Secure Multiparty Computations for Privacy Preserving Classifiers

M. Sumana, K. S. Hareesha

***Abstract*—**Secure computations are essential while performing privacy preserving data mining. Distributed privacy preserving data mining involve two to more sites that cannot pool in their data to a third party due to the violation of law regarding the individual. Hence in order to model the private data without compromising privacy and information loss, secure multiparty computations are used. Secure computations of product, mean, variance, dot product, sigmoid function using the additive and multiplicative homomorphic property is discussed. The computations are performed on vertically partitioned data with a single site holding the class value.

***Keywords*—**Homomorphic property, secure product, secure mean and variance, secure dot product, vertically partitioned data.

I. Introduction

MULTIPLE parties need to collaborate to carry out computations on their private data that is horizontally or vertically partitioned. Secure multiparty computations indicate calculating a function on their inputs, without revealing any other information as mentioned in [1]. The joint computations should be constructed such that parties learn nothing except the correct output. As per the literature [2], these protocols can be used in electronic voting, electronic auctions, electronic cash schemes, contract signing, anonymous transactions, private information retrieval schemes, identifying fraud, disease outbreaks. The essential definitions that effect the secure multiparty computations are privacy, correctness of the output, independence of the inputs, guaranteed output delivery and fairness. It is also observed [3], [4] that secure computations work best when cryptographical methods are used. The concepts of cryptography are provided in [5]-[8] , discusses the purpose of homomorphic cryptographic property to perform secure computations. Paillier has proved to having a semantically secured homomorphic cryptographic protocol. The Paillier's key generation and encryption technique, results in satisfying the homomorphic property. This homomorphic property enables computations on encrypted data. [9] builds a secure backpropagation protocol using secure computation of the sigmoid function. Privacy preserving Naive Bayes classifier is constructed using secure dot product and secure sum protocols [10].

Secure Homomorphic protocols have been constructed for vertically partitioned data with only a participating party holding the class label attribute. These protocols are built for multiple parties who do not want to disclose the value that they encompass. The master party (with the class label) initiates the process of computation and is the one that maintains the final computed result. These secure protocols have been used in constructing the privacy preserving data mining algorithms.

Sumana, M. is an assistant professor in the Department of Information Science and Engineering, MSRIT, Bangalore, India (e-mail: sumana.m@msrit.edu).

Dr. Hareesha, K. S. is a professor and Head of the department of Computer Applications, MIT, Manipal, India (e-mail: hareesh.ks@manipal.edu).

A. Homomorphic Property

The probabilistic asymmetric algorithm for public key cryptography is discussed in [12]. An important property of the Paillier cryptosystem (1999) as seen in [11] assists in performing definite types of computations on the ciphertext and produce an encrypted result which when decrypted equates the outcome of an operation carried out on the plaintext.

The two main homomorphic properties used are

1. Homomorphic Addition

Let $E(v_i)$ indicate the encryption of a plaintext v_i and $D(v_i)$ indicate decryption of value v_i then,

$$D(E(v_1)*E(v_2)*E(v_3)*...........*E(v_n) \text{ mod nsquare }) = (v_1 + v_2 + v_3 + ... + v_n) \text{ mod } n.$$

In our case the values v_1, v_2,……., v_n are the locally computed values at each site.

2. Homomorphic Multiplication

Let $E(v_i)$ indicate the encryption of a plaintext v_i and $D(v_i)$ indicate decryption of value v_i then,

$$D(((E(v1)^{v2})^{v3})^{\cdots\cdots})^{vn}) \text{ mod nsquare})=(v1*v2*v3*.............*vn) \text{ mod } n$$

In our case the values v_1, v_2,……., v_n are the locally computed values at each site. This property is used to securely compute the product, mean and variance.

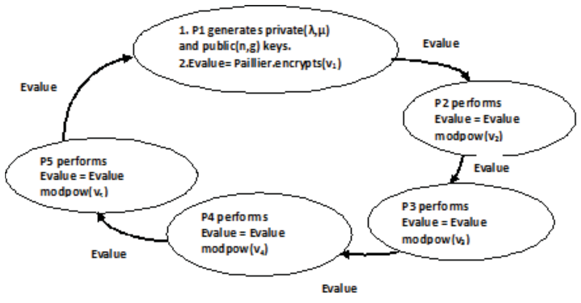
1. P1 generates private(λ,μ) and public(n,g) keys.
2.Evalue= Paillier.encrypts(v₁)
Evalue
P2 performs Evalue = Evalue modpow(v₂)
Evalue
P3 performs Evalue = Evalue modpow(v₃)
Evalue
P4 performs Evalue = Evalue modpow(v₄)
Evalue
P5 performs Evalue = Evalue modpow(v₅)
Evalue

Fig. 1 Secure Product computation

II. Secure Multiparty Protocols

A. Secure Sigmoid Calculation

Party1(master party) locally computes $s_1 = (x_1*w_{1j}+x_2*w_{2j}+\ldots\ldots\ldots+x_n*w_{nj})$ and e^{-s1}. Remaining parties i=2 to k locally compute $s_i = (x_{n+1}*w_{(n+1)j}+\ldots..+x_s*w_{sj})$ and e^{-si}.

Party 1 to k use Algorithm 1 (secure product) to obtain $e^{-S} = e^{-s1}* e^{-s2}*\ldots..* e^{-sk}$. Fig. 1 shows the diagrammatic representation of secure product.

Algorithm 1: Secure Product Computation

```
Input: k parties have values $v_1,v_2\ldots.. v_k$.
Output: Product = $v_1*v_2*\ldots\ldots*v_k$
1. Party 1 with $v_1$ generates the private and public keys.
2. Party 1 paillier encrypts $v_1$ = (int)($v_1$*100) (as only integer values can be encrypted) to obtain evalue and forwards it to neighboring party 2.
3. Party I = 2 to k update evalue = evalue.modpow((int)$v_i$* 100) and forward it to its neighbor.
4. Party k forwards evalue back to Party 1.
5. Party 1 now performs Product = Paillier.decrypts(evalue)/100 to get Product= $v_1*v_2*\ldots\ldots..*v_k$ without knowing the values $v_2,v_3,\ldots\ldots\ldots,v_k$.
```

B. Secure Mean and Variance Computation

To compute the mean for a class label Y, we need to add only those numeric values that belong to the class Y. But a site k (that is not a master site) is not aware of the class to which their numeric value belongs to, hence mean has to be securely computed.

Mean (class_label = 'Y') = sum_of_numeric_values$_k$/(number of tuples belonging to class = 'Y')

sum_of_numeric_values$_k$ = $\sum_{i=1}^{n}$ numericvalue $*$ class_value(given as 1 or 0)

The sum_of_numeric_valuesk is to be computed securely as the class_value is not known to site k (that has the numeric value). Hence the homomorphic property is used to perform this computation as follows:

sum_of_numeric_values$_k$ = $\prod_{i=1}^{n}$(Y_Encrypt[i]modulos(numericvalue))

Master Site (party 1 computes mean)

Mean (class_label = 'Y') = Decrypt (sum_of_numeric_values$_k$) / (number of tuples belonging to class = 'Y')

To compute the variance the following principle is used. Non-secure computation of variance of n numbers:

$$[(\text{numericvalue}_1\text{-mean})^2+(\text{numericvalue}_2\text{-mean})^2+\ldots\ldots\ldots..+(\text{numericvalue}_n\text{-mean})^2]/n$$
$$= [\text{numericvalue}_1^2 + \text{numericvalue}_2^2 +\ldots..+ \text{numericvalue}_n^2+ n*\text{mean}^2 - n * \text{mean} *(\text{numericvalue}_1 + \text{numericvalue}_2+\ldots\ldots\ldots\ldots.+\text{numericvalue}_n)]/n$$

For secure computation, the value of mean and n is known only to the master site, but site k (with the numeric attribute) only is aware of the remaining values numeric value$_1$, numeric value$_2$… numeric value$_n$. Hence site k obtains:

sum_of_numeric_values$_k^2$ = $\prod_{i=1}^{n}$(Y_Encrypt[i]modulos(square(numericvalue$_i$)))

and forwards to master site. Master site computes:

variance$_k$ = [decrypt (sum_of_numeric_values$_k^2$) + n * mean^2 – n*mean * decrypt(sum_of_numeric_value$_k$)]/n

A detailed description of computing the mean and variance for numeric values available at participating sites is provided in Algorithm 2.

Algorithm 2: Secure Multiparty Mean and Variance Computation

```
Master Site (two class labels Y and N)
1. For i= 1 to N //N is the number of tuples
Begin
 if class label value = 'Y'
{Y_Encrypt[i] = Paillier_Encrypt(1);
N_Encrypt[i] = Paillier_Encrypt(0);
}
else
{Y_Encrypt[i] = Paillier_Encrypt(0);
N_Encrypt[i] = Paillier_Encrypt(1);
}
Forward Paillier.nsquare, vectors Y_Encrypt and N_Encrypt to the remaining sites.
Paillier.nsquare is the square (p*q) where p and q are 2 large prime numbers.
2. Site p = 2 to k
2a. Obtains Mean_Numerator_p_Y=
      $\prod_{i=1}^{n}$ Y_Encrypt[i] modpow(numericval[i], Paillier. nsquare)
where numericval[i] is the numeric value of the attribute in tuple i in party p
2b. Computes variance_p_Y=
      $\prod_{i=1}^{n}$ Y_Encrypt[i] modpow(square(numericval[i]), Paillier. nsquare)
2c. Obtains Mean_Numerator_p_N =
      $\prod_{i=1}^{n}$ N_Encrypt[i] modpow(numericval[i], Paillier. nsquare)
where numericval[i] is the numeric value of the attribute in tuple i.
2d. Computes
 variance_p_N =
$\prod_{i=1}^{n}$ N_Encrypt[i] modpow(square(numericval[i]), Paillier. nsquare)
2e. Forwards Mean_Numerator_p_Y, Mean_Numerator_p_N, variance_k_Y and variance_k_N to master site.
3. Master Site computes mean and variance of the numeric attribute r of site k as follows
3a. Mean_r_k_Y = Paillier_Decrypt(Mean_Numerator_p_Y)/ no_tuples_Y
Where no_tuples_Y indicates the number of tuples belonging to class Y.
3b. To compute variance_r_k_Y
      i. Temp_val1 = 2* Mean_r_k_Y* Paillier_Decrypt(Mean_Numerator_p_Y);
      ii.variance_r_k_Y=(Paillier_Decrypt(variance_p_Y)-Temp_val1+ no_tuples_Y*square(Mean_r_k_Y))/no_tuples_Y.
3c. Mean_r_k_N = Paillier_Decrypt(Mean_Numerator_p_N)/ no_tuples_N
where no_tuples_N indicates the number of tuples belonging to class N.
3d. To compute variance_r_k_N
      i. Temp_val2 = 2* Mean_r_k_N* Paillier_Decrypt(Mean_Numerator_p_N);
      ii.variance_r_k_N=(Paillier_Decrypt(variance_p_N)-Temp_val2+no_tuples_N*square(Mean_r_k_N))/ no_tuples_N.
Once the Variance and Standard Deviation= squareroot(variance)
```

C. Secure Dot Product Computation

K (xixj) is a dot product for linear SVM classifiers. The dot product of tuples xi and xj , K(xixj) = $\sum_{k=1}^{p} x_{ki} * x_{kj}$, where p indicates the number of features of the tuples. Algorithm 3 gives a detailed description of this protocol.

Algorithm 3: Multiparty Secure Dot product

```
1.  Party k locally computes the dot product from tuple x to x` as
    follows Fk(x,x`) = (x1*x`1)+(x2*x`2)+........+(xn*x`n) for each
    of its n features other than the class attribute, paillier encrypts to
    obtain Ek=E(Fk(x,x`)) .It broadcast its public keys (n,g) to all the
    remaining parties(1 to k-1).
2.  Party I(I = 1 to k-1 i.e remaining parties) locally compute the dot
    product
  2.1. FI(x,x`) = (x1*x`1)+(x2*x`2)+........+(xm*x`m) for each of its
       m features for samples x and x`.
  2.2. Obtains EI = E(FI(x,x`)) using the keys sent by party k.
3.  Party 1 forwards its val= E1 to neighboring party 2  party I = 2 to
    k-1 update val = val *EI and forward it to its neighbor I+1.
4.  Party k on receiving the encrypted value val  from (k-1)th party
    performs
Encrypt_dot_prod(x,x`) = val *Ek
```

It further obtains F(x,x`) i.e the final dot product from tuple x to x` for all the features distributed from site 1 to site k i.e. F(x,x`)=F1(x,x`)+F2(x,x`)+.........+Fk(x,x`) by decrypting the Encrypt_dot_prod(x,x`) i.e D(Encrypt_dot_prod(x,x`). It is important to note that the homomorphic property of Paillier is used to obtain the dot product of tuples x and x` with distributed features at multiple sites. Fig. 2 gives a diagrammatical representation of the above protocol. The numbers of parties involved in mining are 5 with party 5 considered as the master site that initiates the secure dot product computation process by broadcasting the paillier public keys n and g to all the participating parties.

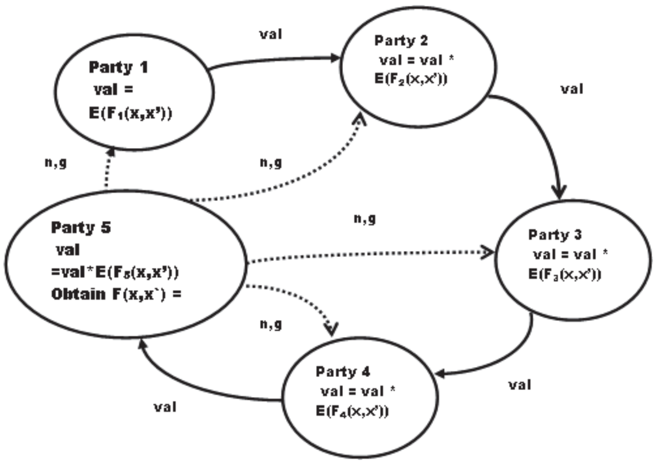

Fig. 2 5- party Secure Dot product computation from tuple x to x`

III. Results Obtained

Because of the cryptographic nature, each of the parties does not learn anything about the other inputs hence privacy of the data is maintained during computation. Also due to the homomorphic property, the parties obtain the result as desired by the function indicating the correctness of the input. Privacy preserving Naive Bayes classifier is built using the secure mean and variance protocols. Secure sigmoid protocols are used to construct privacy preserving back propagation algorithms. Similarly secure dot product protocols are used for the construction of privacy preserving linear support vector machines. The results are as shown in Fig. 3.

The constructed protocols are compared with the protocols in [9], [10]. As observed algorithms build using homomorphic secure methods discussed above perform better.

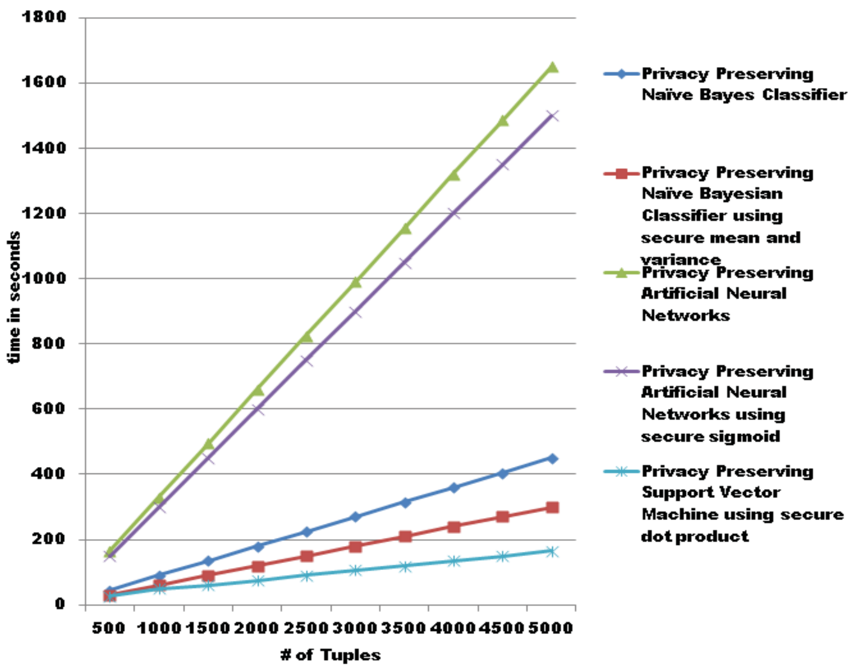

Fig. 3 Computation time Comparison

IV. Conclusion

This paper discusses the secure protocols that can be used for privacy preserving data mining. The commonly used methods are secure product, sigmoid, dot product, mean, and variance. Classifiers build using these secure computations perform better than the previous classifier models as the communication cost is lower. In future, we extend our work for parties that behave as malicious adversaries. These secure computations with their homomorphic nature can also be used for cloud computing.

References

[1] B. Pinkas, “Cryptographic techniques for privacy-preserving data,” *ACM SIGKDD Explorations,* pp. 12-19, Volume 4, Issue 2, 2002.
[2] B. P. Yehuda Lindell, "Secure Multiparty Computation for Privacy Preserving Data Mining," *The Journal of Privacy and Confidentiality , Number 1,* pp. 59-98, 2009.
[3] P. D. Qingkai Ma, "Secure Multi-Party Protocols for Privacy Preserving Data Mining," *Wireless Algorithms, Systems, and Applications, Lecture Notes in Computer Science Volume 5258, Springer,* pp. 526-537, 2008.
[4] R. Canetti, "Security and Composition of Multi-party Cryptographic Protocols," *JOURNAL OF CRYPTOLOGY,* 1998.
[5] O. Goldreich, "Cryptography and Cryptographic Protocols.," *In Distributed Computing,* p. 177–199, 2003.
[6] J. K. a. Y. Lindell, "Introduction to Modern Cryptography," *CRC Press,* 2007.
[7] O. Goldreich, "Foundations of Cryptography: Volume 2 – Basic Applications.," *Cambridge University Press,* 2004.
[8] M. Kumbhar and R. Kharat, "Privacy preserving mining of Association Rules on horizontally and vertically partitioned data: A review paper," in *IEEE Hybrid Intelligent Systems (HIS), 2012 12th International Conference*, 2012.
[9] T. C. a. S. Zhong, "Privacy-Preserving Backpropagation Neural Network Learning," *IEEE Transactions of Neural Networks,* p. Vol 20, 2009.
[10] M. K. C. J Vaidya., "Privacy Preserving Naïve Bayes Classification," *The VLDB Journal 17,* pp. 879-898, 2008.
[11] P. Paillier, " "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes"," *EUROCRYPT. Springer.,* p. 223–238, 1999.
[12] M. O. Keeffe, "The Paillier Cryptosystem, A Look into The Cryptosystem and Its Potential Application," the College of New Jersey Mathematics Department, 2008.