Privacy-Preserving Naive Bayesian Classification Over Vertically Partitioned Data

Justin Zhan[1], Stan Matwin[2], and LiWu Chang[3]
[1,2]School of Information Technology & Engineering, University of Ottawa, Canada
[3]Center for High Assurance Computer Systems, Naval Research Laboratory, USA
{zhizhan, stan}@site.uottawa.ca, lchang@itd.nrl.navy.mil

Abstract: Protection of privacy is a critical problem in data mining. Preserving data privacy in distributed data mining is even more challenging. In this paper, we consider the problem of privacy-preserving naive Bayesian classification over vertically partitioned data. The problem is one of important issues in privacy-preserving distributed data mining. Our approach is based on homomorphic encryption. The scheme is very efficient in the term of computation and communication cost.

Keywords: Privacy, security, naive Bayesian classification.

I. Introduction

Recent advances in computer networking and database technologies have resulted in creation of large quantities of data which are located in different sites. Data mining is a useful tool to extract valuable knowledge from this data. Well known data mining algorithms include association rule mining, classification, clustering , outlier detection, etc. However, extracting useful knowledge from distributed sites is often challenging due to real world constraints such as privacy, communication and computation overhead.

In this paper, we focus on privacy-preserving data mining in a distributed setting where the different sites have diverse sets of features. Specially, we consider the problem of privacy-preserving naive Bayesian classification that is one of the most successful algorithms in many classification domains. A Bayesian network is a high-level representation of a probability distribution over a set of variables that are used for constructing a model of the problem domain. It has been widely used in sales decision making, marketing systems, risk analysis, cost benefit factor inference in E-services, and other business applications. A Bayesian network have many applications. For instance it can be used to compute the predictive distribution on effects of possible actions since it is a model of the problem domain probability distribution.

A naive Bayesian classifier is one of Bayesian classifiers under conditional independence assumption of different features. Over the last decade, the naive Bayesian classification has been widely utilized. Although the techniques that have been developed are effective, new techniques dealing with naive Bayesian classification over private data are required. In other words, we need methods to learn a naive Bayesian classifier over distributed private data. In this paper, we develop a novel scheme based on homomorphic encryption without compromising data privacy.

II. Related Work

In early work on privacy-preserving data mining, Lindell and Pinkas [13] propose a solution to privacy-preserving classification problem using oblivious transfer protocol, a powerful tool developed by secure multi-party computation (SMC) research [21, 10]. The techniques based on SMC for efficiently dealing with large data sets have been addressed in [20]. Randomization approaches were firstly proposed by Agrawal and Srikant in [2] to solve privacy-preserving data mining problem. Researchers proposed more random perturbation-based techniques to tackle the problems (e.g., [5, 18, 7]). In addition to perturbation, aggregation of data values [19] provides another alternative to mask the actual data values. In [1], authors studied the problem of computing the kth-ranked element. Dwork and Nissim [6] showed how to learn certain types of boolean functions from statistical databases in terms of a measure of probability difference with respect to probabilistic implication, where data are perturbed with noise for the release of statistics. The problem we are studying is actually a special case of a more general problem, the Secure Multi-party Computation (SMC) problem. Briefly, a SMC problem deals with computing any function on any input, in a distributed network where each participant holds one of the inputs, while ensuring that no more information is revealed to a participant in the computation than can be inferred from

Proceedings of the Fifth International Conference on Electronic Business, Hong Kong, December 5-9, 2005, pp. 483 - 488.

that participant's input and output [12]. The SMC problem literature is extensive, having been introduced by Yao [21] and expanded by Goldreich, Micali, and Wigderson [11] and others [8]. It has been proved that for any function, there is a secure multi-party computation solution [10]. The approach used is as follows: the function F to be computed is first represented as a combinatorial circuit, and then the parties run a short protocol for every gate in the circuit. Every participant gets corresponding shares of the input wires and the output wires for every gate. This approach, though appealing in its generality and simplicity, means that the size of the protocol depends on the size of the circuit, which depends on the size of the input. This is highly inefficient for large inputs, as in data mining. It has been well accepted that for special cases of computations, special solutions should be developed for efficiency reasons.

III. Building Naive Bayesian Classifiers

III.1 Notations

- e: public key.
- d: private key.
- P_i: the ith party.
- n: the total number of parties. Assuming $n > 2$.
- m: the total number of class.
- α is the number of bits for each transmitted element in the privacy-preserving protocols.
- N: the total number of records.

III.2 Cryptography Tools

Our scheme is based on homomorphic encryption which was originally proposed in [17]. Since then, many such systems have been proposed [3, 14, 15, 16]. We observe that some homomorphic encryption schemes, such as [4], are not robust against chosen cleartext attacks. However, we base our secure protocols on [16], which is semantically secure [9].

In our secure protocols, we use additive homomorphism offered by [16]. In particular, we utilize the following characterizer of the homomorphic encryption functions: $e(a_1) \times e(a_2) = e(a_1 + a_2)$ where e is an encryption function; a_1 and a_2 are the data to be encrypted. Because of the property of associativity, $e(a_1 + a_2 + .. + a_n)$ can be computed as $e(a_1) \times e(a_2) \times \cdots \times e(a_n)$ where $e(a_i) \neq 0$. That is

$$d(e(a_1 + a_2 + \cdots + a_n)) = d(e(a_1) \times e(a_2) \times \cdots \times e(a_n)) \quad (1)$$

$$d(e(a_1)^{a_2}) = d(e(a_1 a_2)) \quad (2)$$

III.3 Introducing Naive Bayesian Classification

The naive Bayesian classification is one of the most successful algorithms in many classification domains. Despite of its simplicity, it is shown to be competitive with other complex approaches, especially in text categorization and content based filtering. The naive Bayesian classifier applies to learning tasks where each instance x is described by a conjunction of attribute values and where the target function f(x) can take on any value from some finite set V. A set of training examples of the target function is provided, and a new instance is presented, described by the tuple of attribute values $< a_1, a_2, \cdots, a_n >$. The learner is asked to predict the target value for this new instance. Under a conditional independence assumption, i.e., $Pr(a_1, a_2, \cdots, a_n | v_j) = \prod_{i=1}^{n} Pr(a_i | v_j)$, a naive Bayesian classifier can be derived as follows:

$$V_{NB} = argmax_{v_j \in V} Pr(v_j) \prod_{i=1}^{n} Pr(a_i | v_j)$$

$$= argmax_{v_j \in V} Pr(v_j) \prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)}$$

To build a NB classifier, we need to compute $Pr(v_j)$ and $Pr(a_i, v_j)$. $Pr(v_j)$ can be computed by the owner of v_j without breaching privacy. To compute $Pr(a_i, v_j)$, we need consider two aspects: (1) All the parties share the class label; (2) The class label is hosted by only one party. For the first case, $Pr(a_i, v_j)$ can be calculated by the owner of v_j without breaching privacy. For the second case, if the owner of v_j also owns the class label, he can compute $Pr(a_i, v_j)$. However, when a_i and v_j belong to different parties, the computation of $Pr(a_i, v_j)$ is challenging.

In this paper, we will design a privacy-preserving system to show how to compute a naive Bayesian classifier. The goal of our privacy-preserving classification system is to disclose no private data in every step. We firstly select a key generator who produces the encryption and decryption key pairs. The computation of the whole system is under encryption. For the purpose of illustration, let's assume that P_n is the key generator who generates a homomorphic encryption key pair (e, d). The key generator should not host the class label. For the purpose of illustration, let's assume that P_1 holds the class label. Next, we will show how to conduct each step.

III.4 Privacy-Preserving Naive Bayesian Classification

III.4.1 To Compute $e(Pr(a_i, v_j))$

Protocol 1 .

1. P_l for $l \in [2, n]$ performs the following operations:
   (a) She computes $e(a_{ik})$s ($k \in [1, N]$) and sends them to P_1.
2. P_1 performs the following operations:
   (a) He computes $t_1 = e(a_{i1})^{v_{j1}} = e(a_{i1} \cdot v_{j1})$, $t_2 = e(a_{i2})^{v_{j2}} = e(a_{i2} \cdot v_{j2})$, $\cdots$, $t_N = e(a_{iN})^{v_{jN}} = e(a_{iN} \cdot v_{jN})$.
   (b) He computes $t_1 \times t_2 \times \cdots \times t_N = e(a_{i1} \cdot v_{j1} + a_{i2} \cdot v_{j2} + \cdots + a_{iN} \cdot v_{jN}) = e(\overrightarrow{a_i} \cdot \overrightarrow{v_j})$.
   (c) He computes $e(\overrightarrow{a_i} \cdot \overrightarrow{v_j})^{\frac{1}{N}} = e(\frac{(\overrightarrow{a_i} \cdot \overrightarrow{v_j})}{N}) = e(Pr(a_i, v_j))$.

Theorem 1 *(Correctness). Protocol 1 correctly computes* $e(Pr(a_i, v_j))$.

Proof When P_1 receives $e(a_{ik})(k \in [1, N])$, he computes $t_k = e(a_{ik})^{v_{jk}}$. According to Eq. 2, it is equal to $e(a_{ik}v_{jk})$ for $k \in [1, N]$. He then computes $\prod_{k=1}^{N} t_k$ which is equal to $e(\overrightarrow{a_i} \cdot \overrightarrow{v_j})$ according to Eq. 1. Finally, he computes $e(\overrightarrow{a_i} \cdot \overrightarrow{v_j})^{\frac{1}{N}} = e(Pr(a_i, v_j))$. Therefore, the protocol 1 correctly computes $e(Pr(a_i, v_j))$. ∎

Theorem 2 *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof In protocol 1, all the data transmission are hidden under encryption. The parties who are not the key generator can't see other parties' private data. On the other hand, the key generator doesn't obtain the encryption of other parties' private data. Therefore, protocol 1 discloses no private data. ∎

Theorem 3 *(Efficiency). Protocol 1 is efficient in terms of computation and communication complexity.*

Proof To prove the efficiency, we need conduct complexity analysis of the protocol. The bit-wise communication cost of this protocol is αmN. The computation cost is upper bounded by $(m+2)N$. Therefore, the protocol is sufficient fast. ∎

Via the above protocol, the class holder P_1 gets $e(Pr(a_i, v_j))(P_l)$ for $l \in [2, n]$ and $e(Pr(a_i, v_j))(P_l)$ denotes $e(Pr(a_i, v_j))$ for lth party. Next, we will show how to construct a NB classifier.

III.4.2 To Compute $e(Pr(v_j) \prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)})$ for each $v_j \in V$

Protocol 2 .

1. *P_1 generates a set of random numbers, r_2, r_3, $\cdots$, and r_n.*
2. *P_1 computes $e(Pr(a_i, v_j)(P_2)) \times e(r_2) = e(Pr(a_i, v_j)(P_2) + r_2)$, $e(Pr(a_i, v_j)(P_3)) \times e(r_3) = e(Pr(a_i, v_j)(P_3) + r_3)$, $\cdots$, $e(Pr(a_i, v_j)(P_n)) \times e(r_n) = e(Pr(a_i, v_j)(P_n) + r_n)$.*
3. *P_1 sends $e(Pr(a_i, v_j)(P_l) + r_l)$ to P_n where $l \in [2, n]$.*
4. *P_n decrypts them and obtains $Pr(a_i, v_j)(P_l) + r_l$ for $l \in [2, n]$.*
5. *P_n sends $Pr(a_i, v_j)(P_l) + r_l$ to P_l for $l \in [2, n-1]$.*
6. *P_1 computes $e(\prod_{v_j \in P_1} \frac{Pr(a_i, v_j)}{Pr(v_j)})$ for $a_i \in P_1$. Let's denote it by $e(G_1)$. P_1 sends $e(G_1)$ to P_2.*
7. *P_2 computes $e(G_1)^{(Pr(a_i, v_j)(P_2) + r_2)} = e((Pr(a_i + v_j)(P_2) + r_2) \times G_1)$ and sends it to P_1.*
8. *P_1 computes $e((Pr(a_i + v_j)(P_2) + r_2) \times G_1) \times e(-r_2 G_1) = e(\prod_{a_i \in P_1, P_2} \frac{Pr(a_i, v_j)}{Pr(v_j)})$ denoted by $e(G_2)$.*
9. *Continue until P_1 gets $e(\prod_{a_i \in P_1, P_2, \cdots, P_n} \frac{Pr(a_i, v_j)}{Pr(v_j)}) = e(\prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)})$.*
10. *P_1 computes $e(\prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)})^{Pr(v_j)} = e(Pr(v_j) \prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)})$.*

Theorem 4 *(Correctness). Protocol 2 correctly computes* $e(Pr(a_i, v_j))$.

Proof In step 3, P_n obtains $e(Pr(a_i, v_j)(P_l) + r_l)$ for $l \in [2, n]$. In step 6, P_1 computes $e(G_1) = e(\prod_{v_j \in P_1} \frac{Pr(a_i, v_j)}{Pr(v_j)})$ for $a_i \in P_1$ based on his own data. In step 9, P_1 obtains $e(G_2) = e((Pr(a_i + v_j)(P_2) + r_2) \times G_1) \times e(-r_2 G_1) = e(\prod_{a_i \in P_1, P_2} \frac{Pr(a_i, v_j)}{Pr(v_j)})$ according Eq. 1 and Eq. 2. Finally, P_1 obtains $e(\prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)})^{Pr(v_j)} = e(Pr(v_j) \prod_{i=1}^{n} \frac{Pr(a_i, v_j)}{Pr(v_j)})$ according to Eq. 2. ∎

Theorem 5 *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof In protocol 2, all the data transmission, among the parties who have no decryption key, are hidden under encryption. Therefore, these parties cannot know the private data. In step 4, P_n obtains $Pr(a_i, v_j)(P_l) + r_l$ for

$l \in [2, n]$. Since it is hidden by a random number r_l known only by P_1. P_n cannot get $Pr(a_i, v_j)(P_l)$. Therefore, protocol 2 discloses no private data.

∎

Theorem 6 *(Efficiency). Protocol 2 is efficient in terms of computation and communication complexity.*

Proof The bit-wise communication cost of this protocol is upper bounded by $4n\alpha$. The computation cost is upper bounded by $8n$. Therefore, the protocol is sufficient fast.

∎

Through the above protocol, $e(Pr(v_j)\prod_{i=1}^{n}\frac{Pr(a_i,v_j)}{Pr(v_j)})$ can be computed for each $v_j \in V$. Without loss of generality, let's assume P_1 gets $e(V_{NB_1})$, $e(V_{NB_2})$, $\cdots$, $e(V_{NB_k})$ The goal is to find the largest one.

III.4.3 To Compute V_{NB}

Protocol 3 .

1. *P_1 computes $e(V_{NB_i})\times e(V_{NB_j})^{-1} = e(V_{NB_i} - V_{NB_j})$ for all $i, j \in [1, k], i > j$, and sends the sequence denoted by φ to P_n in a random order.*
2. *P_n decrypts each element in the sequence φ. He assigns the element $+1$ if the result of decryption is not less than 0, and -1, otherwise. Finally, he obtains a $+1/-1$ sequence denoted by φ'.*
3. *P_n sends $+1/-1$ sequence φ' to P_1 who computes the largest element.*

Theorem 7 *(Correctness). Protocol 3 correctly computes V_{NB}.*

Proof P_1 is able to remove permutation effects from φ' (the resultant sequence is denoted by φ'') since she has the permutation function that she used to permute φ, so that the elements in φ and φ'' have the same order. It means that if the qth position in sequence φ denotes $e(V_{NB_i} - V_{NB_j})$, then the qth position in sequence φ'' denotes the evaluation results of $V_{NB_i} - V_{NB_j}$. We encode it as $+1$ if $V_{NB_i} \geq V_{NB_j}$, and as -1 otherwise. P_1 has two sequences: one is the φ, the sequence of $e(V_{NB_i} - V_{NB_j})$, for $i, j \in [1, k](i > j)$, and the other is φ'', the sequence of $+1/-1$. The two sequences have the same number of elements. P_1 knows whether or not V_{NB_i} is larger than V_{NB_j} by checking the corresponding value in the φ'' sequence. For example, if the first element φ'' is -1, P_1 concludes $V_{NB_i} < V_{NB_j}$. P_1 examines the two sequences and constructs the index table (Table 1) to compute the largest element.

In Table 1, $+1$ in entry ij indicates that the information gain of the row (e.g., V_{NB_i} of the ith row) is not less than the information gain of a column (e.g., V_{NB_j} of the jth column); -1, otherwise. P_1 sums the index values of each row and uses this number as the weight of the information gain in that row. She then selects the one that corresponds to the largest weight.

	V_{NB_1}	V_{NB_2}	V_{NB_3}	$\cdots$	V_{NB_k}
V_{NB_1}	+1	+1	-1	$\cdots$	-1
V_{NB_2}	-1	+1	-1	$\cdots$	-1
V_{NB_3}	+1	+1	+1	$\cdots$	+1
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
V_{NB_k}	+1	+1	-1	$\cdots$	+1

Table 1:

	S_1	S_2	S_3	S_4	Weight
S_1	+1	-1	-1	-1	-2
S_2	+1	+1	-1	+1	+2
S_3	+1	+1	+1	+1	+4
S_4	+1	-1	-1	+1	0

Table 2:

To make it clearer, let's illustrate it by an example. Assume that: (1) there are 4 information gains with $V_{NB_1} < V_{NB_4} < V_{NB_2} < V_{NB_3}$; (2) the sequence φ is $[e(V_{NB_1} - V_{NB_2}), e(V_{NB_1} - V_{NB_3}), e(V_{NB_1} - V_{NB_4}), e(V_{NB_2} - V_{NB_3}), e(V_{NB_2} - V_{NB_4}), e(V_{NB_3} - V_{NB_4})]$. The sequence φ'' will be $[-1, -1, -1, -1, +1, +1]$. According to φ and φ'', P_1 builds the Table 2. From the table, P_1 knows V_{NB_3} is the largest element since its weight, which is $+4$, is the largest.

∎

Theorem 8 *(Privacy-Preserving). Assuming the parties follow the protocol, the private data are securely protected.*

Proof In protocol 3, we need prove it from two aspects: (1) P_1 doesn't get information gain (e.g.,V_{NB_i}) for each attribute. What P_1 gets are $e(V_{NB_i} - V_{NB_j})$ for all $i, j \in [1, k], i > j$ and $+1/-1$ sequence. By $e(V_{NB_i} - V_{NB_j})$, P_1 cannot know each information gain since it is encrypted. By $+1/-1$ sequence, P_1 can only know whether or not V_{NB_i} is greater than P_j. (2) P_n doesn't obtain information gain for each attribute either. Since the sequence of $e(V_{NB_i} - V_{NB_j})$ is randomized before being send to P_n who can only know the sequence of $V_{NB_i} - V_{NB_j}$, he can't get each individual information gain. Thus private data are not revealed. ∎

Theorem 9 *(Efficiency). The computation of protocol 3 is efficient from both computation and communication point of view.*

Proof The total communication cost is upper bounded by αm^2. The total computation cost is upper bounded by $m^2 + m + 1$. Therefore, the protocols are very fast. ∎

IV. Overall Discussion

Our privacy-preserving classification system contains several components. In Section , we show how to correctly compute $e(Pr(a_i, v_j)))$. In Section , we present protocols to compute $e(Pr(v_j) \prod_{i=1}^{n} \frac{Pr(a_i,v_j)}{Pr(v_j)})$ for each $v_j \in V$. In Section , we show how to compute the final naive Bayesian classifier. We discussed the correctness of the computation in each section. Overall correctness is also guaranteed.

As for the privacy protection, all the communications between the parties are encrypted, therefore, the parties who has no decryption key cannot gain anything out of the communication. On the other hand, there are some communication between the key generator and other parties. Although the communications are still encrypted, the key generator may gain some useful information. However, we guarantee that the key generator cannot gain the private data by adding random numbers in the original encrypted data so that even if the key generator get the intermediate results, there is little possibility that he can know the intermediate results. Therefore, the private data are securely protected with overwhelming probability.

In conclusion, we provide a novel solution for naive Bayesian classification over vertically partitioned private data. Instead of using data transformation, we define a protocol using homomorphic encryption to exchange the data while keeping it private. Our classification system is quite efficient that can be envisioned by the communication and computation complexity. The total communication complexity is upper bounded by $\alpha(mN+m^2+4n)$. The computation complexity is upper bounded by $mN + 2N + 8n + m^2 + m + 1$.

References

[1] G. Aggarwal, N. Mishra, and B. Pinkas. Secure computation of the k th-ranked element. In *EUROCRYPT pp 40-55*, 2004.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, May 2000.

[3] J. Benaloh. Dense probabilistic encryption. In *Proceedings of the Workshop on Selected Areas of Cryptography, pp. 120-128, Kingston, Ontario, May*, 1994.

[4] J. Domingo-Ferrer. A provably secure additive and multiplicative privacy homomorphism. In *Information Security Conference, 471-483*, 2002.

[5] W. Du and Z. Zhan. Using randomized response techniques for privacy-preserving data mining. In *Proceedings of The 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 24-27 2003.

[6] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO 2004 528–544*.

[7] A. Evfmievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the Twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, pages 211-222, San Diego, CA, June 9-12*, 2003.

[8] M. Franklin, Z. Galil, and M. Yung. An overview of secure distributed computing. Technical Report TR CUCS-00892, Department of Computer Science, Columbia University, 1992.

[9] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen. On secure scalar product computation for privacy-preserving data mining. In *Proceedings of The 7th Annual International Conference in Information Security and Cryptology (ICISC 2004), volume 3506 of Lecture Notes in Computer Science, pages 104–120, Seoul, Korea, December 2–3, 2004, Springer-Verlag*, 2004.

[10] O. Goldreich. Secure multi-party computation (working draft). http://www.wisdom.weizmann.ac.il /home/oded/public_html/foc.html, 1998.

[11] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, 1987.

[12] S. Goldwasser. Multi-party computations: Past and present. In *Proceedings of the 16th Annual ACM Symposium on Principles of Distributed Computing*, Santa Barbara, CA USA, August 21-24 1997.

[13] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Advances in Cryptology - Crypto2000, Lecture Notes in Computer Science, Volume 1880*, 2000.

[14] D. Naccache and J. Stern. A new public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM conference on Computer and Communication Security, pp. 59-66, San Francisco, California, United States*, 1998.

[15] T. Okamoto and S. Uchiyama. A new public-key cryptosystem as secure as factoring. In *Eurocrypt'98, LNCS 1403, pp.308-318*, 1998.

[16] P. Paillier. Public key cryptosystems based on composite degree residuosity classes. In *In Advances in Cryptology - Eurocrypt '99 Proceedings, LNCS 1592, pages 223-238. Springer-Verlag*, 1999.

[17] R. Rivest, L. Adleman, and M. Dertouzos. On data banks and privacy homomorphisms. In *Foundations of Secure Computation, eds. R. A. DeMillo et al., Academic Press, pp. 169-179.*, 1978.

[18] Shariq Rizvi and Jayant R. Haritsa. Maintaining data privacy in association rule mining. In *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.

[19] L. Sweeney. k-anonymity: a model for protecting privacy. In *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10 (5), pp 557–570*, 2002.

[20] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 639-644, Edmonton, Alberta, Canada, July 23-26*, 2002.

[21] A. C. Yao. Protocols for secure computations. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, 1982.