A Distributed Mining Of Association Rules On Horizontally Partitioned Data By Preservingprivacy.

Krishna Sowjanya.K[#1], R.Srinivas[*2]

Computer Sceince and Engineering, Jawaharlal Nehru Technological University Kakinada.
[1]Krishna Sowjanya.K, Student, M. Tech., Sri SaiAdityaInistitute of Science &Technology, Kakinada, India

[*]R.Srinivas, M.Tech, (Ph.D)
HOD_CSE.Dept, Sri SaiAdityaInistitute of Science &Technology, Kakinada, India

Abstract:
Data mining on large databases has been a major concern in research community. It extracts important knowledge from huge amount of data. Sometimes these data may be split among various parties. Privacy issues may prevent parties from sharing data or information about data directly. This paper addresses mining of association rules by creating privacy to the data of individual databases. In this scenario we consider two parties having confidential data and running a mining algorithm on the union of their databases without revealing any unnecessary information. Here we need to protect both privileged information and enable its use for research or other purposes.

Index Term
Data Mining, FDM , Support, Confidence, Security, Privacy.

Introduction

Data mining technology identifies patterns from large quantities of data. Most data mining tools operate by running algorithm on data which was gathered into a centralized site. However, privacy concerns may not allow us to build centralized warehouse because data is distributed among several sites where data is not allowed to transfer from one site to another. We assume homogenous databases in all sizes i.e; all sites have same schema but each site has information on different entities. This paper addresses mining of association rules over such data by reducing information shared on each site.

Deriving association rules without revealing individual information of sites can be simply done by computing global support and confidence of an Association rule AB=>C knowing only local support of AB &ABC and size of each database. It doesn't involve in sharing any individual data.

Thus, it can be done by extending Apriori algorithm to distributed case using following lemma.

i. If a rule has support >S% globally, it must have support>S% on at least one of individual sites.

This algorithm works as follows:

1. Request each site to send all rules with support at least 'S'.
2. For each rule given by site, request other sites to send count of their transaction that support the rule and total count of all transactions at the site.
3. From this rules with support 's' can be found by computing following lemmas.

$$\text{Support}_{AB=>C} = \frac{\sum_{i=1}^{sites} support\_count_{ABC(i)}}{\sum_{i=1}^{sites} database\_size(i)}$$

$$\text{Support}_{AB} = \frac{\sum_{i=1}^{sites} support\_count_{AB(i)}}{\sum_{i=1}^{sites} database\_size(i)}$$

$$\text{Confidence}_{AB=>C} = \frac{Support_{AB=>C}}{Support_{AB}}$$

The above procedure protects individual data, but it reveals about the rule it support which may be sometimes sensitive data. For example, Industry collaborations/ Trade groups may want to identify best practices to help members. But, some practices are trade secrets. How do we provide results to all while preserving secrets? This paper addresses a solution for not revealing sensitive data. We consider more than two parties here.

Overview:

Private association rule mining follows the method of passing the values to local data mining sites instead of centralized warehouse. There are two phases in this method.

1. Finding candidate item sets.
2. Finding which item set has global support/ confidence values.

Finding candidate item sets

In this phase we use commutative encryption. Here each site encrypts its frequent item set and are passed to a common party to eliminate duplicates and were decrypted. This item set is passed to each party and each party decrypts each item set which finally gives us common item set. Diagrammatically shown as follows:

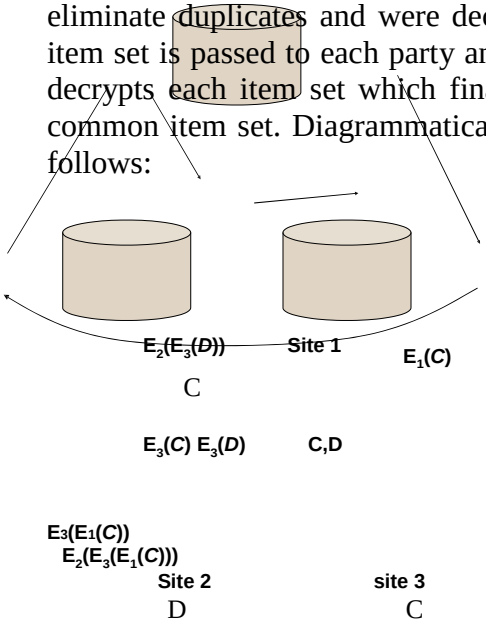

Fig1: Determining global candidate itemsets

Finding global support/ confidence values:

In this phase, each locally supporteditemsets is tested to see if it is supported globally. In the figure,the itemset *ABC* is known to be supported at one or more sites,and each computes their local support. The first site chooses arandom value *R*, and adds to *R* the amount by which its supportfor *ABC* exceeds the minimum support threshold. This value ispassed to site 2, which adds the amount by which its supportexceeds the threshold. This is passed to site three, which again addsits excess support. The resulting value (18) is tested using asecure comparison to see if it exceeds the Random value (17).If so, itemset *ABC* is supported globally.

Diagrammatically represented as follows:

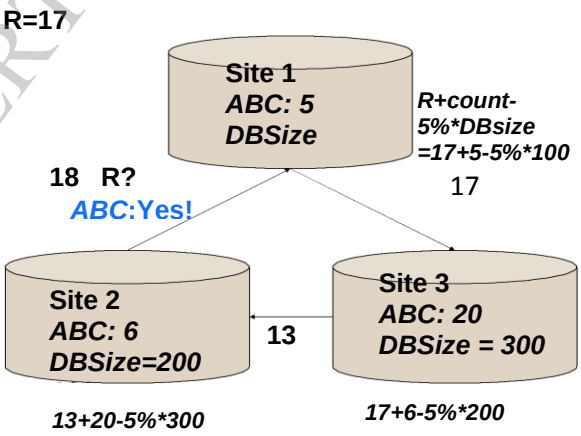

Fig. 2. Determining if itemset support exceeds 5% threshold

Background and Related work:

Privacy preserving has been used in several fields. It mainly addresses two issues

1. Privacy by data distortion
2. Building Decision tree.

In data distortion, it does not reveal information

and thus data is safe for mining. Here distorted data and information on distribution of data can be used to generate an approximation to original distribution rather than original values.

So, we can mine on distorted data directly. Recently, data distortion is applied to Boolean association rules and also used for modifying data values and reducing reconstruction effort by simply mining on distorted data.

In building decision trees, aim is to build ID3 decision tree where training set is distributed between two parties. The main idea is to find an attribute having maximum information gain and minimum conditional entropy.

Association rules mining

The association rule mining problem is defined as follows. Let I = $\{i_1, i_2, \ldots i_n\}$be set of items. Let DB is a set of transactions where each transaction T is an item set such that T $\underline{c}$I. given an item set X $\underline{c}$ I, a transaction T contains X if and only if X $\underline{c}$T. An association rule is of the form X=>Y has support S in the transaction database DB if S% of transaction in DB contain X U Y. The association rule in DB with confidence C if C% of transactions contain X and also Y. So, mining association rules is finding all rules whose support and confidence are higher than user specified support and confidence.

We consider association rules in this paper as follows:

Distributed mining

Let us assume that a transaction DB is horizontally partitioned among 'n' sites namely ($S_1, S_2, \ldots S_n$) where DB = (DB_1 U DB_2 U… DBn) and DB_i is S_i($1 \leq i \leq n$). An item set X has local support count $X.sup_i$of the transaction contain X. the global support count is given by X.sup= $\sum_{i=1}^{n} X.supi$. Now, an item set is globally supported if

X.sup $\geq$ S x ($\sum_{i=1}^{n} |DBi|$) . So, global confidence rule is given for X=>Y is {XUY}.sup/X.sup.

For distributed association rule mining, a fast distributed mining (FDM) of association rules were given and it follows the procedure as follows.

a. **Generating candidate sets:** Using classic Apriori candidate generation algorithm candidate sets are generated $CG_{i(k)}$ based on $GL_{i(k-1)}$, itemsets supported by S_i at 'i' iteration.

b. **Local Pruning:**For each item set $X \epsilon CG_{i(k)}$, compare with DB_i in site S_i to compute support $X.sup_i$. If X is locally large S_i it is included in $LL_{i(k)}$ set.

c. **Exchanging support counts:** $LL_{i(k)}$ is sent to each site and local support is computed for items and are unioned$U_i LL_{i(k)}$.

d. **Distributing results:** Each site sends its local support in $U_i LL_{i(k)}$. From this we can compute $L_{(k)}$.

Where,

$L_{(k)}$ – set of large item sets consists of k-item sets that are globally supported.

$LL_{i(k)}$ – locally large item sets consists k-item sets supported locally at site S_i.

$GL_{i(k)}$ - globally large k-item sets locally supported at site S_i.

Secure Association Rule Mining

We will now construct distributed association rule mining algorithm that provides privacy of site data. Here we consider two or more parties. Assuming no collision, let us consider i ≥ 3 be number of sites. Each site has a database DB_ihaving support S% and confidence C%. Our goal is to discover all association rules satisfying thresholds. Along with this, no site should learn contents of other site i.e; rules or support and confidence values. For this we follow the methodology as follows.

Methodology:

Our procedure follows general FDM algorithm with special protocols. First we union locally supported item set without revealing originator of particular item set. Secondly, a method for securely testing if support count exceeds the threshold

1. Locally large item sets union :FDM algorithm reveals large item sets supported by each site. For not revealing it we exchange

locally large item sets in a way that source of each item is unknown. For this we assume commutative encryption algorithm with no collision probability.

The main idea is each site encrypts locally supported item sets along with enough 'fake' item sets to hide the actual number supported. Each site then encrypts item sets from other sites, then these item sets are merged in phase 2 and 3 as discussed below and duplicates are deleted.

For union of locally large item sets, the protocol is given as follows.

Protocol1 Findingsecureunionoflargeitemsets ofsizek

Require: N ≥3sitesnumbered1..N −1,setF ofnon-item sets.

//Encryptionofalltherulesbyallsites

for eachsitei **do**

generate$LL_{i(k)}$as in steps1and2 oftheFDMalgorithm

$LLe_{i(k)}=\emptyset$

for each$X \in LL_{i(k)}$ **do**

$LLe_{i(k)}=LLe_{i(k)} \cup \{E_i(X)\}$

endfor

for $j=|LLe_{i(k)}|+1$to$|CG_{(k)}|$ **do**

$LLe_{i(k)}=LLe_{i(k)} \cup \{E_i$(randomselection fromF)}

endfor

endfor

//Encryptionbyallsites

for Round$j=0$toN−1 **do**

if Round$j=0$ **then**

Each siteisends permuted$LLe_{i(k)}$tosite$(i+1) \bmod N$

else

Eachsite iencryptsall itemsin $LLe_{(i-j \bmod N)(k)}$ withE_i,permutes,andsendsittosite$(i+1) \bmod N$

endif

endfor

//*Mergeodd/evenitemsets*

Each site i sends $LLe_{i+1 \bmod N}$ to site i mod 2

Site 0 sets $RuleSet_1 = \bigcup_{j=1}^{[(N-1)/2]} LLe_{(2j-1)(k)}$

Site 1 sets $RuleSet_2 = \bigcup_{j=1}^{[(N-1)/2]} LLe_{(2j)(k)}$

//Mergeallitemsets

Site1sends permuted $RuleSet_1$ to site$_0$

Site0sets $RuleSet = RuleSet_0 \cup RuleSet_1$

//Decryption

for i=0toN−1 **do**

Siteidecryptsitemsin RuleSetusingD_i

Siteisends permutedRuleSettositei+1m odN

endfor

Site N-1 decrypts items in RuleSet using D_{N-1}

$RuleSet_{(k)}$= RuleSet-F

Site N-1 broadcasta $RuleSet_{(k)}$ to sites 0.. N-2

In protocol 'F' represents the data that can be used as fake itemsets. $|LLe_{i(k)}|$ represents the set of encrypted 'k' item sets at site i. E_i is the encryption and D_i is the decryption by site 'i'.

Clearly, protocol 1 finds the union without revealing which item set belongs to which site. However, it reveals number of itemsets having common support between sites i.e; reveals information which is less harmful.

The phases of above protocol are discussed as follows:

Phase0: No communication occurs here. Each site runs the algorithm on its own input.

Phase1: Firstly, each site computes $LLe_{i-1(k)}$. The size of this set is size of global candidate set $CG_{(k)}$, known to each site. So, a site can produce set using a uniform random number generator.

Phase2: In this phase, each site gets fully encrypted sets of itemsets from other even sites. If there are k item sets known to e common among [N/2] odd sites, generate

'k' random numbers and put them into $LLe_{i(k)}$. Repeat for each [N/2]-1 subset. Then fill each $LLe_{i(k)}$ with randomly choosen values until it reaches size $|CG_{i(k)}|$. The same is done for site1.

Phase3: If there are k items in common between even and odd sites, site0 selects k random items from RuleSet0 and inserts them into RuleSet1. RuleSet1 is then filled with randomly generated values. Since the encryption guarantees that the values are computationally indistinguishable from a uniform distribution, and the set sizes $|RuleSet_0|$, $|RuleSet_1|$, and $|RuleSet_0 \cap RuleSet_1|$ (and thus $|RuleSet|$) are identical in the simulation and real execution. Hence proves security in this phase.

Phase4: Each site sees only the encrypted items after decryption by the preceding site. Some of these may be identical to items seen in phase2, but since all items must be in the union this reveals nothing. The simulator for site i is built as follows: take the values generated in Phase2 step $N-1-i$, and place them in the RuleSet. Then insert random values in RuleSet up to the proper size (calculated as in the simulator for Phase3). The values we have not seen before are computationally indistinguishable from data from a uniform distribution, and the simulator includes the values we have seen (and knew would be there), so the simulated view is computationally indistinguishable from the real values.

The simulator for site N-1 is different, since it learns $RuleSet_{(k)}$. To simulate what it sees in Phase4, site $N-1$ takes each item in $RuleSet_{(k)}$, the final result, and encrypts it with E_{N-1}. These are placed in RuleSet. RuleSet is then filled with items chosen from F, also encrypted with E_{N-1}. Since the choice of items from F is random in both the real and simulated execution, and the real items exactly match in the real and simulation, the RuleSet site $N-1$ receives in Phase4 is computationally indistinguishable from the real execution.

Thus above protocol is privacy preserving with above assumptions.

2. Support threshold testing without revealing support count: The above algorithm gives complete set of locally large item sets $LL_{(k)}$. From these we have to find out item set swhich are globally supported. For this, we have to know if X.sup> S% * |DB| for each item set X ϵ $LL_{(k)}$. to compare against a sum of local values we should follow the following lemmas:

$$X.sup \geq S * |DB| = S * (\sum_{i=1}^{n} |DBi|)$$

$$\sum_{i=1}^{n} X.sup_i \geq S * (\sum_{i=1}^{n} |DBi|)$$

$$\sum_{i=1}^{n} (X.supi - S * |DBi|)) \geq 0$$

So checking for support is same as checking if $\sum_{i=1}^{n} (X.supi - S * |DBi|)) \geq 0$. But it should be done without revealing $X.sup_i$ or $|DB_i|$. That can be done by following the below algorithm.

Algorithm for securely finding global support counts

Input: $N \geq 3$ sites numbered 0..$N-1$, $m \geq 2 * |DB|$

ruleset=∅

at site0:

for each r∈candidateset **do**

choose random integer x_r from a uniform distribution over 0.. m-1;

$t = r.sup_i - s * |DB_i| + x_r \pmod{m}$;

ruleset=ruleset∪{(r,t)};

endfor
sendrulesettosite1;
for i=1toN−2 **do**
for each(r,t)∈ruleset **do**
t =r.sup**i**−s∗|DB**i**|+t (modm);
ruleset=ruleset−{(r,t)}∪{(r,t)} ;
endfor
sendrulesettositei+1;
endfor
atsiteN-1:
foreach(r,t)∈ruleset **do**
t =r.sup**i**−s∗|DB**i**|+t (modm);
securelycomputeif(t −x**r**)
(modm)<m/2withthe
site0;{Site0knowsx**r**}
if (t −x**r**) (modm)<m/2**then**
multi-castrasagloballylargeitemset.
endif
endfor

In this algorithm for each item set x. each site chooses a random number x_r and adds it to support $X.supi - S * |DBi|$ and sends to next site. So the actual value was hidden by adding this random number to the original value. Similarly second site also adds its own random value and sends to other sites and so on. It can be written as $\sum_{i=1}^{n}(X.supi - S * |DBi|))$ + x_r (mod m).since |DB|≤m/2 we may also get negative values. Since no values are known to other sites, this method of considering random value is secure.

3. Finding confidence of a rule securely: To find if the confidence of a rule X => Y is higher than the given confidence threshold C, we have to check if $\frac{\{X\ U\ Y\}.sup}{Y.sup} \geq C$. Algortihm2 only reveals if an item set is supported, it does not reveal the support count. The following equations show how to securely compute if confidence exceeds a threshold using algorithm2. The support of {X U Y}.$sup_i$is denoted as XY.sup_i.

$$\frac{\{X\ U\ Y\}.sup}{Y.sup} \geq C \quad => \quad \frac{\sum_{i=1}^{i=n} XY.supi}{\sum_{i=1}^{i=n} X.supi} \geq C$$

$$\sum_{i=1}^{i=n} XY.supi \geq C * \sum_{i=1}^{i=n} X.supi$$

$$\sum_{i=1}^{i=n} XY.supi - C * X.supi) \geq 0$$

Since each site knows XY.$sup_i$and X.sup_i, we can easily use algorithm2 to securely calculate the confidence of a rule.

Conclusion and Further work

The main contribution of this paper is proposing a general framework to privacy preserving mining of association rules. We have given procedures to mine distributed association rules on horizontally partitioned data and also that privacy concerns will increase for mining of data. We also shown that our algorithms works with less communication complexity. It is possible to mine globally valid results from distributed data without revealing information that having private data of individual sources.

In the future research, a common framework with more formal and reliablefor privacy preservation will enable next generation data mining technology tomake substantial advances in alleviating privacy concerns.

References

[1] R.Agrawaland R.Srikant,"Fastalgorithms forminingassociation rules,"in*Proceedingsofthe20th InternationalConferenceonVery LargeDataBases*. Santiago, Chile: VLDB,Sept. 12-151994, pp.487–499.Available: http://www.vldb.org/dblp/db/conf/vldb/ vldb94-487.

[2] D.W.-L.Cheung,V.Ng,A.W.-C.Fu,andY.Fu,"Efficientmining ofassociationrulesindistributeddatabases,"*IEEETrans.KnowledgeDataEng.*,vol.8,no.6,pp.911–922,Dec.1996.

[3] Privacy-preserving Distributed Mining ofAssociation Rules on Horizontally Partitioned DataMurat antarcıoglu and Chris Clifton, *Senior Member, IEEE*

[3] R.AgrawalandR.Srikant,"Privacy-preservingdatamining,"in

Proceedingsofthe2000ACMSIGMODConference onManagement ofData. Dallas,TX:ACM,May14-192000,pp.439–450.[Online].Available:http://doi.acm.org/10.1145/342009.335438

[4] D.AgrawalandC.C.Aggarwal,"Onthedesignandquantification of privacy preserving data mining algorithms," in *ProceedingsoftheTwentiethACMSIGACT-SIGMOD-SIGART Symposiumon Principles of Database Systems*. Santa Barbara, California, USA: ACM, May 21-23 2001, pp. 247–255.

[5] D.W.-L.Cheung,J.Han,V.Ng,A.W.-C.Fu,andY.Fu,"Afast distributed algorithm for miningassociation rules," in*Proceedingsofthe 1996InternationalConferenceonParallelandDistributedInformationSystems(PDIS'96)*.MiamiBeach,Florida,USA:IEEE,Dec.1996, pp.31–42.