Secure Two Party Privacy Preserving Classification Using Encryption

K. S. Hareesha and M. Sumana

Abstract—**Distributed computing demands training methods that handle distributed input data. While training, as the parties that collaborate are concerned about the privacy of their data, the concept of privacy preservation should be extended in data mining classifiers. In this paper, data holders make practical use of their data to construct a precise classifier model by not revealing either their training data or the intermediate results. We propose a privacy preserving two-party Naive Bayes classifier for horizontally partitioned distributed data. This protocol is built such that both the parties through their random shares compute probabilities, mean and variance. To classify a new instance with numeric attributes, parties need to jointly cooperate with each other. The correctness and the security analysis of our algorithm are provided.**

Index Terms—**Privacy preserving, data mining, horizontal partitioned, paillier cryptosystem, RSA cryptosystem, secure computations, homomorphic property, commutative property.**

I. Introduction

The ability to store personal data has increased and the advanced capability of the data mining algorithms to extract this information has enhanced the need of privacy preservation while mining data. In privacy preserving data publishing different transformation techniques such as randomization, k-anonmity and ℓ-diversity are linked with privacy [1]-[3]. The perturbed data can be used in combination with the traditional data mining methods such as association rule mining, classification or clustering. Other form of privacy includes changing the results of data mining applications to preserve privacy. An example of this approach is association rule hiding methods [4]. In [1] the data mining will be performed by third party. Before the data could be handed over to a third party the confidential values in the database, such as the salary of employees, needed to be perturbed in a way that the original probability distribution could be estimated from the perturbed data but not the original data values. A decision tree [5] is constructed from the perturbed data within a certain error margin that the authors approve. The authors in [6] involves two parties with private data sources who would like to do data mining without seeing each other's data and propose cryptographic techniques to achieve that. They also demonstrated their approach on decision tree construction. In [7], [8] Yao's millionaire's problem, two millionaires want to find out who is richer, but without revealing their wealth to the each other. Here the ability to compare or compute numerical data is decisive in most data mining tasks. The definition of privacy is based on equivalence to having a trusted third party perform the computation. The third party acts in complete segregation, calculates the result and reveals them. After revealing the results, the trusted party need not remember everything it has seen. Finding such a third party is very rare resulting in secure multiparty computations being constructed such that no party learns more than it what the third party would reveal.

This instigated research in secure multi-party computation which includes two or more players who securely compute on their joint inputs. Nothing but the final result of the computation will be revealed to the parties. Also, no party will know the inputs of the other parties. Any computation which can be done in polynomial time by one party can be performed securely by multiple parties [9]. However, the vital understanding needed in these standard protocols is encryption. The definition of security in secure multiparty protocols should intuitively be similar to protocols where all players send their inputs to an honest third party, which performs the computation and returns the results to the players. This perfect protocol is clearly secure, since no player sees anything else than its own inputs and outputs. Some of the well-known public-key encryption schemes used in these protocols is RSA and ElGamal. RSA encrypts messages of approximately 1024 bits in ciphertexts of 1024 bits. El-Gamal is an elliptic curve based encryption which can handle messages of around 160 bits that are much smaller than what RSA can handle. Public key encryption schemes are easier to use and administrate.

A concept used in computing a wide range of functions with computational security is homomorphic encryption. With homomorphic encryption we can avoid the bitwise encryption from the scrambled circuits. Homomorphic encryption schemes are a particular class of public key encryption schemes. The first homomorphic cryptosystem, called the Goldwasser-Micali (GM) cryptosystem, was proposed in [9]. Due to its prohibitive message expansion during encryption, this approach is not useful for data mining applications. The extension of the GM cryptosystem is the Benaloh cryptosystem [9] which allows the encryption of larger block sizes at a time. Although the message expansion is not as bad as in the GM cryptosystem, it is still not suitable for data mining applications. Furthermore, the fact that the decryption is based on exhaustive search over all possible plain-texts also makes the Benaloh cryptosystem unpractical for privacy preserving data mining. The authors of [9] use homomorphic encryption for computing secure scalar products used in privacy preserving data mining. A more

Manuscript received January 4, 2015; revised December 12, 2015.

K. S. Hareesha is with the Department of Computer Applications, MIT, Manipal, India (e-mail:hareesh.ks@manipal.edu).

M. Sumana is with M. S. Ramaiah Institute of Technology, Bangalore, India (e-mail: sumana.m@msrit.edu).

recent scheme is the Paillier cryptosystem [10], which avoids many of the drawbacks of the earlier homomorphic cryptosystems. The Paillier cryptosystem provides fast encryption and decryption algorithms, and it encrypts 1024-bit messages in ciphertexts of at least 2048-bits, which is reasonable if we work with large plaintexts [11] discusses the usage of homomorphic encryption methods for privacy preserving k-means clustering.

We propose a light-weight two-party distributed algorithm for privacy preserving Naive Bayesian classification with horizontally partitioned dataset. Horizontal partition of data means that the same sets of information about different entities are distributed on different sites. An example in distributed data mining where privacy can be of importance is in the field of medical research. All the features involved in distributed data mining are known to the sites. Hospitals holding patient information such as age, location, type of disease, treatment and result of treatment. These hospitals can cooperate with each other to identify whether the type of treatment given to a person is successful or not.

A. Homomorphic Property

Homomorphic encryption is a form of encryption which allows specific computations to be carried out on ciphertext and obtain an encrypted result which decrypted matches the result of operations performed on the plaintext. For instance, one person could add two encrypted numbers and then another person could decrypt the result, without either of them being able to find the value of the individual numbers. Encryption techniques such as ElGamal [12] and Paillier [10] have the homomorphic property, i.e. for messages m1 and m2 $D(E(m1, r1).\ g^{m2}) = m1+m2 \bmod n$ without decrypting any of the two encrypted messages.

Also $D(E(m1, r1) \times E(m2, r2) \bmod n^2) = m1 + m2 \bmod n$. D indicates decryption and E indicates encryption.

In our algorithms, a homomorphic cryptographic scheme of Paillier is utilized. This asymmetric public key cryptography [10], [13] approach of encryption is largely used in privacy preserving data mining methods. The scheme is an additive homomorphic cryptosystem that are used in algorithms where secure computations need to be performed. Paillier is a public key encryption scheme which can be defined on any cyclic group. The original cryptosystem provides semantic security against chosen-plaintext attacks.

Paillier schemes have probabilistic [14] property, which means beside the plain texts, encryption operation needs a random number as input. Under this property there can be many encryptions for the same message. Therefore no individual party can decrypt any message by itself.

B. Commutative Property

The asymmetric (public key) encryption scheme that performs $E1(E2(M)) = E2(E1(M))$, where $E1$ and $E2$ are the encryption keys used in party 1 and 2 and M is the message then it is said that the scheme is commutative [15]. This encryption technique [15]-[17] founded by Rivest, Shamir & Adleman of MIT in 1977 is best known and widely used public-key scheme known to have a commutative property. It is based on exponentiation in a finite (Galois) field over integers modulo a prime. This method uses large integers (e.g. 1024 bits). Apply the EME-OAEP [17] encoding operation to the message M and the encoding parameters P to produce an encoded message EM to encrypt the message. Then convert this encoded message EM to an integer message representative m. Apply the RSAEP encryption primitive to the public key (n, e) and the message m to produce an integer ciphertext C. Convert this ciphertext to ciphertext of length k octets. To decrypt the ciphertext C the owner, uses their private key $K = \{d, p, q\}$ convert the ciphertext C to an integer ciphertext representive C. Use the RSA decryption primitive to the private key K and the ciphertext c to produce an integer message representative m. Convert the message representative m to an encoded message EM of length k-1 octets. Further apply the EME-OAEP decoding operation to the encoded message EM and the encoding parameter P to recover a message M. In our approach this property is used to perform computations such as secure product and secure division.

II. Privacy Preserving Naive Bayes Classification Using Random Shares

The protocols presented work well on horizontally partitioned data and do not compromise on security. Here all the attributes are known to the parties involved in building up the model. The numerator, denominator, mean and variance are maintained as random shares in either of the parties. However the final probability and variance is known only to Party A. The total number of instances belonging to a particular class is also unknown by either of the parties involved in the computation. However collaboration between the parties is essential for classifying a new instance with numeric attributes. The approaches of computing the probability for categorical and numeric attributes are dissimilar.

A. Categorical Attributes

Protocol 1 indicates the computation of probability for a value of the categorical attribute held by both the parties. The Party X locally compute the count of the number of instances that an attribute value i belongs to class y (C^x_{yi}) and the number of instances that belong to class y (n^x_y) from their training data. Each of them then communicate with each other to compute the function $v1/(v2+v3)$ using protocol 2 which uses the commutative property of RSA-OAEP encryption method. Now each of these parties have 2 random shares each as the result of the computation i.e. h^A_{1i}, h^A_{2i} in Party A and h^B_{1i} and h^B_{2i} in Party B for attribute value i. Further homomorphic encryption property of Paillier's is used to find the sum of h^A_{1i}, h^A_{2i} (Party A) h^B_{1i}, h^B_{2i} (Party B). In Protocol 2, for a range of values i, Party A performs $v1/(v2+i) - R$, 19 rounds it up by a few decimal points and converts it into a Biginteger, where R is a random value. These range of values are encrypted using RSA and sent to Party B. Party B chooses the v3th encrypted value, encrypts its value and forwards it to Party A. Party A decrypts the received encrypted value and forwards it to Party B which has the value $v1/(v2+i) - R$. This is possible because of the commutative property of RSA approach.

Protocol 1: Handling a categorical attribute

Requirements: 2 parties, *r* class values, *x* attribute values

C^{x}_{yi} — represents number of instances with party P_x having class *y* and attribute value *i*.

n^{x}_{y} — represents number of instances with party P_x having class *y*.

For all attribute value $i = 1$ to x

For all class value *y* do

Party A locally computes C^{A}_{yi} and n^{A}_{y} for every attribute value *i*.

Party B locally computes C^{B}_{yi} and n^{B}_{y} for every attribute value *i*.

Party A and Party B use algorithm 2 to jointly compute $h^{A}_{1i} + h^{B}_{1i} = C^{A}_{yi}/(n^{A}_{y} + n^{B}_{y})$, where h^{A}_{1i} is the random share maintained in Party A and h^{B}_{1i} is the random share in Party B.

Similarly Party A and Party B use algorithm 2 to jointly compute $h^{A}_{2i} + h^{B}_{2i} = C^{B}_{yi}/(n^{B}_{y} + n^{A}_{y})$, where h^{A}_{2i} is the random share maintained in Party A and h^{B}_{2i} is the random share in Party B .

Party B performs $E\ (h^{B}_{1i} + h^{B}_{2i})$ using paillier encryption with the public key forwarded by Party A and forwards it to A.

Party A performs $D\ (E\ (h^{A}_{1i}+ h^{A}_{2i}) \times E(h^{B}_{1i} + h^{B}_{2i}))/1000$ to obtain the probability.

End for

Note: Since Homomorphic encryption approach is used $D\ (E\ (\text{Sum1}).\ E\ (\text{Sum2})) = \text{Sum1} + \text{Sum2}$.

To indicate the class to which the test/new tuple with categorical attributes can be classified by party A without any collaboration.

Protocol 2: To compute $v1/(v2+v3)$

Where $v1$ and $v2$ are known to Party A and $v3$ is known only to Party B.

1) Party A computes $(v1/(v2+(i/1000))\times1000 - R$, for a range of $i = 0$ to n, where R is the random value and n is the maximum value that Party B can have.
2) Party A further encrypts all the values using RSA and sends it to Party B in the increasing order of *i*.
3) Party B selects the $(v3\times1000)^{\text{th}}$ encrypted value and further encrypts it and sends it to Party A.
4) Party A decrypts the value obtained and forwards this value to Party B.
5) Party B decrypts the value and has the random share h^B.

Note: Party A has the random share $h^A = R$ and the random share $h^B = (v1/(v2+(v3/1000))\times1000) - R$.

B. Numeric Attributes

Protocol 3 securely computes variance and standard deviation of numeric attributes. Each of the parties locally calculates S^{X}_{y} and N^{x}_{y}. Party A and B communicate using protocol 2 to obtain random shares h^{A}_{1}, h^{B}_{1} as result of $S^{A}_{y}/(N^{A}_{y} + N^{B}_{y})$ and h^{A}_{2} , h^{B}_{2} as the result of $S^{B}_{y}/\ (N^{B}_{y} + N^{A}_{y})$. To evaluate variance, each of the parties needs to evaluate $\Sigma(x_{jy}\text{-mean})^2$ of j^{th} numeric value that belongs to a class *y* . The value x_{jy} is present in both the parties and mean is maintained by them as random shares. Hence computation is performed by using these random shares. Protocol 4 used for secure product computation. Since the mean is unknown to both the parties and maintained in the form of random shares, parties have to collaborate to compute the probability of the new instance belonging to a class.

Protocol 3: For a numeric attribute

Input: x_{jy} -> value of instance *j* having class value *y*.

S^{x}_{y} -> represents the sum of instances having class value *y*

N^{x}_{y} -> represents of instances having class value *y* in Party *x*

$h^{x}_{k}\ P^{x}_{k},\ V^{x}_{k}$-> kth random share maintained by Party *x*

For all class values *y* do

{Party A locally computes $S^{A}_{\ y} = \sum_{j} x_{\ jy}$ and N^{A}_{y}

Party B locally computes $S^{B}_{\ y} = \sum_{j} x_{\ jy}$ and N^{B}_{y}

Party A and B use algorithm 2 to jointly Compute $h^{A}_{1} + h^{B}_{1} = S^{A}_{y}/\ (N^{A}_{y} + N^{B}_{y})$ and

Party A and B use algorithm 2 to jointly Compute $h^{A}_{2} + h^{B}_{2} = S^{B}_{y}/\ (N^{B}_{y} + N^{A}_{y})$

Note: - Mean = $h^{A}_{1} + h^{B}_{1}+ h^{A}_{2} + h^{B}_{2}$.

To compute variance

Party A locally computes

For each of its instance *j*

1. $t_j^A = x_{jy}\text{-}(h^{A}_{1} + h^{A}_{2})/1000$.

2. $\text{temp}_j^A = (t_j^A)^2$ that belongs to the class y .

Party A computes $\text{Sum}^A = \Sigma_{j=1}^{NA} t_j^A$.

Party A and B use algorithm 4(secure product) to jointly compute $P^{A}_{1} + P^{B}_{1} = 2\times\text{Sum}_{A}\times((h^{B}_{1}+ h^{B}_{2})/1000)$.

Party A and B use algorithm 4(secure product) to jointly compute $V^{A}_{1} + V^{B}_{1} = N^{A}_{y}\times((h^{B}_{1}+ h^{B}_{2})/1000)$.

Party A locally computes $\text{Num}^A = \Sigma_{\ j=1}^{\ n} \text{temp}_j^A + P^{A}_{1} + V^{A}_{1}$.

Party B locally computes

For each of its instance *j*

1. $t_j^B = x_{jy}\text{-}(h^{B}_{1} + h^{B}_{2})/1000$.

2. $\text{temp}_j^B = (t_j^B)^2$ that belongs to the class *y*.

Party B computes $\text{Sum}^{\mathbf{B}} = \Sigma_{j=1}^{NB} t_j^B$.

Party A and B use algorithm 4(secure product) to jointly compute $P^{A}_{2} + P^{B}_{2} = 2\times\text{Sum}^{B}\times((h^{A}_{1}+ h^{A}_{2})/1000)$.

Party A and B use algorithm 4(secure product) to jointly compute $V^{A}_{2} + V^{B}_{2} = N^{B}_{y}\times((h^{A}_{1}+ h^{A}_{2})/1000)$.

Party B locally computes $\text{Num}^B = \Sigma_{\ j=1}^{\ n} \text{temp}_j^B + P^{B}_{2} + V^{B}_{2}$.

Party A obtains $\text{Var}^{A}_{y} = \text{Num}^A + V^{A}_{2} + P^{A}_{2}$.

Party B obtains $\text{Var}^{B}_{y} = \text{Num}^B + V^{B}_{1} + P^{B}_{1}$.

Party B performs $E(\text{Var}^{B}_{y})$ using Paillier encryption and send to Party A.

Party A performs $E(\text{Var}^{A}_{y})$ and then $\text{Var}_y = D(E(\text{Var}^{A}_{y}) \times E(\text{Var}^{B}_{y}))$.

Party A now has $N_y = N^{A}_{y}$-1. It computes $\text{Stan\_dev}^{2}_{y} = \text{Var}_y / (N_y + N^{B}_{y})$ using protocol 2.

Stan_dev^2 is forwarded to Party B.

}

To indicate the class to which the test/new tuple with a numeric attribute can be classified into is computed as follows:

For each of the class value I:

$$P\ (\text{given that (attribute\_value = test\_record\_numeric\_value)}\ |\text{Class}_i) = (1/(\text{sqrt}(2\times\pi)\times\text{Stan\_dev})\times\exp(-(\text{test\_record\_numeric\_value–Mean})/(2\times\text{Stan\_dev}(\text{Class}_i))$$

Party A computes $\text{Val}^A = -\ (h^{A}_{1}+ h^{A}_{2})/1000)/2\ \times \text{Stan\_dev}_y$.

Party B computes $\text{Val}^B = ((h^{B}_{1}+ h^{B}_{2})/1000)/\ 2\ \times \text{Stan\_dev}_y$ send Paillier encrypted $E(\text{Val}^B)$ to Party A.

Party A computes $\text{Val} = D(E(\text{Val}^A)\times E(\text{Val}^B))$, to further obtain

$$P\ (\text{given that (attribute\_value = test\_record\_numeric\_value)}|\text{Classy}) = 1/(\text{sqrt}(2\pi)\times\text{stan\_dev})\times\exp^{-\text{Val}}$$

Protocol 4 initiated by Party 1 having a value m_1, generates a random value and performs $m_1 \times p - R$ for a range of values indicated by p. Each of the p values obtained is encrypted and forwarded to Party 2 in the increasing order of p. Party 2 selects the m_2th encrypted value , further encrypt it and forward it to Party 1. Party 1 then decrypts the received value sends it to Party 2 which further decrypts it and retains the random share of the product.

Protocol 4: Securely computing the Product

1. m_1 at Party 1 is modified to $m_1 = m_1 \times 1000$ to

2. For p = range of n values

2.1. It then generates a random number R and computes the product $M = m_1 \times p - R$.

2.2. M is encrypted using RSA.

2.3. The encrypted result is forwarded to Party 2.

3. Party 2 also performs $m_2 = m_2 \times 1000$

3.1. Of the p encrypted values sent by Party 1, Party 2 selects the m_2th value.

3.2. It further encrypts the value (RSA) to get $E`$.

3.3. $E`$ is communicated to Party 1.

4. Party 1 decrypts using RSA to obtain $D`$.

5. Party 2 decrypts using RSA to obtain D.

6. Obtain the final result by performing $D = D/1000000$.

Note: random share r_1 at Party 1 is R and share at party 2 r_2 is $m_1 \times m_2 - R$ such that $r_1 + r_2 = m_1 \times m_2$.

III. Security Analysis

In semihonest model, the parties follow the protocol and may try to analyze the intermediate results obtained during protocol execution. To assure security we must assure that the parties learn nothing from the information they get doing the protocol process. A standard way to show this is to construct a simulator which can simulate what the party can see in the protocol given only the input and output of the protocol for this party. Since the privacy-preserving protocols 1 and 3 use protocol 2 and 4 as building blocks, we initially analyze the security of protocols 2 and 4 before performing a security analysis on protocols 1 and 3. Protocol 2 and 4 use the RSA-OAEP encryption which is semantically secure [17] where each ciphertext can be simulated by a random ciphertext.

The simulator in party 1 performs the computation and generates a random ciphertext. In steps 2 and 3, party 1 does what it is supposed to do in the protocol. In step 4, party 1 decrypts the encrypted value that it receives and sends it to party 2. The simulator for party 2 receives random ciphertexts in step 2. It selects one of the values and generates a random encryption of it in step 3. In step 5, it decrypts the received random ciphertext and does what it should do in the protocol.

Security in protocol 1 results in probability known to party 1 and total variance and standard deviation known to party 1 in protocol 3. In the training round, most of the message transmissions are taking place inside the calls of protocols 2 and 4. Paillier Homomorphic encryption scheme is used to finally compute the sum of the shares maintained at both parties. This encryption approach is also semantically secure as mentioned earlier .Simulators are used by both parties to generate h^A_{1i}, h^B_{1i}, h^A_{2i}, h^B_{2i} in protocol 1. P^A_2, P^B_2, V^A_2, V^B_2 in protocol 3.

IV. Analysis of Complexity and Accuracy Loss

A. Securely Computing Protocol 2 and Protocol 4

This protocol includes n encryptions, where n is the parameter of the function an encryption followed with decryptions from party 1 and 2. Hence the total computation cost is $T = (n+1) \times E_n + 2D_n$, where E_n and D_n are the cost of encryption and decryption.

Correspondingly, the total computation cost of Protocol 4 is also $(n+1) \times E_n + 2D_n$.

B. Computation Time for Training in Protocol 1

Party 1 and 2 locally perform 2 summations in parallel which results in a computation time $2 \times p$, where p is the time for summation. Protocol 2 is called twice resulting in $2 \times (n+1) \times E_n + 2D_n$. Party 1 and 2 encrypt their values using Paillier encryption followed with a decryption from party 1. Hence the computation time is $2p + 2 \times (n+1) \times E_n + 2D_n + 2\,E_p + D_p$ where E_p and D_p are the Paillier encryption and decryption time.

C. Computation Time for Training in Protocol 3

Party 1 and 2 locally perform 2 summations each in parallel resulting in a computation time of $2 \times p$ where p is the time for summation. Protocol 2 is called twice resulting in $2 \times (n+1) \times En + 2Dn$. Both parties locally perform a division, subtraction and squaring for every instance belonging to a class resulting in $j \times (D + S + Q)$, where D is the time for division and S is the computation time for subtraction, Q is the squaring time and j is the number of tuples belonging a class y. Both parties collaborate to execute secure product twice resulting in a computation time of $2(n+1)\,En + 2Dn$. After certain additions, encryptions and decryptions the total computation time of this protocol is $12p + 5 \times (n+1) \times En + 2Dn + j \times (D + S + Q) + 2 \times En + Dn$. where Ep and Dp are the Paillier encryption and decryption time.

1) Accuracy loss

Approximations in our algorithms for achieving privacy are used in two places. One is in protocol 2 and the other in protocol 4. The approximation is introduced by mapping the real numbers to fixed-point representations to enable cryptographic results and obtain results up to 3 decimal points. Suppose a system that uses the protocols uses v bits for representation of real numbers. Assuming that the v-bit number is chopped off by x-bits then the precision error ratio is bounded by Err = 2^{v-x}.

V. Performance Comparison

In this section, we compare our 2-party protocol with kantarcioglu-Vaidya [18] and Xun Yi-Yanchun Zhang [19] protocols. Both the protocols depend on the parties receiving and sending messages to and fro hence increasing the amount of communication between the parties. In our protocol parties communicate the encrypted data only while performing secure computations. Parties perform their local computations with the random values they have.

Protocol [18] is vulnerable to collusion attack as mentioned in [19]. Our protocol assumes no collusion among the two parties. kantarcioglu-Vaidya Protocol transmits protocol in almost natural form and hence is more susceptible to snooping attacks. Our protocol encrypts all transmissions from one site to another. Visibly, our protocol is more secure than kantarcioglu-Vaidya Protocol. Xun Yi-Yanchun Zhang [19] protocol does not present any protocols to handle numeric attributes. Supervised learning includes attributes that are categorical or numeric in nature hence Xun Yi-Yanchun Zhang [19] protocol may not be suitable. Also kantarcioglu-Vaidya Protocol does not work securely for only two parties.

For the hospital data set maintained at 2 sites. From the model obtained the accuracy of the privacy preserving classifiers was compared as shown in Fig. 1. The test instances considered had only categorical attributes. The results showed that Privacy Preserving Naive Bayesian Classifier using random shares (PPNBC-Random share) classifier.

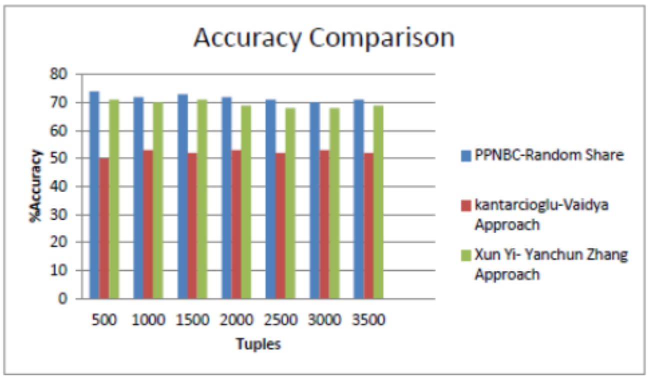

Fig. 1. Comparison of accuracy of the protocols.

VI. Conclusion

In this paper, we present a privacy preserving algorithm for Naive Bayesian learning. The protocols guarantee privacy using a paradigm cryptographic model. PPNB-Random Shares can be securely used on both categorical and numeric attributes with reasonable accuracy loss in the results obtained. Using these techniques we could further develop algorithms for distributed systems with three or more participants. In future, we would work on constructing different secure protocols for privacy preservation on diverse types of classifiers.

References

[1] R. Agrawal and R. Srikant, “Privacy-preserving data mining,” in *Proc. ACM SIGMOD International Conference on Management of Data*, 2000, vol. 29, pp. 439-450.

[2] C. C. Aggarwal, “On k-anonymity and the curse of dimensionality,” in *Proc. VLDB Conference*, 2005.

[3] A. Machanavajjhala, J. Gehrke, and D. Kifer, “ℓ-diversity: Privacy beyond k-anonymity,” in *Proc. IEEE ICDE Conference*, 2006.

[4] V. S. Verykios, A. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, “Association rule hiding,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 434-447, 2004.

[5] L. Liu, M. Kantarcioglu, and B. Thuraisingham, “Privacy preserving decision tree mining from perturbed data,” in *Proc. the 42nd Hawaii International Conference on System Sciences*, 2009.

[6] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen, “On private scalar product computation for privacy-preserving data mining,” in *Proc. 7th International Conference on Information Security and Cryptology*, 2005, pp. 104–120.

[7] Y. Lindel and B. Pinkas, “Privacy preserving data mining,” in *Proc. International Conference on Advances in Cryptology* (CRYPTO’00), 2000, pp. 36–53.

[8] G. R. Blakley, “Safeguarding cryptographic keys,” in *Proc. National Computer Conference*, June 1979, pp. 313–317.

[9] O. Goldreich, “The foundations of cryptography,” *Basic Applications*, vol. 2, May 2004.

[10] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes,” in *Proc. Advances in Cryptology EUROCRYPT’99*, 1999, pp. 223–238.

[11] J. Vaidya and C. Clifton, “Privacy-preserving k-means clustering over vertically partitioned data,” in *Proc. the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003, pp. 206–215.

[12] T. ElGamal, “A public key cryptosystem and a signature based scheme on discrete logarithms,” *IEEE Transactions on Inf. Theory*, vol. IT-31, no. 4, pp. 469-472, Jul. 1985.

[13] M. Blum and S. Goldwasser, “An efficient probabilistic public-key encryption that hides all partial information,” *Advances in Cryptography — Crypto 84 Proceedings*, 1984.

[14] S. Goldwasser and S. Micali, “Probabilistic encryption,” *Journal of Computational Systems*, vol. 28, no. 2, pp. 270–299, March 1984.

[15] B. Kaliski, “The mathematics of the RSA public-key cryptosystem,” RSA Laboratories, 2008.

[16] E. Milanov, “The RSA algorithm,” RSA Laboratories, 2009.

[17] RSAES-OAEP Encryption Scheme, algorithm specification and supporting documentation, RSA Laboratories, 2000.

[18] J. Vaidya, M. Kantarcioglu, and C. Clifton, “Privacy-preserving Na ïve Bayesian classification,” *VLDB Journal*, vol. 17, pp. 879-898, 2008.

[19] X. Yi and Y. Zang, “Privacy preserving naive Bayes classification on distributed data via semi-trusted mixers,” *Information Systems*, 2008.

K. S. Hareesha was born in Chikmagalore, Karnataka State, India on April 11, 1970. He received the BCA, MCA degrees in computer applications and Ph.D. degree in computer science & engineering from Kuvempu Universtiy, Shankaraghatta, Karnataka, India in 2000, 2003 and 2008 respectively. His major research interest is digital image processing, data mining, bioinformatics and artificial intelligence.

Since 2008, he is working in Manipal University, Karnataka, India as an associate professor, before to this he was worked in Bapuji Institute of Technology, Davangere, Karnataka, India. Also, he took charge as the head of Department of Computer Applications, Manipal Institute of Technology, Manipal University since Jan. 2013. His research interests encompass privacy preservations in data mining, spatial data mining and its relevance in society as a whole, bio-informatics, biologically inspired algorithms, soft computing and computer vision. He has published quite a good number of research papers in these areas. He has fellowship award from Boston University to present his research contributions.

Dr. K. S. Hareesha has been a life member of ISTE, New Delhi, India and IACSIT, Singapore. He has been a member of numerous program committee of IEEE, IACSIT conferences in the area of data mining, bioinformatics, digital image processing and artificial intelligence.

M. Sumana was born in Madikeri, Karnataka, India on December 15, 1978. She has received her B.E. degree in computer science and engineering from Manipal Institute of Technology, Karnataka, India, in 2000, and her M.Tech. degree from 3 VTU University in 2007, Karnataka, India and is currently pursuing the Ph.D. degree in privacy preserving data mining in computer science and engineering from the Manipal University, Karnataka, India.

She is presently working as an assistant professor in the Department of Information Science and Engineering in M. S. Ramaiah Institute of Technology since 2007. Previously she had worked as a lecturer in the Manipal Institute of Technology. Her research interests include data mining, cryptography and secure multiparty computations. She is a life member of the Indian Society for Technical Education (ISTE) and the System Society of India.