_role; \)
   
   \( Cd = \text{the label of } n\_role; \)
   
(4) return \( Cd \);
In the algorithm of classification, the function \text{Sort}(P, \text{AR}) returns \text{PL} and the itemsets in \text{PL} are sorted by descending support values, the function \text{GetElem}(pL, i) returns first I rule in the set of P L. Also, we can deduce the returns of the function of \text{Sort}(\text{N,AR}) and \text{GetElem}(\text{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

<table>
<thead>
<tr>
<th>File name</th>
<th>data.num</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support</td>
<td>default 20% = 20.0</td>
</tr>
<tr>
<td>Confidence</td>
<td>default 80% = 80.0</td>
</tr>
<tr>
<td>Reading input file</td>
<td>data.num</td>
</tr>
<tr>
<td>Number of records</td>
<td>95</td>
</tr>
<tr>
<td>Number of columns</td>
<td>38</td>
</tr>
<tr>
<td>Min support</td>
<td>19.0 (records)</td>
</tr>
<tr>
<td>Generation time</td>
<td>0.0 seconds (0.0 mins)</td>
</tr>
<tr>
<td>FP tree storage</td>
<td>2192 (bytes)</td>
</tr>
<tr>
<td>FP tree updates</td>
<td>694</td>
</tr>
<tr>
<td>FP tree nodes</td>
<td>97</td>
</tr>
</tbody>
</table>

\text{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............

\text{Generating ars:}

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

\text{Approximate 624 generated}

\text{Association Rules}

(1) \{32,5\} \rightarrow \{19\} 100.0%
(2) \{9,1,32,5\} \rightarrow \{19\} 100.0%

\text{Approximate 7855 generated}

\text{Positive Class Itemsets RULES}

\{9,27,1,32,14\} \rightarrow \{19\}
\{9,27,1,32,14\} \rightarrow \{23\}
\{9,27,1,32,14\} \rightarrow \{27\}
\{9,19,27,1,32,14\} \rightarrow \{23\}
\{9,19,27,1,32,14\} \rightarrow \{9\}

\text{Negative Class Itemsets RULES}

\{9,14,37\} \rightarrow \sim \{23,27\}
\{9,19,23,14,37\} \rightarrow \sim \{27\}
\{9,19,14,37\} \rightarrow \sim \{23,27\}
\{9,19,14,37\} \rightarrow \sim \{23\}
\{9,19,14,37\} \rightarrow \sim \{27\}

The result shows that the proposed system works more efficiently than exiting positive and negative using Apriory technique. We have evaluated that it can handle very large data set and able to mine efficiently. A current experiment shows that it can handle data 129941 KB of data. This statistics is chosen by us. Even our system can handle and generate more mined data.

Conclusion

In this paper we have proposed a new hybrid approach to 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 that efficiency degrades. To achieve better efficiency in association mining positive and negative rules generation help out. Same concept has been applied in the proposed method. Results shows that proped method perform well and can handle very large size of data set.

References

[5] R Srikan R, Quoc Vu and R Agrawal “Mining Association Rules with Item Constrains”. IBM Research Centre, San Jose, CA 95120, USA.
[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.


