PREPROCESSING AND SECURE COMPUTATIONS FOR PRIVACY PRESERVATION DATA MINING

Sumana M¹, Hareesha K S²

¹Information Science and Engineering Department, MSRIT, India
²Department of Computer Applications, Manipal Institute of Technology, Manipal, India

ABSTRACT

Privacy Preserving Data Mining is very largely used for extracting knowledge from the data distributed over multiple sites without disclosing information. Dataset maintained by different organizations/sites involved in collaboration include missing, noisy, redundant and irrelevant data. Such data has to be preprocessed for obtaining efficient mining results from multiple sites. This paper discusses the essential preprocessing tasks performed by the individual sites to prepare the data for mining. Also in order to retain privacy, sites should jointly perform computations such as addition, division and union/intersection in a secure manner. The various implemented secure protocols to perform computations on the preprocessed datasets while classification/prediction have been conversed indicating the best approaches.

Keywords: data mining, data preprocessing, privacy preservation, secure sum, secure union, secure division.

I. INTRODUCTION

Let us consider the following scenario where several banks present in different localities want salary status of an individual satisfying certain criteria, or want to predict a fraud while issuing credit card or would like to obtain the characteristics of with salary less than 50K per month. These banks do not want to reveal any information about their data in the process of mining. Hence mining has to be done by preserving privacy.

Each of the sites has to preprocess their data to obtain appropriate mining results. For finance dataset some essential preprocessing tasks needs to be performed before performing any form of computations or mining on them. The earlier approach includes transforming nominal and categorical
attributes to their numeric form and normalizing them. But most of the data mining results need to be interpreted with different types of attributes.

In order to avoid obtaining irrelevant, inaccurate results while performing data mining tasks on the dataset we need to eliminate irrelevant information[9]. Also, certain attributes were repetitive. For example detailed occupation code (numeric attribute) has an equivalent attribute called major occupation code (categorical) in the dataset. Besides in the mining process of finding results, values of the education and class of worker attributes need to be available at various levels of abstraction. We use the concept of taxonomy trees.

Data perturbation, anonymization, or cryptographical methods have been used for privacy preserving classification as mentioned in [15]. However cryptographical methods are one the effective approaches for mining and maintaining privacy [3]. These methods use secure multiparty protocols to perform computations during knowledge discovery. Protocols such as secure sum, secure log, union, intersection and dot product have been used in most of the cryptographical data mining procedures. In this paper we discuss the various secure protocols implemented by us and compare them with the existing techniques.

Two or more parties want to conduct a computation based on their private inputs, but neither party is willing to disclose its own input to anybody else[5]. The problem is how to conduct such a computation while preserving the privacy of the inputs. Most of these protocols have been build using homomorphic or commutative encryption techniques. Many encryption algorithms are widely available and used in information security. They can be categorized into Symmetric (private) and Asymmetric (public) keys encryption. As mentioned earlier some of the commonly used secure protocols while mining are secure sum, union/intersection protocols.

II. DATA PREPROCESSING

Preparing the data for mining is known as data preprocessing as mentioned in [1] and [2]. Dataset include redundant attributes and records, missing values, irrelevant data and attributes with values at the lowest level of abstraction. Certain essential modifications that we have performed on the dataset while mining are as follows

A. Handling Irrelevant Data

While working on the finance datasets we needed to analyze the value of the age attribute. For age less than 18, had a job, children and a salary greater than 50000 then we would term such a record as irrelevant. For analyzing the characteristics of people based on the salary (using clustering method) we require to use only those objects with salary age individual sites need to update their datasets based on the age attribute. On testing our dataset it was observed that only 5% of the total records were irrelevant based on the age criteria. Such records hinder the accuracy of the models build. Hence such noisy data were placed in a different location for further analysis. However it is important that these noisy data need to be eliminated from the dataset used for mining.

B. Taxonomy Trees

For the data mining methods such as classification and clustering on the datasets we want the data of certain attributes to be available at higher levels of abstraction for example education attributes of the finance dataset. Such attributes can be discretized using the concept of taxonomy trees as shown in fig 1. The data miner at a site can access the values of the education attribute at the preschool, highschool or advanced level of abstraction. Using this concept interpreting of the results for classifying and clustering new datasets can be performed in an effective manner.
C. Eliminate Redundancy

Every dataset include duplicated entries. One of the most suitable approaches while mining involves sorting the tuples of the dataset based on quasi-identifiers, for example in finance dataset \{age, occupation, and gender\}, after which repetitive records are eliminated. In the process of the entries were reduced from the dataset maintained in each of the site.

D. Handling Missing Values

Most of the dataset have missing values\[14\]. In order to obtain results that yield better patterns from the dataset, these missing values needs to handle. Different ways\[14\] of managing missing values include computing and placing the mean for numeric values, finding and inserting the value that has occurred the maximum number of times, estimate the missing value by using the remaining values. For the type of data mining we intend to perform we have replaced the missing values with the average attribute value of the nearest neighbours for numeric values and the most probable values using the Euclidean distance measures for categorical values.

On performing preprocessing on the census dataset, the accuracy of the decision tree classifier on the rapidminer tool is shown in fig 2. As shown, the accuracy of the ID3 decision classifier is better for preprocessed data as compared to the dataset before preprocessing.

![Fig. 2. Accuracy before and after preprocessing](image)

III. SECURE SUM

In the course of building a classification model we would like to indicate the total number of records that belong to a salary class less than 10K. At one of the classification step we need to compute the number of records that belong to a particular class. Records are distributed through
various parties we need them to jointly perform the computation. Secure addition secure sum protocols are used for this purpose. A discussion on secure union is given in [4] and [5].

A. Random Secure Sum Protocol

As shown in fig 2 and [10], a party p1 generates a random to distort his private input and the summation is computed along the cycle p1->p2->p3->p1. p1 then obtains the result by subtracting the value obtained by the random number. However it been proved that the collusion of pi’s immediate predecessor and immediate successor can find out the private input. As the result this approach has hardly any capability of resisting collusion.

B. Privacy Preserving K-Secure Sum Protocol

In this approach as shown in fig 3 and mentioned in [11] and [12] of secure sum computation the party1 divides its data into segments. The amount of segments into which the data is to be divided is collectively decided by all the parties involved in the computation. Party1 distributes each of its segments to other parties and retains one for it. Other parties too divide their data into segments and distribute these segments to all the other parties.

Fig. 3. Random Secure Sum Protocol

Fig. 4. k-secure sum protocol

This approach provides outstanding security when the number of parties is greater than 3 and each of the parties have a computing facility. All the parties agree on number of segments of a data.
C. Secure sum protocol using Homomorphic Encryption

In this protocol the homomorphic encryption technique, Paillier cryptosystem approach has been used. The algorithm is as follows:

1. Generate an integer \( X > \sum_{j=1}^{n} x_j \). \( x_j \) is the value to be added at party \( j \) to obtain the final sum.
2. Party \( p_1 \) generates a random number \( r_1 \), encryption\( (E) \), decryption\( (D) \) keys and then encrypts its data \( x_1 + r_1 \) and forwards it to the its next party. It also forwards the \( (E) \) to the next party.
3. Each of the remaining parties \( p_i \) compute \( E(x_1 + x_2 + \ldots + x_i + (r_1 + r_2 + \ldots + r_i) * X) \) and passes it to \( p(i+1) \).
4. At last \( p_1 \) obtains \( D(E(\sum_{j=1}^{n} x_j + X * \sum_{j=1}^{n} r_j)) \mod X = \sum_{j=1}^{n} x_j \).

Where \( n \) is the number of parties involved in the computation of sum.

![Diagram of Secure sum using homomorphic encryption](image)

**Fig. 5.** Secure sum using homomorphic encryption

The security of this approach largely depends on the homomorphic technique used. Since paillier public key encryption technique is used capability of collusion resistance is larger when compared to the other protocols. However the disadvantage of this method is its high computation cost owing to employing homomorphic cryptosystem.

D. Comparison

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Secure Sum Protocol</td>
<td>Simple</td>
<td>Hardly any capability of resisting collusion</td>
</tr>
<tr>
<td>Privacy Preserving K-Secure Sum Protocol</td>
<td>Capability of resisting collusion.</td>
<td>Works well when number of parties ( \geq 3 )</td>
</tr>
<tr>
<td>Secure sum protocol using Homomorphic Encryption</td>
<td>Capability of resisting collusion.</td>
<td>High Computation Cost because of homomorphic Encryption</td>
</tr>
<tr>
<td></td>
<td>Works well for parties ( \geq 2 )</td>
<td></td>
</tr>
</tbody>
</table>

**Table 1:** Comparison of the secure sum protocols

As shown in table 1 a comparison of the various secure sum protocols implemented have been discussed. We have used to secure sum protocol using homomorphic encryption for secure sum computations on the finance dataset.
The computation cost involved on implementing the above three secure sum approaches on 4 sites interacting with each other. We have implemented the protocols in Java using the client server concepts where the server initiates the process of computation and the client interact in a ring format. As observed in the chart the secure sum protocol using homomorphic encryption takes more time compared to the previous approach but is much more secure.

![Chart showing computation time of secure sum protocols](image)

**Fig. 6.** Computation time of the secure sum protocols in millisecond

IV. SECURE UNION

Let us imagine a situation where data miners would want to know the tuples that satisfy a criterion. Here the assumption is that data is vertically distributed, which means information (characteristics/attributes) regarding each record is maintained at multiple sites. Sites should collectively obtain the union of the tuples. The computation of union is to be performed in a secure manner without revealing any other information other than what has been queried for. A discussion on secure union is given in [4] and [5].

A. Commutative Secure Union protocol

Given in Fig.7 is the first level commutative secure union protocol as mentioned in [6]. This protocol starts with party 1 who encrypts its data and forwards it to site 2. Party 2 encrypts its values as well as the encrypted result values obtained by the other sites. These values are moved on to the neighboring parties where again they are encrypted by the key provided by the party. In the process all the values maintained by the n parties are encrypted n times by the encryption keys of each of the parties.

![Diagram of level 1 commutative secure union protocol](image)

**Fig. 7.** level 1 of the Commutative Secure Union Protocol
Once the data maintained at each of the sites is encrypted. Party1 shuffles all the n times encrypted values and eliminates the duplicate values. Because of commutative encryption property used while deciding on encryption and decryption approach i.e. $E_3 (E_2 (E_1 (ABC))) = E_2 (E_3 (E_1 (ABC)))$, we can identify duplicate values. Finally the encrypted values are decrypted by each of the n sites to know that the union of all the data maintained by multiple sites.

For the example in fig 7, union of 3 parties will result in ABC and ABD. The above approach uses commutative encryption technique which adds a large overhead to the mining process. This protocol requires the need for a trusted party.

B. Secure Union using Public-key Cryptography

Site 1 generates a public encryption key $k_1$ and a private key $k_2$ (decryption). Site1 then forwards its encrypted values to site2 who adds its encrypted values to the database sent by site1 and shuffles them.

The shuffled dataset is passed to the next site. The next site repeats the step of encryption, shuffling and forwarding. All sites works locally finding local values and encrypting them using the public key sent by site 1. The last party passed the database obtained, shuffles it and sends it to site1. Site1 obtains the union, decrypts them and forwards it to all the other parties. Public key cryptography used for 3 parties with data is as shown in fig 8.

This method reduces the time of mining process compared to the previous method. Also the parties can share the union of their data without the need for an outside trusted party.

![Fig. 8. Secure union using public key cryptography](image)

C. Comparison

![Fig. 9. Computation time for Algo1 and Algo2 with the pohlig hellman encryption](image)
Given that algo1 is secure union protocol using commutative encryption and algo2 is secure union protocol using public encryption. Both the algorithms have been implemented for securely computing the union of data available at 3 parties. The computation time required for each of the approaches when, given the number of tuples involved in secure union computation is being plotted in fig 9. This graph shows that algo2 has a better computation method over algo1.

V. SECURE DIVIDE

As shown in [8], where naïve Bayesian approach is used for privacy preserving classification. Here as per the protocol used on horizontal distributed datasets, to find the number of tuples that belongs to a class and with certain features, besides secure sum, secure division protocol is used. ie All parties want to calculate \( p_{yz} = C_{yz}/n_y \), where \( C_{yz} \) represents number of instances belonging to class \( y \) and having an attribute value of \( z \) and \( n_y \) indicates number of instances belonging to class \( y \).

A. Secure Division using Secure Log Approach

As mentioned in [8] the algorithm is implemented as follows. This is implemented by 2 parties p1 and last party pn

- Party p1 generates two random values \( R_c \) and \( R_n \) and forwards it to pn.
- Party pn generates \( R_c + Cyz \mod p \) and \( R_n + ny \mod p \) for some value \( p \) which is large enough to fit the result.
- Party 1 generates another random value \( v1 \) and \( u1 \) and forwards it to party pn.
- Party n obtains \( v2 = C \cdot \ln(R_c + Cyz - R_c \mod p) \mod p - v1 \) and \( u2 = C \cdot \ln(R_n + ny - R_n \mod p) \mod p - u1 \).

Where \( C \) is a large value assume by party n.

- Party n calculates \( s_k = v2 - u2 \mod p \) and sends \( C \) (to all parties) and \( s_k \) to party1.
- Party 1 calculates the \( s_1 = s_k + v1 - u1 \mod p \) and sends it to all parties.

Now all parties can calculate the divided result as \( \exp(s_1/C) \).

However in this approach the amount of security is less since the numerator cannot exceed 512 bits. This approach will also work when the numerator and the denominator are encrypted. The values are in encrypted form in most of the cryptograpical approaches of data mining. Hence this method is not suitable.

B. Secure Division for encrypted Data

Since the numerator(n) and the denominator(d) are in the encrypted form we use this method. The encrypted values are of BigInteger type that exceeds the size of 512 bits as in the earlier approach. The working[13] is as follows:

- The sub-linear protocol for obtaining \( 2^{ld} \) is used by first obtaining the binary equivalent of \( d \) (l-bit size) and using binary search approach on bits in \( d \).
- Then the inverse of \( 2^{ld} \) is generated i.e. \( 2^{-ld} \).

From this \( [p] = (2^{ld} - d) \cdot 2^{-ld} \).

- Then party 1 \( a^- = 2^k \cdot [2^{ld}]. \) poly(p). where poly(p) uses the square and multiply method to obtain a evaluate where \( w = 2^R \) for some integer \( R \).
- The \( q^w = n \cdot a^- \)
- \( q^- = \text{truncate} q^w \)
- generate \( [r] = n - d.q^- \)
- Obtain pos_err = result of the bit comparison (\( <= \)) between\([d]+[d] \) and \([r]+[d] \).
- Obtain neg_err = result of the bit comparison (\( > \)) between\([d] \) and \([r] + [d] \).

Obtain the divided result \( q = q^- + \text{pos_err} - \text{neg_err} \).

Party 1 circulates \( q \) to all the sites which is the divided result.
The entire sequence of events are performed in party 1 itself hence except for broadcasting the result to all the sites there is no other communication cost. Also security is high since the computations are performed on the encrypted datasets. It has also been observed that the number of rounds is $O(\log_2 l)$ where $l$ is the bit size of $d$.

C. Comparison

Given below is the comparison between the two protocols used for performing secure divisions. Secure division approach involves paillier encryption approach hence though computation cost increases the communication cost is reduced compared to the secure log approach. As observed in fig 10 secure division is better than secure log approach for performing secure distributed division. The results have been obtained with data maintained at 3 sites. X axis indicates the number of tuples in each of the site. Y axis indicates the time in millisecond.

![Computation time with dataset size on the X axis and execution time in the Y axis](image)

**Fig. 10.** Computation time with dataset size on the X axis and execution time in the Y axis

VI. CONCLUSIONS

The preprocessing techniques have been used on the datasets maintained on each of the sites in order to prepare the data for mining. Secure protocols have been implemented in Java platform with a minimum of 3 parties and analyzed the working of the above methods. The best approaches such as secure sum using homomorphic encryption, secure union using encryption and secure division on encrypted data has been used in implementing our data mining approaches on the horizontally classified datasets maintained at 3 sites.

VII. ACKNOWLEDGMENT

We acknowledge Manipal University and M S Ramaiah Institute of Technology for giving us an opportunity for working and publishing this paper.

REFERENCES

**Journal Papers**


Books:

Conferences: