

# Privacy-Preserving Classification of Customer Data without Loss of Accuracy\*

Zhiqiang Yang<sup>1</sup> Sheng Zhong<sup>1,2</sup> Rebecca N. Wright<sup>1</sup>

<sup>1</sup>Computer Science Department, Stevens Institute of Technology, Hoboken, NJ 07030

<sup>2</sup>DIMACS Center, Rutgers University, Piscataway, NJ 08854

## Abstract

Privacy has become an increasingly important issue in data mining. In this paper, we consider a scenario in which a data miner surveys a *large* number of customers to learn classification rules on their data, while the sensitive attributes of these customers need to be protected. Solutions have been proposed to address this problem using randomization techniques. Such solutions exhibit a tradeoff of accuracy and privacy: the more each customer's private information is protected, the less accurate result the miner obtains; conversely, the more accurate the result, the less privacy for the customers.

In this paper, we propose a simple cryptographic approach that is efficient even in a many-customer setting, provides *strong privacy* for each customer, and *does not lose any accuracy as the cost of privacy*. Our key technical contribution is a privacy-preserving method that allows a data miner to compute frequencies of values or tuples of values in the customers' data, without revealing the privacy-sensitive part of the data. Unlike general-purpose cryptographic protocols, this method requires no interaction between customers, and each customer only needs to send a single flow of communication to the data miner. However, we are still able to ensure that nothing about the sensitive data beyond the desired frequencies is revealed to the data miner.

To illustrate the power of our approach, we use our frequency mining computation to obtain a privacy-preserving naive Bayes classifier learning algorithm. Initial experimental results demonstrate the practical efficiency of our solution. We also suggest some other applications of privacy-preserving frequency mining.

## 1 Introduction

The rapid growth of the Internet makes it easier than ever to collect data on a large scale. Data mining, with its promise to efficiently discover valuable knowledge

from vast amounts of data, is playing an increasingly important role in the current world. However, data mining, given its power in knowledge discovery, can be misused to violate people's privacy if proper measures are not taken. Privacy concerns have become one of the top priorities in technical, business, and political discussions of data mining. Due to these concerns, when customer data is collected from a large population, many people may decide to give false information, or simply decline to provide any information. Applying data-analysis tools and knowledge-discovery tools to data collected in this manner can therefore produce results with unacceptably low accuracy. This is in spite of the fact that, assuming privacy is properly protected, many people say they are willing to provide correct information in exchange for certain benefits the miner gives based on the result of mining [Wes99]. For example, a survey conducted to understand Internet customers' attitudes towards privacy showed that only 27% respondents say they are willing to provide their data without protection of privacy [Cra99]. Consequently, providing privacy protection measures may be critical to the success of data collection, and to the success of the entire task of data mining.

In this paper, we consider a "fully distributed" setting, in which each of many customers hold their own data. Existing solutions [AS00, AA01, ESAG02, EGS03, RH02, DZ03] to the privacy problem in this setting depend on randomization of each customer's data, which induces a tradeoff between privacy and accuracy: the more privacy of each customer has, the more accuracy the miner loses in his result. While in some cases the tradeoff may simultaneously offer sufficient privacy and sufficient accuracy, it is more desirable to have solutions that are both fully private and fully accurate. It has been demonstrated that in many cases random data distortion preserves very little privacy [KDWS03]. Related work on confidentiality in statistical databases [AW89, DN03, DN04] and random response techniques [War65, AJL04] also give nice dis-

---

\*This work was supported by the National Science Foundation under grant number CCR-0331584.

cussions of randomization of data.

In contrast, cryptographic solutions to privacy-preserving data mining that provide strong privacy have been proposed [LP02, VC02, VC03, KC02, WY04], but these tend to do so at a high performance cost. In particular, in order to have efficiency that is reasonable (say, linear in the size of the “virtual” database), these solutions require the data to be shared among only a small number of parties, often just two parties. In addition, these solutions typically require multiple rounds of communication between the participants.

In this work, we consider the fully distributed setting described above. Each customer maintains her own data. This can be thought of a horizontally partitioned database in which each transaction is owned by a different customer. A data miner wishes to compute some data mining results based on the customer data. In some applications, the data miner may also have some additional information about each customer. Note that the fully distributed setting can inherently provide more customer privacy than a horizontal partition into a small number of partitions, as each customer retains full control of her data in the fully distributed setting.

We propose a new approach that provides cryptographically strong privacy for each customer, but does not lose any accuracy as a cost of privacy, and is very efficient even in the fully distributed setting. In particular, we consider a scenario in which a miner surveys a large number of customers to learn classification rules by mining the set of collected data. In this scenario, we assume each customer is willing to provide her data to the miner as long as the privacy-sensitive part of her data is protected. By adding the efficient cryptographic operations we propose, we can prove that each customer’s privacy is protected in a strong cryptographic sense.

**1.1 Related Work** We note that our scenario is similar to the scenario of electronic voting [BT94]. However, e-voting systems usually assume that the voting authority consists of a group of servers that are threshold-trusted, or that another authority independent from the voting authority also participates in the protocol. Both of these possibilities are not justifiable in our scenario, however. Another difference is that in our setting, additional specialized concerns in e-voting such as receipt-freedom of the protocol are not an issue.

We also note that [AJL04] uses cryptography in the randomization setting to ensure that participants properly follow their specified instructions about randomization. However, their solution still has a privacy/accuracy tradeoff; in contrast, we use cryptography to “break” the privacy/accuracy tradeoff. Yet another piece of related work is [KV02], which proposes

a general architecture for privacy-preserving data mining. But their approach needs multiple service providers that do not collude with each other, which may not be available in practice.

In the context of privacy-preserving data mining, recent work gives privacy-preserving solutions for mining a naive Bayes classifier across a database horizontally or vertically partitioned into a small number of partitions [KV03, VC04]. In contrast, our solution provides a privacy-preserving method for mining a naive Bayes classifier in a fully distributed setting where each customer holds one record.

**1.2 Our Contribution** Our key technical contribution is a privacy-preserving method that allows a data miner to compute frequencies of values or tuples of values in the customers’ data, without revealing the privacy-sensitive part of the data. Technically, this can be reduced to a problem of securely summing private inputs and thus in theory it can be solved by either a general-purpose protocol for secure multi-party computation, e.g. [Yao86, GMW87, BGW88], or a special-purpose protocol for the secure sum problem, e.g. [Sch96, Chapter 6]. However, our solution has a number of advantages over the existing general-purpose protocols:

- Our solution is very efficient, while the existing general-purpose protocols are prohibitively expensive.
- Our solution does not assume any communication channels between customers and so is superior in the fully distributed scenario, where it would be infeasible to require such channels.
- In our solution, only one round of interaction is needed between the miner and each customer. This brings great convenience to the data mining task, since the customers can “submit data and go.”

To illustrate the power of our proposed approach, we take naive Bayes classification as an example and enable a privacy-preserving learning algorithm to protect customers’ privacy using our privacy-preserving frequency mining computation. We also suggest other privacy-preserving algorithms that are enabled by joint frequencies, such as decision trees and association rule mining.

Cryptographic techniques are often dismissed as being too expensive to use in practical data mining applications. Indeed, performance measured from implementations of cryptographic data mining protocols have often corroborated this impression [SWY04]. However, some cryptographic solutions are very efficient, and

their performance may be good enough to be useful in practice. We have implemented our privacy-preserving frequency mining algorithm and our privacy-preserving naive Bayes classifier learning algorithm, and show that both have very reasonable overhead.

We present our privacy-preserving joint frequency algorithm, including experimental results, in Section 2. We apply joint frequencies to naive Bayes classification in Section 3, again including experimental results. We discuss the application of the frequency algorithm to decision trees and association rule mining in Section 4, and we conclude in Section 5.

## 2 Privacy-Preserving Frequency Mining

This section describes a privacy-preserving primitive for frequency mining. In Section 2.1, we formulate the problem of frequency mining. We state our privacy definition in Section 2.2, and present our privacy-preserving protocol for frequency mining in Section 2.3. We give a security proof for this protocol in Section 2.4 and provide experimental results in Section 2.5.

**2.1 Problem Formulation** We consider a very basic problem: there are  $n$  customers  $U_1, \dots, U_n$ ; each customer  $U_i$  has a Boolean value  $d_i$ ; the miner would like to find out how many  $d_i$ 's are 1's and how many are 0's.

This problem essentially amounts to computing the sum  $d = \sum_{i=1}^n d_i$  without revealing each  $d_i$ . However, there are a few important restrictions:

- Each customer only sends one flow of communication to the miner; there is no further interaction between the customer and the miner.
- No customer communicates with other customers.

We call this the *reduced interaction* model. Solutions in this model are highly practical because they do not need communication channels between different customers or multi-round interaction between any customer and the miner.

In addition, we require that each customer's  $d_i$  is protected as defined in Section 2.2.

**2.2 Definition of Privacy** In the context of the randomization approach of protecting data privacy, there are two approaches to quantify the privacy-preserving property of a randomization methods. One approach relies on *information theory* [AA01], the other approach is based on the notion of *privacy breaches* [ESAG02, EGS03]. In the context of the cryptographic approach, the definition of privacy can be derived from the general definition of security in multi-party computations [Gol04]. Our definition of privacy given below, for example, can be viewed as a simplification of

the general definition in the *semi-honest* model, where the simplification results from our reduced interaction model. In our definition, we consider the possibility that some corrupted customers might share their information with the miner in order to help derive the private information of honest customers. We require that no extra information about the honest customers' values be leaked even if the miner above receives such help from corrupted customers. In the following definition, we do not consider problem of customers sharing their information with each other since, as we discuss after the definition, this will not give them any additional information in the reduced interaction model.

**DEFINITION 1.** Assume that each customer  $U_i$  has private keys  $x_i, y_i$  and public keys  $X_i, Y_i$ . A protocol for the above defined mining problem protects each customer's privacy against the miner and  $t$  corrupted users in the semi-honest model if,  $\forall I \subseteq \{1, \dots, n\}$  such that  $|I| = t$ , there exists a probabilistic polynomial-time algorithm  $M$  such that

$$(1) \quad \{M(d, [d_i, x_i, y_i]_{i \in I}, [X_j, Y_j]_{j \notin I})\} \stackrel{c}{=} \{\text{view}_{\text{miner}, \{U_i\}_{i \in I}}([d_i, x_i, y_i]_{i=1}^n)\}.$$

Here,  $\stackrel{c}{=}$  denotes *computational indistinguishability* (see a standard book of cryptography, e.g., [Gol04], for a definition), and  $\{\text{view}_{\text{miner}, \{U_i\}_{i \in I}}([d_i, x_i, y_i]_{i=1}^n)\}$  is the joint *view* (again, see [Gol04] for a precise definition) of the miner and the  $t$  corrupted customers,  $\{U_i\}_{i \in I}$ . Intuitively, this definition states that a polynomial-time algorithm  $M$ , called a *simulator*, can simulate what the miner and the corrupted customers have observed in the protocol using only the final result  $d$ , the corrupted users' knowledge, and the public keys. Therefore, the miner and the corrupted customers jointly learn nothing beyond  $d$ .

Definition 1 only addresses privacy in the semi-honest model [Gol04] (which assumes that all parties follow the protocol). However, in our reduced interaction model, a protocol protecting customer privacy in the semi-honest model actually also protects customer privacy even when the miner and the corrupted customers are fully malicious (i.e., may deviate arbitrarily from the protocol). This is because these malicious parties cannot have any influence on the honest customers due to the restrictions of the reduced interaction model. In this sense, our solution provides privacy against malicious parties "for free". We note, however, that the correct completion of the protocol cannot be guaranteed with malicious parties, as they may send garbage or refuse to send anything at all.

**2.3 Protocol** Our protocol design is based on the additively homomorphic property of a variant of ElGamal encryption, which has been used in, e.g., [HS00]. The privacy of our protocol is based on the believed computational difficulty of the *discrete logarithm* problem, and the related ElGamal cryptosystem, which we will describe in more detail in Section 2.4. The protocol itself uses the mathematical properties of exponentiation, which allows the miner to combine encrypted results received from the customers into the desired sums.

Let  $G$  be a group in which discrete logarithm is hard, and let  $g$  be a generator of  $G$ . All computations in this section are carried out in the group  $G$ . Suppose that each customer  $U_i$  has two pairs of keys:  $(x_i, X_i = g^{x_i}), (y_i, Y_i = g^{y_i})$ . Define

$$(2) \quad X = \prod_{i=1}^n X_i$$

$$(3) \quad Y = \prod_{i=1}^n Y_i$$

The values  $x_i$  and  $y_i$  are private keys (i.e., each  $x_i$  and  $y_i$  is known only to customer  $U_i$ );  $X_i$  and  $Y_i$  are public keys (i.e., they can be publicly known). In particular, the protocol requires that all customers know the values  $X$  and  $Y$ . In addition, each customer knows the group  $G$  and the common generator  $g$ .

Recall that each customer  $U_i$  holds the Boolean value  $d_i$ , and the miner's goal is to learn  $d = \sum_{i=1}^n d_i$ . The privacy-preserving protocol for the miner to learn  $d$  is detailed in Figure 1.

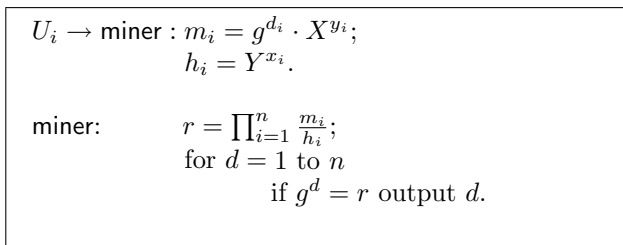


Figure 1: Privacy-Preserving Protocol for Frequency Mining.

**THEOREM 2.1.** *The protocol for frequency mining presented in Figure 1 correctly computes the sum of all customers' inputs.*

**Proof:** We show that, in the protocol, when the miner finds  $g^d = r$ , the value  $d$  is the desired sum. Suppose

that  $g^d = r$ . Then:

$$\begin{aligned} g^d &= r \\ &= \prod_{i=1}^n \frac{m_i}{h_i} \\ &= \prod_{i=1}^n \frac{g^{d_i} \cdot X^{y_i}}{Y^{x_i}} \\ &= \prod_{i=1}^n g^{d_i} \cdot \prod_{i=1}^n \frac{X^{y_i}}{Y^{x_i}} \\ &= g^{\sum_{i=1}^n d_i} \cdot \prod_{i=1}^n \frac{(\prod_{j=1}^n X_j)^{y_i}}{(\prod_{j=1}^n Y_j)^{x_i}} \\ &= g^{\sum_{i=1}^n d_i} \cdot \prod_{i=1}^n \frac{(g^{\sum_{j=1}^n x_j})^{y_i}}{(g^{\sum_{j=1}^n y_j})^{x_i}} \\ &= g^{\sum_{i=1}^n d_i} \cdot \frac{g^{\sum_{i=1}^n \sum_{j=1}^n x_j y_i}}{g^{\sum_{i=1}^n \sum_{j=1}^n y_j x_i}} \\ &= g^{\sum_{i=1}^n d_i} \end{aligned}$$

Thus,  $g^d = g^{\sum_{i=1}^n d_i}$ , and therefore  $d = \sum_{i=1}^n d_i$ , as desired. ■

**2.4 Privacy Analysis** Next, we establish our privacy guarantee based on a standard cryptosystem—the ElGamal encryption scheme. In this encryption scheme, to encrypt a message  $\alpha$  using public key  $X$ , we compute

$$C = (\alpha X^k, g^k),$$

where  $k$  is chosen uniformly at random in  $[0, q - 1]$ . It has been shown in [TY98] (under standard complexity-theoretic assumptions) the ElGamal encryption scheme is secure in the sense of *semantic security* (see, e.g., [GM84] for the definition of semantic security). Informally, the notion of *semantic security* means that whatever the attacker could compute from the ciphertext the attacker could compute without ciphertext, then attacker learns nothing new by seeing ciphertext.

Our protocol makes use of a *homomorphic* property of a modified version of ElGamal. Specifically, if the message  $\alpha$  is changed to  $g^\alpha$  before encrypting, then from encryptions  $(g^{\alpha_1} X^{k_1}, g^{k_1})$  and  $(g^{\alpha_2} X^{k_2}, g^{k_2})$  of  $\alpha_1$  and  $\alpha_2$ , respectively, then it is possible to derive an encryption of  $\alpha_1 + \alpha_2$ , as  $(g^{\alpha_1} X^{k_1} \cdot g^{\alpha_2} X^{k_2}, g^{k_1} \cdot g^{k_2}) = (g^{(\alpha_1 + \alpha_2)} X^{(k_1 + k_2)}, g^{(k_1 + k_2)})$ .

In our protocol, the message  $m_i$  sent by customer  $U_i$  is equivalent to the first part of an ElGamal encryption of  $d_i$  under a private key  $(\sum x_i) y_i$ , while the message  $h_i$  is the second part is part of an ElGamal encryption of  $d_i$  under private key  $(\sum y_i) x_i$ . Together, all the customer

messages are combined by the miner to be an encryption of  $g^d$ . In order to undo the resulting encryption, the miner must first, in its first step, decrypt to learn  $r$ , which is  $g^d$ , and then since even the miner cannot take discrete logarithms, the miner must use trial and error to learn  $d$ . Since the range of possible values of  $d$  is not too large, this use of trial and error is feasible.

In the following, we show that our protocol protects each customer's privacy (even if there are up to  $n - 2$  users colluding with the miner) as long as the ElGamal encryption scheme is secure. Throughout the rest of this paper, we take the semantic security of ElGamal encryption as an assumption.

**THEOREM 2.2.** *Assuming that all keys have been distributed properly when the protocol starts, the protocol for mining frequency presented in Figure 1 protects each honest customer's privacy against the miner and up to  $n - 2$  corrupted customers.*

**Proof:** For *honest* customers' privacy, clearly it suffices to consider the case with the maximum number ( $n - 2$ ) of corrupted customers. Since our protocol is symmetric in customer indices, without loss of generality we assume that  $I = \{3, 4, \dots, n\}$ . Recall that, to prove the protocol protects customer privacy, we need to construct a simulator  $M$  that can generate an ensemble indistinguishable from the miner and the corrupted customers' view using only the final result  $d$ , the corrupted users' knowledge, and the public keys. Given this simulator algorithm, we can state that the miner and the corrupted customers jointly learn nothing beyond  $d$ .

Instead of describing the entire simulator in detail, we give an algorithm that computes the view of the miner and the corrupted customers in polynomial time using only  $d$ , corrupted customers' knowledge, public keys, and some ElGamal encryptions. Under the assumption that the ElGamal encryption is semantically secure, we already know that each ElGamal ciphertext can be simulated. Therefore, combining our algorithm with a simulator for ElGamal ciphertexts, we obtain a complete simulator.

Below is the algorithm that computes the view of the miner and the corrupted customers. It takes four encryptions as its input:  $(u_{11}, v_{11}) = (g^{d_1} \cdot g^{x_1 y_1}, g^{x_1})$ ,  $(u_{12}, v_{12}) = (g^{d_1} \cdot g^{x_2 y_1}, g^{x_2})$ ,  $(u_{21}, v_{21}) = (g^{d_2} \cdot g^{x_1 y_2}, g^{x_1})$ ,  $(u_{22}, v_{22}) = (g^{d_2} \cdot g^{x_2 y_2}, g^{x_2})$ . Then it computes  $m_1, m_2$  by the computation:

$$(4) \quad m'_1 = u_{11} u_{12} X_1^{\sum_{i \in I} x_i};$$

$$(5) \quad m'_2 = u_{21} u_{22} Y_2^{\sum_{i \in I} x_i}.$$

It computes  $h_1, h_2$  by the computation:

$$(6) \quad h'_1 = u_{11} u_{21} X_1^{\sum_{i \in I} y_i} / g^{d - \sum_{i \in I} d_i};$$

$$(7) \quad h'_2 = u_{12} u_{22} X_2^{\sum_{i \in I} y_i} / g^{d - \sum_{i \in I} d_i},$$

completing the proof.  $\blacksquare$

We note that the security of ElGamal encryption depends on new random values being used for each encryption. In our setting, this means that the  $x_i$  and  $y_i$  values, and associated  $X$  and  $Y$ , cannot be reused in different uses of the protocol. However, since these parameters do not depend on the actual data values, they can in general be precomputed off-line before the protocol starts. In particular, if the protocol is to be run many times, many sets of values could be precomputed in advance so that only a single phase of key distribution is required.

### 2.5 Experimental Results of Frequency Mining

We implemented our privacy-preserving frequency mining algorithm in C, using the OpenSSL libraries for the cryptographic operations. We ran a series of experiments on a PC with a 1GHz processor and 512MB memory under NetBSD. In our experiments, the length of each cryptographic key is 512 bits. We measured the computing time of the privacy-preserving frequency mining protocol for different numbers of customers, from 2,000 to 10,000.

To set up for the privacy-preserving frequency mining protocol, the key-generation time for each customer is 4.2 seconds. Computing the protocol parameters  $X$  and  $Y$  for 10,000 customers takes 140 milliseconds. As previously noted, these values can be precomputed off-line before the protocol starts.

In the privacy-preserving frequency mining protocol, it takes each customer to prepare her message to the miner only 1 millisecond, as it requires just a single modular exponentiation. The miner's computation is somewhat longer, but still quite efficient. Figure 2 shows the times the miner uses to compute one frequency for different numbers of customers. For example, for 10,000 customers, the miner's computation takes 146 milliseconds. As these results demonstrate, the privacy-preserving frequency mining protocol is very efficient.

### 3 Protecting Customer Privacy in Learning Naive Bayes Classifier

The primitive of frequency mining is simple, but is very useful in data mining applications. Correspondingly, our privacy-preserving frequency mining solution is also

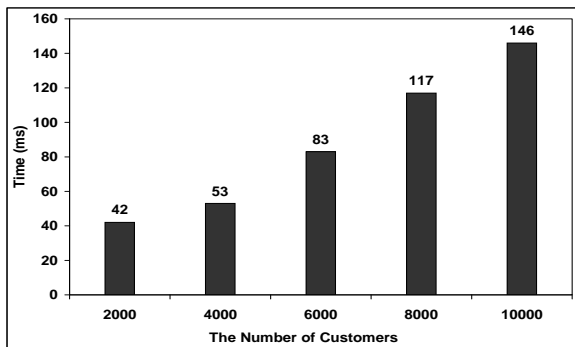


Figure 2: Server's Computation Time for a Single Frequency Calculation

quite simple, but is potentially useful whenever privacy is a top concern. In this section, we demonstrate the power of our primitive by showing a privacy-preserving naive Bayes classifier computation in the fully distributed setting (which can be thought of a horizontally partitioned database in which *each* record is held by a different party).

**3.1 Naive Bayes Learning with Privacy Concerns** Naive Bayes classifiers have been used in many practical applications. They greatly simplify the learning task by assuming that attributes are independent given the class. Although independence of attributes is an unrealistic assumption, naive Bayes classifiers often compete well with more sophisticated models, even if there is modest correlation between attributes. Naive Bayes classifiers have significant advantages in terms of simplicity, learning speed, classification speed, and storage space. They have been used, for example, in text classification and medical diagnosis [DP97, Mit97].

In (non-private) naive Bayes learning, the miner is given a set of training examples. We assume that each example is an attribute vector of a customer together with her class label. From these examples the miner learns a classifier that can be used to classify new instances.

In this paper, we consider the following scenario: there are  $m$  attributes,  $(A_1, A_2, \dots, A_m)$ , and one class attribute  $V$ . Without loss of generality, we assume that each attribute  $A_i$  ( $1 < i < m$ ) has a domain of  $\{a_i^{(1)}, \dots, a_i^{(d)}\}$  and the class attribute  $V$  has a domain of  $\{v^{(1)}, \dots, v^{(p)}\}$ . We also assume that there are  $n$  customers  $(U_1, \dots, U_n)$ , where each customer  $U_j$  has a vector denoted by  $(a_{j1}, \dots, a_{jm}, v_j)$ . In the customer's vector,  $(a_{j1}, \dots, a_{jm}, v_j)$  is an instance of the attributes vector  $(A_1, \dots, A_m)$  and  $v_j$  is  $U_j$ 's class label. In our problem, these data are the training samples from which

the miner learns the classifier without learning the samples themselves. The miner surveys all customers for values based on their data, and constructs a classifier to classify a new instance by selecting the most likely class label  $v$ :

$$(8) \quad v = \operatorname{argmax}_{v^{(\ell)} \in V} \Pr(v^{(\ell)}) \prod_{i=1}^m \Pr(a_i | v^{(\ell)}),$$

where  $(a_1, \dots, a_m)$  is the attributes vector of the new instance.

To learn the naive Bayes classifier, traditionally the miner collects all customers' data into one central site, and then learns the classifier at that central site. In our setting, there is a set  $S$  of privacy-sensitive attributes where  $S \subseteq A$ . Formally, for any  $j \in \{1, \dots, n\}$ ,  $U_j$  is not willing to reveal any information about  $(a_{ji})_{i \in S}$  to the miner; but she is willing to reveal to the miner all the remaining non-sensitive attribute values  $(a_{ji})_{i \in \{1, \dots, m\} - S}$ . To protect customers' privacy and also enable learning classifier, we design a privacy-preserving protocol for naive Bayes learning.

### 3.2 A Privacy-preserving Protocol for Naive Bayes Learning

We use the primitive for frequency mining in Section 2 as a building block to design a privacy-preserving protocol for naive Bayes learning. In this protocol the miner knows the schema of customer data. Without loss of generality, we assume all customers' sensitive attributes need to be protected. We have two goals to achieve:

- *Correctness*: the miner learns the naive Bayes classifier accurately.
- *Privacy*: the miner learns nothing about each customer's sensitive data except the knowledge derivable from the naive Bayes classifier itself.

To achieve these two goals, we first rewrite (8) as:

$$(9) \quad v = \operatorname{argmax}_{v^{(\ell)} \in V} \#(v^{(\ell)}) \prod_{i=1}^m \frac{\#(a_i, v^{(\ell)})}{\#(v^{(\ell)})},$$

where  $\#(v^{(\ell)})$  ( $\#(a_i, v^{(\ell)})$ , resp.) denotes the frequency, or number of occurrences, of attribute value  $v^{(\ell)}$  (attribute pair value  $(a_i, v^{(\ell)})$ , resp.) in all customers' data. To learn the classifier, all the miner needs to do is to learn  $\#(v^{(\ell)})$  and  $\#(a_i^{(k)}, v^{(\ell)})$  for each  $i \in S$ , each  $k$ , and each  $\ell$ . Since the occurrence of  $v^{(\ell)}$  or of the pair  $(a_i^{(k)}, v^{(\ell)})$  can be denoted by a Boolean value, we can use the primitive presented in Section 2. A detailed specification of the protocol is given in Figure 3.

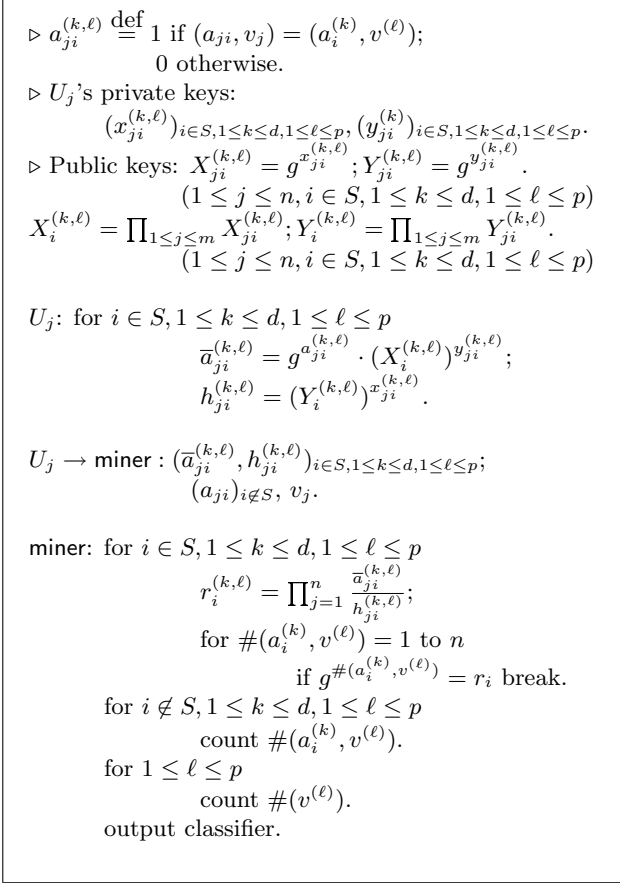


Figure 3: Privacy-Preserving Protocol for Naive Bayes Learning.

**3.3 Protocol Analysis** In the following theorem, we implicitly assume that the output classifier is encoded in such a way that it contains the frequencies  $\#(v^{(\ell)})$  and  $\#(a_i^{(k)}, v^{(\ell)})$  for all  $(i, k, \ell)$ .

**THEOREM 3.1.** *The protocol for naive Bayes learning presented in Figure 3 protects each customer's sensitive data against the miner and up to  $n - 2$  corrupted customers.*

**Proof:** Since all the frequency computations are done independently, the theorem follows immediately from Theorem 2.2. ■

For accuracy, we compare our privacy-preserving protocol with a traditional naive Bayes learning algorithm running on all customers' data without protection of privacy. Suppose that the learning algorithm without privacy protection outputs a classifier  $c$  and that our privacy-preserving protocol outputs  $c'$ . We claim  $c = c'$ , which means our protocol does not lose any accuracy as a cost of privacy.

**THEOREM 3.2.** *The protocol for naive Bayes learning presented in Figure 3 does not lose accuracy.*

**Proof:** This is straightforward from our protocol specification. Our protocol counts each  $\#(v^{(\ell)})$  and each  $\#(a_i^{(k)}, v^{(\ell)})$  for  $i \notin S$  directly. It uses the privacy-preserving method we presented in Section 2 to count each  $\#(a_i^{(k)}, v^{(\ell)})$  for  $i \in S$ . Since the method in Section 2 computes frequencies precisely, our protocol outputs exactly the same classifier as a non-privacy-preserving naive Bayes algorithm. ■

**Overhead Analysis** Recall that there are  $n$  customers and  $m$  attributes and that each (non-class) attribute has a domain of size  $d$ , and the class label has a domain of size  $p$ . Also recall that the set of privacy-sensitive attributes is  $S$ . Assume  $s = |S|$  is the number of sensitive attributes. It is easy to see that each customer has a computational overhead—as compared to a non-private solution—of  $dps$  ElGamal encryptions. In data mining applications, we usually have  $n \gg dps$ ; thus the computational overhead for each customer is small. The computation overhead for the miner is  $O(dpsn)$  modular exponentiations, which is also reasonable. The communication overhead is  $dpsn$  ciphertexts.

### 3.4 Experimental Results of Bayes Classifier Learning

The basic experimental set-up is the same here as described in Section 2.5. The Bayes classifier learning algorithm is implemented in C, and uses the frequency mining implementation as a subroutine. Again, we ran a series of experiments on a PC with a 1GHz processor and 512MB memory under NetBSD, using 512 bit cryptographic keys. For the Bayes classifier

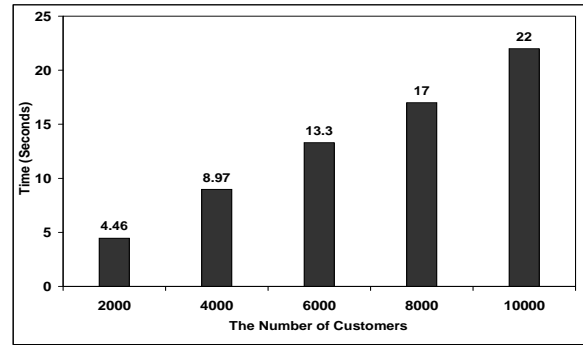


Figure 4: Server's Learning Time for Naive Bayes Classifier vs. Number of Customers

experiments, we assumed that each customer has ten attributes, that each attribute has eight nominal values, and that there are two classes. We measured the

computation time of each customer and the miner in our privacy-preserving protocol for Bayes classifier learning. Our results show that each customer needs only 120 milliseconds to compute her message flow to the miner. Figure 4 shows the computation times the miner needs to learn a naive Bayes classifier for different numbers of customers. For example, when the number of customers is 10,000, the miner’s computation requires only 22 seconds.

Figure 5 further studies how the server’s learning time changes when both the customer number and the attribute number vary. In this experiment, we fix the domain size of each non-class attribute to four and the domain size of the class attribute to two.

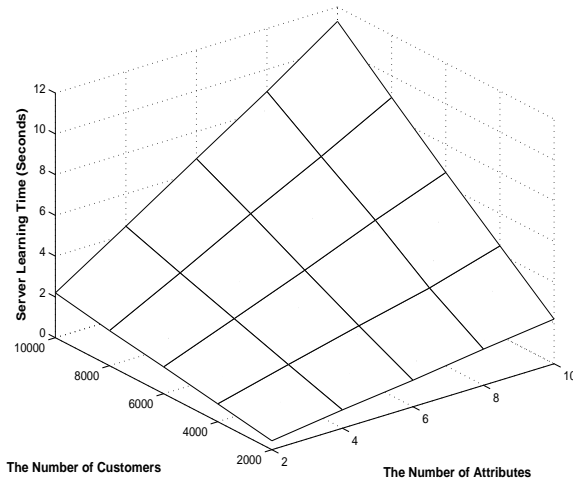


Figure 5: Server’s Learning Time for Naive Bayes Classifier vs. Number of Customers and Number of Attributes

#### 4 Extension to Other Data Mining Algorithms

We describe how our privacy-preserving frequency computation can be used for other important data mining algorithms in the fully distributed model. In Section 4.1, we describe a privacy-preserving ID3 decision tree learning algorithm. In Section 4.2, we sketch a privacy-preserving association rule mining algorithm. Both algorithms are in the fully distributed model without loss of accuracy. Both of them leak no information about the customer data beyond the computed frequencies. Both can be efficient even if the number of customers (transactions) is very large. However, both require certain parameters (such as the number of attributes) are small, as they require exponential computation in those parameters, as we discuss further below.

#### 4.1 Privacy-Preserving Learning of ID3 Trees

Using our privacy-preserving frequency primitive, we can learn ID3 trees in the fully distributed setting without loss of accuracy. Solutions such as [AS00] can be used in the fully distributed setting, but they lose some accuracy as cost of privacy; solutions such as [LP02] do not lose accuracy for privacy, but they do not work in the fully distributed setting.

The miner’s algorithm has the same complexity as the original ID3 tree algorithm, except for an additional linear overhead factor whose value is determined by the ElGamal key size used. However, the computation time of each customer is exponential on the domain size of her attribute. Therefore, our algorithm is efficiently applicable only if the attribute domains are small.

We define the problem of privacy-preserving learning of ID3 decision trees as *PPLID3*:

**DEFINITION 2.** *PPLID3*: A data miner queries  $n$  customers and learns an ID3 tree based on the customers’ responses. The miner should not be able to derive any information about each customer’s data beyond the computed frequencies.

In the *PPLID3* problem, each customer has a  $(m + 1)$ -tuple of data in which there are  $m$  non-class attributes and one class attribute. The schema of customer data and the domain of each attribute are assumed to be publicly known. Our solution is in the reduced interaction model: each customer sends only a single message flow to the data miner, and there is no further communication. Hence, if customer  $i$  sends  $E_i$  to the miner, the *security goal* is that the miner learns nothing (beyond the computed frequencies) about customer  $i$ ’s data from  $E_i$ .

First, we give a brief review of ID3 decision trees. (See [Mit97] for additional details.) An ID3 tree is a rooted tree containing nodes and edges. Each internal node is a test node and corresponds to an attribute. The edges going out of a node correspond to the possible values of that attribute. The ID3 algorithm works as follows. The tree is constructed top-down in a recursive fashion. At the root, each attribute is tested to determine how well it alone classifies the samples. The “best” attribute is then chosen and the samples are partitioned according to this attribute. The ID3 algorithm is then recursively called for each child of this node, using the corresponding subset of data.

Next, we review how the ID3 algorithm chooses the best attribute for a node. We use the following notation. Let  $A = \{A_i \mid 1 \leq i \leq m\}$  be the set of (non-class) attributes,  $V$  the class attribute, and  $T$  the set of samples (or records). For simplicity, we assume that all attributes have the same domain size  $d$ :

$A_i = \{a_i^{(j)} \mid 1 \leq j \leq d\}$ . The set of possible values of the class attribute is  $V = \{v^{(i)} \mid 1 \leq i \leq p\}$ . Let  $T(v^{(i)})$  be the set of samples with class  $v^{(i)}$ . Then the entropy is:

$$H_V(T) = \sum_{i=1}^p -\frac{|T(v^{(i)})|}{|T|} \log \frac{|T(v^{(i)})|}{|T|}.$$

Consider an attribute  $A_t = \{a_t^{(j)} \mid 1 \leq j \leq d\}$ . The conditional information of  $T$  given  $A_t$  is:

$$H_V(T \mid A_t) = \sum_{i=1}^d \frac{|T(a_t^{(i)})|}{|T|} H_V(T(a_t^{(i)})).$$

For each attribute  $A_t$ , the information gain is defined by:

$$\text{gain}(A_t) = H_V(T) - H_V(T \mid A_t)$$

The chosen attribute  $A_t$  is the attribute that can achieve the maximum information gain at each node. Clearly, the problem of choosing the best attribute can be reduced to computing entropies. Accordingly, the *PPLID3* problem can be reduced to a problem in which the miner computes entropies while no sensitive data of customers are revealed to her. Again, we use our

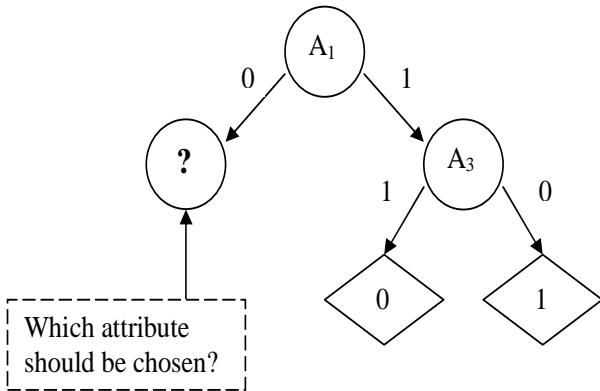


Figure 6: ID3 example

technique of privacy-preserving frequency mining to solve this problem. The solution is showed in Figure 6. Suppose that there are  $n$  customers, each holding a record with three Boolean attributes  $A_1, A_2$  and  $A_3$ . The class value is also Boolean. Figure 6 shows an intermediate state of the ID3 algorithm in which the algorithm needs to choose an attribute for the node “?”. To compute the information gain  $H_V(A_i)$  (for  $i = 2$  and  $3$ , resp.), we need to compute  $H_V(T)$  and  $H_V(T \mid A_i)$ :

$$H_V(T) = -\frac{|T(V=0)|}{|T|} \log \frac{|T(V=0)|}{|T|} - \frac{|T(V=1)|}{|T|} \log \frac{|T(V=1)|}{|T|},$$

$$H_V(T \mid A_i) = \frac{|T(A_i=0)|}{|T|} H_V(T(A_i=0)) + \frac{|T(A_i=1)|}{|T|} H_V(T(A_i=1))$$

The above formulae involve several frequencies:  $|T|$ ,  $|T(V=0)|$ ,  $|T(V=1)|$ ,  $|T(A_i=0)|$ ,  $|T(A_i=1)|$ , etc. All these frequencies can be computed using our privacy-preserving frequency mining protocol.

The general protocol is sketched in Figure 7, where  $A$  is the attribute set,  $V$  the class and  $T$  the set of all customers’ data.

**ID3** ( $A, V, T$ )

1. If  $A$  is empty, the miner returns a leaf node with the dominating class value in  $T$ .
2. Use the privacy-preserving method to count the number of records with each class label. If  $T$  consists of records which have the same class label  $v$ , return a leaf node with  $v$ .
3. Otherwise:
  - (a) Determine the best attribute  $A_i$  for  $T$  using the privacy-preserving method.
  - (b) For  $A_i = \{a_1, \dots, a_d\}$ , let  $T(a_1), \dots, T(a_d)$  be a partition of  $T$  s.t. every record in  $T(a_j)$  has attribute value  $a_j$ .
  - (c) Return a tree whose root is labeled  $A_i$ ; the root has outgoing edges labeled  $a_1, \dots, a_d$  s.t. each edge  $a_j$  goes to the tree  $\text{ID3}(A - A_i, V, T(a_j))$ .

Figure 7: Privacy-preserving Protocol for Learning ID3 Tree

Unlike naive Bayes classification, which assumes independence between non-class attributes given the class label, attributes are interdependent with ID3. If the class attribute has  $p$  class labels, and each of the  $m$  non-class attributes has a domain of size  $d$ , then the number of joint frequencies that need to be counted is exponential in  $m$ . In some cases we have small  $m$  and  $d$  and thus we can still achieve reasonable overhead. For example, the data set of Car Evaluation Database from UCI repository [BM98] has six nominal attributes: buying, maint, doors, persons, lug\_boot and safety, and the class attribute has a domain of size four. For such

a scenario, we estimate that each customer needs only one minute to compute her message flow to the miner. Another example [Mit97, Ch. 3] is a weather data set containing four data attributes: **outlook**, **temperature**, **humidity**, and **windy**, and a class attribute, **play**. If each customer holds one weather record, we estimate that each customer needs only about 0.5 seconds to compute her message flow to the miner.

**4.2 Association Rule Mining** The mining of association rules can be formulated as follows. Let  $I = \{i_1, i_2, \dots, i_m\}$  be a set of items. Let  $D$  denote a set of  $n$  transactions (or records), where each transaction  $T$  is a subset of  $D$ . Associated with each transaction is a unique identifier. We say that a transaction  $T$  contains  $P$  if  $P \subset T$ . An association rule is an implication of the form  $P \Rightarrow Q$ , where  $P \subset I$ ,  $Q \subset I$ , and  $P \cap Q = \emptyset$ . The rule  $P \Rightarrow Q$  has *support*  $s$  in  $D$  if  $s\%$  of the transactions in  $D$  contain  $P \cap Q$ .  $P \Rightarrow Q$  holds with  $c$  *confidence* if  $c\%$  transactions in  $D$  that contain  $P$  also contain  $Q$ .

Association rule mining can be reduced to computing the frequency of a number of particular sets. Using the privacy-preserving frequency mining protocol in Section 2, we enable the miner to compute the frequency of any set of items from customer response. From these, the miner can learn all association rules. Note that this leaks *all* the computed frequencies to the miner, rather than only revealing the actual frequent itemsets.

In the fully distributed setting, each customer has a transaction. The miner wants to learn the association rules based the customer data, but each customer does not want to reveal her data to the miner. Consider a small example in which  $P = \{p_1, p_2, p_3\}$  and  $Q = \{q_1, q_2\}$ . We describe how to compute confidence and support for the rule  $P \Rightarrow Q$ . Each customer sends the encryptions of the occurrences of  $\{p_1, p_2, p_3, q_1, q_2\}$ ,  $\{p_1, p_2, p_3\}$  and  $\{q_1, q_2\}$ . By applying the privacy-preserving frequency-mining protocol, the miner can easily learn the confidence and support from customers' output. To learn all association rules with threshold of  $c$  and  $s$  on  $D$ , each customer must compute all possible combinations of the occurrence of sets of items, which is of course exponential on the size  $m$  of the item domain. Hence, the solution is only practical if  $m \ll n$ .

## 5 Conclusion

In this paper, we proposed a privacy-preserving method of frequency mining and applied it to naive Bayes learning in a fully distributed setting. If this problem is solved using randomization techniques, then there is a trade-off between privacy and accuracy. However, our proposed solution enjoys cryptographically strong privacy without losing any accuracy as cost of privacy.

Furthermore, both theoretical analysis and experimental results show that the method itself and its application to naive Bayes learning are very efficient. Our work assumes that the data are horizontally partitioned such that each customer holds a row. Therefore, an immediate open question is whether a solution similar to ours can be found for the case in which the data are fully vertically partitioned among different parties.

We also discussed other possible applications of the proposed privacy-preserving frequency mining: privacy-preserving learning of ID3 trees and association rule mining. These are very practical in some cases, but rely on certain parameters being small. A second open question is whether those applications can be made as efficient as our privacy-preserving protocol for naive Bayes learning. We conjecture that additional techniques are needed to make such protocols efficient.

A third open question is whether it is possible to combine our method with randomization techniques to further improve efficiency. For example, is there a protocol for frequency mining or naive Bayes learning that is more efficient than the one presented in this paper, but still enjoys full privacy and does not lose any accuracy? Alternately, it may be possible to combine some randomization techniques with some cryptographic techniques to further improve the efficiency while the resulting loss of privacy can be quantified and thus limited to an acceptable extent. We believe these questions are worth further investigation.

## Acknowledgement

We thank the anonymous reviewers for their insightful comments.

## References

- [AA01] D. Agrawal and C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proc. of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 247–255. ACM Press, 2001.
- [AJL04] A. Ambainis, M. Jakobsson, and H. Lipmaa. Cryptographic randomized response techniques. In *Proc. of the 2004 International Workshop on Practice and Theory in Public Key Cryptography (PKC)*, pages 425–438. Springer, 2004.
- [AS00] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the ACM SIGMOD Conference on Management of Data*, pages 439–450. ACM Press, May 2000.
- [AW89] N. Adam and J. Worthmann. Security-control methods for statistical databases: a comparative study. *ACM Comput. Surv.*, 21(4):515–556, 1989.

- [BGW88] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. of the 20th Annual ACM Symposium on Theory of Computing*, pages 1–10. ACM Press, 1988.
- [BM98] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.
- [BT94] J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proc. of the 26th Annual ACM symposium on Theory of Computing*, pages 544–553. ACM Press, 1994.
- [Cra99] L. Cranor, editor. *Comm. ACM* 42(2), *Special Issue on Internet Privacy*, 1999.
- [DN03] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proc. of the 22nd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210. ACM Press, 2003.
- [DN04] C. Dwork and K. Nissim. Privacy-preserving data mining on vertically partitioned databases. In *Advances in Cryptology - Proceedings of CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, Santa Barbara, California, August 2004.
- [DP97] P. Domingos and M. Pazzani. On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.
- [DZ03] W. Du and Z. Zhan. Using randomized response techniques for privacy-preserving data mining. In *Proc. of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 505–510. ACM Press, 2003.
- [EGS03] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proc. of the 22nd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 211–222. ACM Press, 2003.
- [ESAG02] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 217–228. ACM Press, 2002.
- [GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Computer and System Sciences*, 28:270–299, 1984.
- [GMW87] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proc. of the 19th Annual ACM Conference on Theory of Computing*, pages 218–229. ACM Press, 1987.
- [Gol04] O. Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.
- [HS00] Martin Hirt and Kazue Sako. Efficient receipt-free voting based on homomorphic encryption. *Lecture Notes in Computer Science*, 1807:539+, 2000.
- [KC02] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *The ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'02)*, pages 24–31, June 2002.
- [KDWS03] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *The Third IEEE International Conference on Data Mining*, 2003.
- [KV02] M. Kantarcioglu and J. Vaidya. An architecture for privacy-preserving mining of client information. In *IEEE ICDM Workshop on Privacy, Security and Data Mining*, pages 37–42, 2002.
- [KV03] M. Kantarcioglu and J. Vaidya. Privacy preserving naive Bayes classifier for horizontally partitioned data. In *IEEE Workshop on Privacy Preserving Data Mining*, 2003.
- [LP02] Y. Lindell and B. Pinkas. Privacy preserving data mining. *J. Cryptology*, 15(3):177–206, 2002.
- [Mit97] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
- [RH02] S. Rizvi and J. Haritsa. Maintaining data privacy in association rule mining. In *Proc. of the 28th VLDB Conference*, 2002.
- [Sch96] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, second edition, 1996.
- [SWY04] H. Subramaniam, R. N. Wright, and Z. Yang. Experimental analysis of privacy-preserving statistics computation. In *Proc. of the VLDB Workshop on Secure Data Management*, pages 55–66, August 2004.
- [TY98] Y. Tsiounis and M. Yung. On the security of ElGamal-based encryption. In *Public Key Cryptography'98*, volume 1431 of *Lecture Notes in Computer Science*, pages 117–134, 1998.
- [VC02] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proc. of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 639–644. ACM Press, 2002.
- [VC03] J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proc. of the Ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215. ACM Press, 2003.
- [VC04] J. Vaidya and C. Clifton. Privacy preserving naive Bayes classifier on vertically partitioned data. In *2004 SIAM International Conference on Data Mining*, 2004.
- [War65] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.
- [Wes99] A. Westin. Freebies and privacy: What net users think. Technical report, Opinion Research Corporation, 1999.
- [WY04] R. N. Wright and Z. Yang. Privacy-preserving Bayesian network structure computation on distributed heterogeneous data. In *Proc. of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 713–718. ACM Press, 2004.
- [Yao86] A. Yao. How to generate and exchange secrets. In *Proc. of the 27th IEEE Symposium on Foundations of Computer Science*, pages 162–167, 1986.