International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 18 (2015) pp 39126-39130

An Enhanced and Productive Technique for Privacy Preserving mining of Association rules from Horizontal Distributed Database

Sreevidya. B

Lecturer Department of Computer Science and Engineering,
Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Bangalore, India.
sreevidya. bkrishnan@gmail. com

Abstract:
The past two decades has seen a dramatic increase in the amount of information or data being stored in electronic format. This accumulation of data has taken place at an explosive rate. It has been estimated that the amount of information in the world doubles every 20 months and the size and number of databases are increasing even faster. The increase in use of electronic data gathering devices such as point-of-sale or remote sensing devices has contributed to this explosion of available data. Data confidentiality is a major concern in database systems especially when are huge amounts of data to be processed, so we try to implement a system where we could preserve security and maintain data confidentiality. Further there is a huge trend today towards distributed databases which make data mining very easy, reliable and efficient. In our paper we implement Fast Distributed Mining of Apriori algorithm for mining this huge transactional dataset. With the help of this algorithm we find the frequent item sets that are consumed in all transaction. Finding this frequent item sets in very important. By knowing this frequent item sets one canunderstand the interest of consumers and focus in profiting his business by mining association rules from them. In other words association rules can be used for decision making.

Keywords: Distributed mining, Apriori Algorithm, Frequent itemset

1. INTRODUCTION

Data storage became easier as the availability of large amounts of computing power at low cost ie the cost of processing power and storage is falling, made data cheap. There was also the introduction of new machine learning methods for knowledge representation based on logic programming etc. in addition to traditional statistical analysis of data. The new methods tend to be computationally intensive hence a demand for more processing power. Having concentrated so much attention on the accumulation of data the problem was what to do with this valuable resource? It was recognised that information is at the heart of business operations and that decision-makers could make use of the data stored to gain valuable insight into the business. Database Management systems gave access to the data stored but this was only a small part of what could be gained from the data. Traditional on-line transaction processing systems, OLTPs, are good at putting data into databases quickly, safely and efficiently but are not good at delivering meaningful analysis in return. Analysing data can provide further knowledge about a business by going beyond the data explicitly stored to derive knowledge about the business. This is where Data Mining[4] or Knowledge Discovery in Databases (KDD) has obvious benefits for any enterprise. The term data mining has been stretched beyond its limits to apply to any form of data analysis. Some of the numerous definitions of Data Mining, or Knowledge Discovery in Databases. The amount of data kept in computer files is growing at a phenomenal rate. It is estimated that the amount of data in the world is doubling every 20 months. At the same time, the users of these data are expecting more sophisticated information. Simple structured languages (like SQL) are not adequate to support the increasing demands for information. Data mining attempts to solve the problem.

Data mining potential can be enhanced if the appropriate data has been collected and stored in a data warehouse. A data warehouse is a relational database management system (RDMS) designed specifically to meet the needs of transaction processing systems. It can be loosely defined as any centralized data repository which can be queried for business benefit but this will be more clearly defined later. Data warehousing is a new powerful technique making it possible to extract archived operational data and overcome inconsistencies between different legacy data formats. As well as integrating data throughout an enterprise, regardless of location, format, or communication requirements it is possible to incorporate additional or expert information. It is, the logical link between what the managers see in their decision support EIS applications and the company's operational activities. In other words the data warehouse provides data that is already transformed and summarized, therefore making it an appropriate environment for more efficient DSS and EIS applications

The amount of data kept in computer files is growing at a phenomenal rate. The data mining field offers to discover unknown information. Data mining is often defined as the process of discovering meaningful, new correlation patterns and trends through non-trivial extraction of implicit, previously unknown information from the large amount of data stored in repositories, using pattern recognition as well as statistical and mathematical techniques. An SQL query is usually stated or written to retrieve specific data, while data miners might not even be exactly sure of what they require. So, the output of a SQL query is usually a subset of the database; whereas the output of a data mining query is an analysis of the contents of the database analysis task.

1. 1 Association Rule Mining

Association rule mining, [3] which was introduced by Agarwal et al. (1993), has become a popular research area due to its applicability in various fields such as market analysis, forecasting and fraud detection. Given a market basket dataset, association rule mining discovers all association rules such as "A customer who buys item *X*, also buys item *Y* at the same time". These rules are displayed in the form of *X*--->*Y* where *X* and *Y* are sets of items that belong to a transactional database. Support of association rule *X*---> *Y* is the percentage of transactions in the database that contain *XUY*. Association rule mining aims to discover interesting relationships and patterns among items in a database.

Association rule mining is a two steps process

- Finding the frequent item sets of items having count equal or greater than the minimum support count
- It discovers the association rules from these obtained frequent item sets.

First step of association rule mining algorithm, Apriori algorithm is proposed in 1994 to discover the frequent item sets. Discovered frequent item sets has the impact in decision making process

2. PROBLEM STATEMENT

2. 1 Existing System

There are many papers proposed since 1993

(i). Generating the Candidate set method, which was proposed in 1993 by Sreekanth and Agarwal [3].
(ii) Fast Distributed Mining
(iii). Secure Mining of Association Rules in Horizontally Distributed Databases

2.2 Proposed System

The proposed system includes the following:

(i). Propose a new enhanced Fast Distributed Mining (FDM) algorithm that improves the privacy of sensitive knowledge (as itemsets) by blocking more inference channels.
(ii). Propose two techniques (item count and increasing cardinality) based on item-restriction that hide sensitive itemsets (and we perform experiments to compare the two techniques).
(iii). Propose an efficient protocol that allows parties to share data in a private way with no restrictions and without loss of accuracy

3. LITERATURE SURVEY

3. 1Existing mining Technique for finding Frequent itemsets (FIS)

We referred to many existing techniques to find the frequent item sets. Following are some of the techniques which we described now.
Apriori algorithm with Candidate set generation approach
Fast Distributed Mining[1]

3. 1. 1 Apriori Algorithm with Candidate Generation

Apriori algorithm, uses prior knowledge of frequent item set properties and runs an iterative approach called level wise search to generate frequent item sets. That is, k-item sets are used to explore k+1 item sets[5]. Before generating the k^{th} frequent item set we generate the candidate sets and scan them against the data base and eliminate the item sets which are not satisfying the minimum support count. This process terminates when no frequent or candidate set can be generated.
This algorithm has two steps
First step is **Joint Step**, where we find Lk a set of candidate k item sets, it is generated by joining Lk-1 with itself. This set of candidates is denoted by Ck.
Second step is **Prune Step**, here we remove the members of Ck whose count is less than the minimum support count.
Inputs for this algorithm are
D = Transactional database, MinSupp = Minimum Support Count
Output given out by this algorithm is
L = Frequent Item Sets.

Limitations of this approach:
Scans the DB for more number of times
We should check if all the subsets of the item sets are present in previously generated frequent item sets.
Execution time and compilation times are high.

3. 1. 2 Fast Distributed Mining for Association Rules

Fast distributed mining (FDM) which is proposed by David W Cheung, used to find association rules in distributed databases. FDM generates a smaller number of candidate sets and substantially reduces the number of messages to be passed. FDM is recommended for distributed databases.
Inputs for this algorithm are
D = distributed databases
MinSupp = Minimum Support Count
MinConf= Minimum Confidence
Output given out by this algorithm is
L = Association rules
FDM has four steps

- Generation of Candidate Sets
- Local Pruning
- Global Pruning
- Count Polling

Table 1: Various symbols used in the algorithm

D	Number of transactions in database
S	Support threshold minsup
L(k)	Globally large k-itemsets
CA(k)	Candidate sets generated from L(k)
X. sup	Global support count of X
Di	Number of transactions in DBi
Gli(k)	gl-large k-itemsets at Si
CGi(k)	Candidate sets generated from Gli(k-1)
LLi(k)	Locally large k-itemsets in CGi(k)

(i). Generation of Candidate sets

It is important to observe some interesting properties related to large itemsets in distributed environments since such properties may substantially reduce the number of messages to be passed across network at mining association rules.

There is an important relationship between large itemsets and the sites in a distributed database: every globally large itemset must be locally large at some site(s). If an itemset X is both is both globally large and locally large at site Si, X is called gl-large itemsets at a site will form a basis for the site to generate its own candidate sets.

Lemma 1

If an itemset X is globally large, there exists a site Si, (1<i<n), such that X and all its subsets are gl-large at site Si.

Example 1:

Assume there are 3 sites in a system which partitions the DB into DB1, DB2 and DB3. Suppose the set of large 1-itemsets (computed at the first iteration)

L(1)={A, B, C, D, E, F, G, H}, in which A, B, C are locally large at site S1. B, C, D are locally large at site S2. E, F, G, H are locally large at site S3. Therefore, GL1(1)={A, B, C}, GL2(1)={B, C, D}, and GL3(1)={E, F, G, H}. Based on theorem, the set of size-2 candidate sets at site s1 is CG1(2), whereCG1(2)=Apriori-gen(GL1(1))={AB, BC, AC}, CG2(2)=Apriori-gen(GL2(1))={BC, CD, BD} andCG3(2)=Apriori-gen(GL3(1)) ={EF, EG, EH, FG, FH, GH}. Hence the set of candidate sets for large 2-itemsets is CG (2) =CG1 (2) U CG2 (2) U CG3 (2) =3+3+4=11. However, if the Apriori-gen is applied to L (1) then it would have 28 candidate sets. This shows that it is very effective to apply theorem to reduce the candidate sets.

(ii). Local Pruning of Candidate sets.

Local Pruning is a technique to prune away some candidate sets at each individual iteration.

When the set of candidate set CG(k) is generated, to find the globally large itemsets, the support counts of the candidate sets must be exchanged among all other sites. Notice that some candidate sets in CG(k) is generated to find globally large itemsets, the support counts of the candidate sets must be exchanged among all other sites. The general idea is that at each site Si if a candidate set X ∈CG(k) is not locally large at Si then there is no need for Si to find its global support count to find whether it is globally large. Therefore in order to compute all the large k itemsets at each site Si the candidate sets can be confined to only the sets X ∈CG(k) which are locally large at site Si.

(iii). Global Pruning of Candidate sets

The local pruning at site Si uses only the local support counts found in DBi to prune a candidate set. In fact, the local support counts from other sites can also be used for pruning. A global pruning technique is developed to facilitate such pruning and is outlined as follows. At the end of each iteration all the local support and global support counts of a candidate set X are available. These local support counts can be broadcasted together with global support counts after a candidate set is found to be globally large. Using this information, some global pruning can be performed on the candidate sets at the subsequent iteration.

Assume that the local support count of every candidate itemset is broadcasted to all the sites after it is found to be globally large at the end of an iteration. Suppose X is a size-k candidate itemset at the kth iteration. Therefore the local support counts of all the size (k-1) subsets of X are available at every site. With respect to a partition DBi, (1<i<n) we use maxsup(X) to denote the minimum value of the local support counts of all the size-(k-1) subsets. It follows from the subset relationship that maxsup(X) is an upper bound of the local support X. sup. Hence the sum of these upper bounds over all the partitions denoted maxsup(X), is an upper bound of X. sup. In other words, X. sup<maxsup(X) can be computed at every site at the beginning of the k-th iteration. Since maxsup(X) is an upper bound of its global support count, it can be used for pruning i. e., if maxsup(X) <s×D, then X cannot be a candidate set. This technique is called global pruning.

Global pruning can be combined with local pruning to from different pruning strategies.

(iv). Count Polling

In the algorithm the local support count of every candidate itemset is broadcasted from every site to every other site. Therefore the number of messages required for count exchange for each candidate sets is O(n2), where n is the number of partitions.

In our method, if a candidate itemset X is locally large at site Si, Si needs O(n) messages to collect all the support counts for X.

4. SYSTEM DESIGN

4. 1 Architecture Diagram

This section gives the complete design details pertaining to the paper. The purpose of this section is to focus on different modules of project and their interaction.

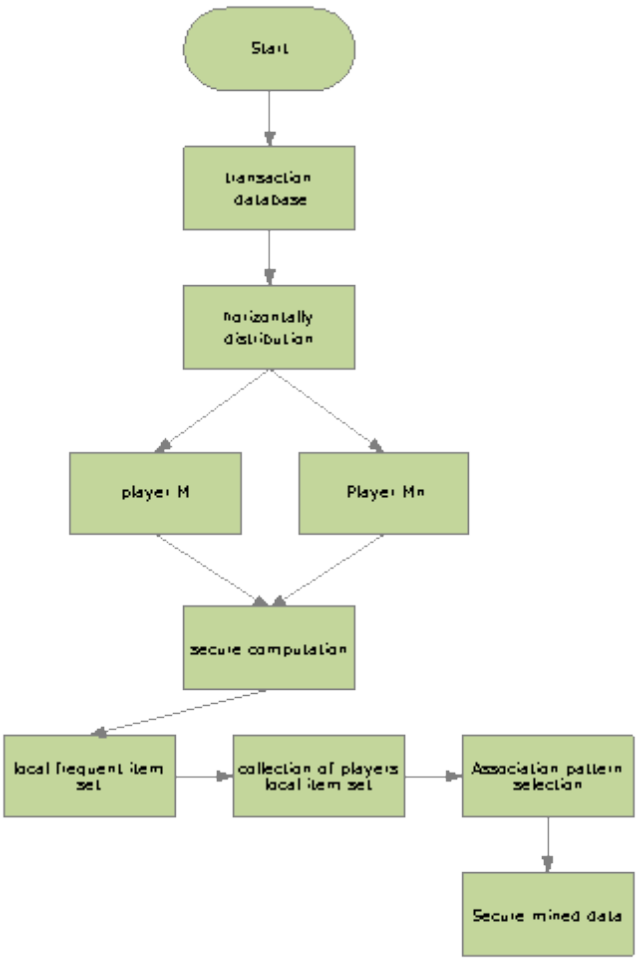

Fig 1: Architecture Diagram

5. PROPOSED ALGORITHM

5. 1Working Procedure

In our paper, we create a scenario where the algorithm accepts the transactional data sets from different distributed databases as inputs and gives association rules as output. To obtain better results we have used FDM algorithm for further efficiency and simplicity. This can be used in our regular life where we have various distributed databases that is databases that are not physically connected to each other. These distributed databases has become a trend in the recent present because of its efficiency, time to process data, simplicity, reliability easier expansion and a lot many things.

The main ingredients are two novel secure multi-party algorithms — one that computes the union of private subsets that each of the interacting players hold, and another that tests the inclusion of an element held by one player in a subset held by another. The goal here is to find association rules with support at least s and confidence level c, that hold in the unified database, while minimizing the information disclosed about the private databases held by those players.

The proposed system basically has 3 phases listed below

Input:

Each player PM has an input set $C^k, \ m \subseteq Ap(F^{k-1})$, $1 \le m \le M$

Output:

$C^k = \ \cup^M{}_{m=1} C^k, m$

Phase 0: Getting started

1. The players decide on a commutative cipher and each player PM, $1 \le m \le M$, selects a random secret encryption key KM.
2. The players select a hash function H and compute H(x) for all $x \in Ap(F^{k-1})$
3. Build a lookup table T= $\{(x, \ h(x)) : x \in Ap(F^{k-1})\}$

Phase 1: Encryption of all itemsets

For all players PM, $1 \le m \le M$, do
Set $X_m = \emptyset$
For all $x \in C^k, \ m$
Player Pm computes $EK_m(h(x))$ and adds it to X_m
End for
Player Pm adds to X_m faked itemsets until its size becomes $|Ap(F^{k-1})|$
End for
For i=2 to M do
For all $1 \le m \le M$ do
Pm sends a permutation of X_m to P_{m+1}
Pm receives from P_{m-1} the permuted X_{m-1}
Pm computes a new X_m as the encryption of the permuted X_{m-1} using the key K_m
end for
end for

Phase 2: Merging Itemsets

1. Each odd player sends his encrypted set to player P1
2. Each even player sends his encrypted set to player p2
3. P1 unifies all sets that were sent by the odd players and removes duplicates
4. P2 unifies all sets that were sent by the even players and removes duplicates
5. P2 sends his permuted list of itemsets to P1
6. P1 unifies his list of itemsets and the list received from P2 and then removes duplicates from the unified list. Denote the final list by $EC_s{}^k$

Phase 3: Decryption

1. For m=1 to M-1 do
2. PM decrypts all itemsets in EC_k using KM
3. PM sends the permuted(and Km decrypted) $EC_k{}^s$to P_{m+1}
4. End for
5. PM decrypts all itemsets in $EC_s{}^k$using K_M; denote the resulting set by C^k_s
6. PM uses the lookup table T to replace hashed values with the actual itemsets, and to identify and remove the faked itemsets
7. PM broadcasts C^k_s

6. CONCLUSION& FUTURE WORK

The proposed Fast Distributed Mining approach for finding association rules is very efficient and fast. Hence it greatly reduces the input and output cost. So improved algorithm decreases temporal complexity and special complexity and have higher efficiency and accuracy in output when compared to our classical Apriori algorithms. In this paper, we are working on transactional data sets. This work can also be extended to many other disciplines such as financial data analysis, market analysis that make use of transactional data sets. Our data mining techniques such as FDM can be used for analysis of collected data. Decisions can be taken by using this algorithm in real life scenarios.

REFERENCES

1. Tamir Tessa, "Secure Mining of Association Rules in Horizontally Distributed Databases. "
2. ShaliniDutt, Naveen Choudhary, DharmSingh, "An Improved Apriori Algorithm based on Matrix Data Structure", Double Blind Peer Reviewed International Research Journal, Volume 14 Issue 5 Version 1. 0 Year 2014
3. Agarwal, R. and Srikant, R. 1994. "Fast algorithms for mining association rules in large databases". Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994.
4. J Han, "Data Mining Concepts and Techniques" SecondEdition. Morgan Kaufmann Publisher, 2006, pp. 123-134.
5. Jaishree Singh, Hari Ram, Dr. J. S. Sodhi, "Improving Efficiency of Apriori Algorithm Using Transaction Reduction", International Journal of Scientific and Research Publications, Volume 3, Issue 1, January 2013