# A New Scheme on Privacy-Preserving Data Classification *

Nan Zhang, Shengquan Wang, and Wei Zhao
Department of Computer Science
Texas A&M University
College Station, TX 77843, USA
{nzhang, swang, zhao}@cs. tamu.edu

## ABSTRACT

We address privacy-preserving classification problem in a distributed system. Randomization has been the approach proposed to preserve privacy in such scenario. However, this approach is now proven to be insecure as it has been discovered that some privacy intrusion techniques can be used to reconstruct private information from the randomized data tuples. We introduce an algebraic-technique-based scheme. Compared to the randomization approach, our new scheme can build classifiers more accurately but disclose less private information. Furthermore, our new scheme can be readily integrated as a middleware with existing systems.

## Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.2.7 [**Database Management**]: Database Administration—*Security, integrity, and protection*

## General Terms

Security

## Keywords

Privacy, Privacy-preserving data mining

## 1. INTRODUCTION

In this paper, we address issues related to privacy-preserving data mining. In particular, we focus on privacy-preserving data classification. General classification techniques have been extensively studied for over twenty years [15]. The main purpose of data classification is to build a model (i.e., classifier) to predict the (categorical) class labels of data tuples [10] based on a training data set where the class label of each data tuple is given. The classifier is usually represented by classification rules, decision trees, neural networks, or mathematical formulae that can be used for classification.

In recent years, the issue of privacy protection in classification has been raised [2, 14]. The objective of privacy-preserving data classification is to build accurate classifiers without disclosing private information in the data being mined. The performance of privacy-preserving techniques should be analyzed and compared in terms of both the privacy protection of individual data and the predictive accuracy of the constructed classifiers.

We consider a distributed environment in which training data tuples are stored in multiple autonomous entities. We can classify distributed privacy-preserving classification systems into two categories based on their infrastructures: *Server-to-Server* (S2S) and *Client-to-Server* (C2S), respectively.

In the first category (S2S), data tuples in the training data set are distributed across several servers. Each server holds a private database, which contains part of the training data set. The servers collaborate with each other to construct a classifier over the integration of all databases without letting either server know the private information of the other parties. This problem is usually formulated as a variation of secure multiparty computation problem [14]. Existing algorithms in this category can build decision trees [7, 14] and naïve Bayesian classifiers [12, 16] when the training data tuples are vertically [16] or horizontally [7, 12, 14] partitioned into multiple databases.

In the second category (C2S), a system usually consists of a data miner (*server*) and numerous data providers (*clients*). Each data provider holds only one training data tuple. As is commonly assumed [2], the class label attribute of each data tuple is not considered as sensitive information by the data providers. All other attributes contains private information which needs to be preserved. The data miner builds a classifier on the aggregate data provided by the data providers. Due to privacy concern, the data miner may compromise private information in the data being mined. To prevent privacy from being compromised by the data miner, countermeasures must be implemented with the data providers. An online survey system is a typical example for C2S systems, as the system consists of one survey collector/analyzer (data miner) and thousands of survey respondents (data providers).

Both S2S and C2S systems have a broad range of applications. Nevertheless, we focus on studying privacy-preserving data classification in C2S systems. In a C2S system, the common objective of the data providers and the data miner is to build a predic-

*This work was supported in part by the National Science Foundation under Contracts 0081761, 0324988, 0329181, by the Defense Advanced Research Projects Agency under Contract F30602-99-1-0531, and by Texas A&M University under its Telecommunication and Information Task Force Program. Any opinions, findings, conclusions, and/or recommendations expressed in this material, either expressed or implied, are those of the authors and do not necessarily reflect the views of the sponsors listed above.


*KDD'05*, August 21–24, 2005, Chicago, Illinois, USA.

tively accurate classifier. Besides, the data providers have an objective to preserve their private information. As such, the goal of privacy-preserving data classification in C2S systems is to limit the information obtained by the data miner to be *minimum necessary* to accomplish the intended purpose of building predictively accurate classifier. This goal is also referred to as "minimum necessary standard" in real-world privacy rules (e.g., Health Insurance Portability and Accountability Act (HIPAA) privacy rule [11]).

Previous studies observed that precise values of individual training data are not necessary in data classification. As is shown in [2], accurate classifiers can be built upon a robust estimate of the distribution of training data tuples. Randomization approach has been proposed for the data providers to add random noise to private data tuples before transmitting them to the data miner. As such, the data providers protect their privacy by using the random noise. The data miner can still reconstruct the original distribution from the randomized data and thereby build an accurate classifier.

Most of the current studies on C2S systems tacitly assumed that randomization was the only effective approach to preserving privacy while keeping the mining results meaningful. In the randomization approach, each attribute of a training data tuple has to be equally processed (i.e., randomized by the data providers and transmitted to the data miner) because a data provider cannot obtain any information from either the data miner or other data providers indicating which attributes are more important for building an accurate classifier. As we will illustrate in Section 2, there are several problems with this kind of approach:

- Some attributes may be unnecessarily transmitted to the data miner, as they are not necessary in building the classifier. This increases the risk of privacy leakage.
- The distribution of some necessary attributes may not be reconstructed accurately after randomization.
- Worst of all, it is shown in [13] that by using the spectral information of the randomized data, a data miner may reconstruct individual data even if they have been randomized.

In this paper, we develop a new scheme based on algebraic techniques. In our scheme, the data providers do not just perturb their data by using random noise. Instead, a perturbation guidance is transferred from the data miner to the data providers as a reference to the data perturbation. Roughly speaking, the perturbation guidance indicates which attributes of the data tuple are the *minimum necessary* ones to build an accurate classifier. After checking the validity of the perturbation guidance, the data providers perturb their data accordingly. As such, our scheme adheres to the minimum necessary standard by transmitting only the minimum necessary information to the data miner.

We will demonstrate that our new scheme has the following important features to distinguish itself from previous approaches.

- Our scheme can help to build classifiers that have better accuracy but disclose less private information. An upper bound on the error introduced to the predictive accuracy of the classifier built is derived and can be used to predict the system accuracy in reality.
- Our scheme allows each data provider to play a role in determining the tradeoff between accuracy and privacy. Specifically, we allow each data provider to choose a different level of privacy protection. This makes our system meet the needs of a wide range of data providers, from hard-core privacy protectionists to privacy marginally concerned individuals.
- Our scheme is flexible and easy to implement. It does not require a distribution reconstruction component as have previous approaches. Our scheme is transparent to the data classification approach and can be readily integrated with existing systems as a middleware.

The algebraic-techniques-based approach was first proposed in our work for association rule mining [17]. Significant differences between our work in this paper and [17] include

- The data mining application is different: We are dealing with data classification instead of association rule mining in [17].
- The adversary model is different: We are preserving privacy against malicious data miners instead of semi-honest data miners (i.e., the data miners which follow the protocol strictly, with the only exception that they may record the intermediate results and communication).

The rest of the paper is organized as follows: We briefly review the randomization approach in Section 2. In Section 3 and Section 4, we introduce our new scheme and its basic components, respectively. We present a theoretical analysis on the performance of our scheme in Section 5. Theoretical bounds on the accuracy and privacy metrics are also derived in this section. An experimental performance evaluation of our scheme is provided in Section 6. In this section, we make a comparison between the performance of our scheme and the randomization approach, and show the simulation results of our scheme on real data sets. The implementation and runtime efficiency of our scheme is discussed in Section 7, followed by final remarks in Section 8.

# 2. RANDOMIZATION APPROACH AND ITS PROBLEMS

In this section, we review the randomization approach, which has been proposed and used to preserve privacy in data classification. We also analyze the problems associated with this approach, motivating us to propose a new scheme on privacy-preserving data classification.

## 2.1 Overview

Based on the randomization approach, the entire privacy-preserving classification process can be considered a two-step process. The first step is for data providers to randomize their data, and transmit the (randomized) data to the data miner. As in an online survey system where different survey respondents come at different time, we consider this step to be iteratively carried out in a group of independent processes [1]. In each process, a data provider applies a randomization operator $R(\cdot)$ to its data tuple and transmits the randomized data tuple to the data miner. In previous studies, several randomization operators have been proposed including the random perturbation operator [2] and the random response operator [8], which are shown in (1) and (2), respectively,

$$R(t) = t + r. \tag{1}$$

$$R(t) = \begin{cases} t, & \text{if } r < \theta. \\ \bar{t}, & \text{if } r \geq \theta. \end{cases} \tag{2}$$

where $t$ is the original data tuple, $r$ is the random noise, and $\theta$ is a parameter predetermined by the data providers. As the result of this step, the data miner obtains perturbed training data tuples from the data providers.

[1]Nevertheless, without loss of generality, we assume that these processes are executed in a serializable manner.

In the second step, the data miner builds a classifier on the aggregate data. With the randomization approach, the data miner must first employ a distribution reconstruction algorithm that intends to reconstruct the original data distribution from the randomized data tuples. Several distribution reconstruction algorithms have been proposed [1,2,8]. For example, the expectation maximization (EM) algorithm [1] reconstructs a distribution which converges to the maximum likelihood estimate of the original distribution.

Also in the second step, a malicious data miner may compromise private information using a privacy data recovery algorithm on the randomized data tuples supplied by the data providers.

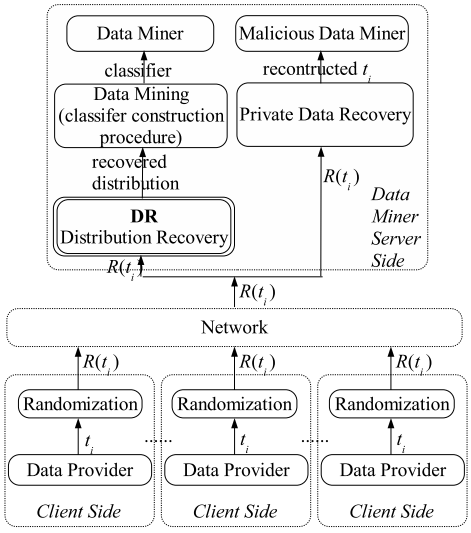


**Figure 1: Randomization Approach**

Figure 1 depicts the system architecture with the randomization approach. Note that there are two kinds of data miners: an honest data miner which performs legal data mining functions without any intent to discover private data of the data providers; and a malicious data miner which is interested in the privacy of data providers and uses private data recovery method to realize this goal. Clearly, any such data classification system should be measured by its capability of both building accurate classifiers and preventing private data leakage.

## 2.2 Problems

While the randomization approach is intuitive, researchers have recently identified privacy breaches as one of the major problems with the randomization approach. It is shown in [13] that the spectral properties of randomized data could help the data miner to separate noise from private data. In particular, a filtering method is proposed based on random matrix thoery to reconstruct private data from the randomized data set [13]. The performance of this method demonstrates that randomization preserves very little privacy in many cases.

Randomization approach also suffers from efficiency problems as it puts a heavy load on the data miner at run time (because of the distribution reconstruction). It is shown in [3] that the cost of mining randomized data set is "*well within an order of magnitude*" in respect to that of mining original data set. [2]

[2]Although the work reported in [3] is based on association rule mining, we believe that the similarity between randomization operators in association rule mining and data classification makes the efficiency concern inherent in the randomization approach.

Another problem with the randomization approach is that it cannot be adapted to the diverse needs of data providers. A survey [6] on privacy concern shows that among the Internet users (potential data providers), there are 17% privacy fundamentalists, 56% privacy pragmatists, and 27% marginally concerned individuals. Privacy fundamentalists are extremely concerned about privacy. Privacy pragmatists are concerned about privacy, but their concerns are much less than those of the fundamentalists. Marginally concerned individuals are generally willing to provide their private data. The randomization approach treats all the data providers in the same manner and does not address the differing need of data providers. As such, a privacy fundamentalist may not want to provide its data while the accurate data from a marginally concerned individual is wasted.

We believe that the following are the reasons behind the abovementioned problems.

- Randomization operator is user-invariant. The same perturbation level is applied to all data providers. The reason is that in a system using randomization approach, the communication is one-way: from the data providers to the data miner. As such, a data provider cannot obtain any user-specified guidance on the randomization of its private data.

- Randomization operator is attribute-invariant. All attributes are equally perturbed. The distribution of each attribute, no matter how useful it is in the classification, is equally preserved in the perturbation. The reason is, again, the lack of communication between the data miner and the data providers. A data provider cannot learn the correlation between different attributes. As such, a data provider has no choice but to perturb its data in an attribute-invariant manner.

The one-way communication scheme is inherent in the randomization approach. This motivates us to propose a new scheme which allows two-way communication between the data miner and the data providers. Another possible solution to the problems of randomization approach is to introduce communication between data providers. We do not take this method because in our system model, different data providers come at different time and thus may not be able to communicate with each other. Our other hesitation with this approach includes 1) it introduces a problem of the trustworthiness of other data providers, and 2) it may place high computational load on data providers, which are supposed to have lower computational power than the data miner.

# 3. OUR NEW SCHEME

In this section, we introduce our scheme. Figure 2 depicts the infrastructure of our new scheme. The communication protocol of our scheme is shown in Algorithm 1. In our scheme, there is an important parameter for each data provider, called *maximum acceptable disclosure level*, which is denoted by $k_i$. Roughly speaking, if we consider the perturbed data tuple as a random vector, then $k_i$ is the degree of freedom of the perturbed data tuple, which in most cases is much smaller than the degree of freedom of the original data tuple. With a larger $k_i$, the data miner can make a more stable estimation on the distribution of original data tuples. Nonetheless, the data miner will also have more information about the individual private data tuple. Thus, the larger $k_i$ is, the more contribution the perturbed data tuple will make to building the classifier. The smaller $k_i$ is, the more private information is preserved. As such, a privacy fundamentalist can choose a small $k_i$ to protect its privacy. A privacy unconcerned individual can choose a large $k_i$ to help building a more accurate classifier. The relationship between

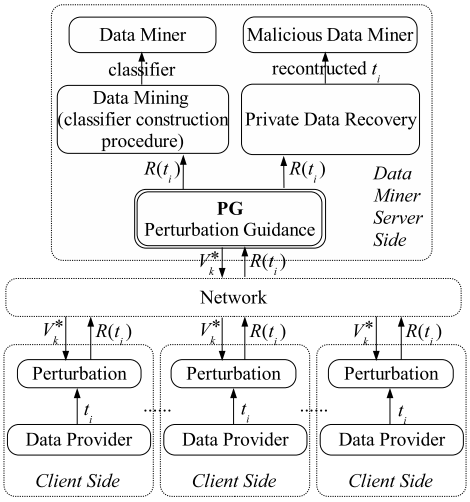


**Figure 2: Our New Scheme**

$k_i$ and the amount of privacy disclosure is analyzed in Section 5 and demonstrated in Section 6.

Before sending its data to the data miner, a data provider first inquires the data miner what the current *system disclosure level* $k^*$ is. Roughly speaking, $k^*$ is the minimum necessary disclosure level for the data miner to construct an accurate classifier. The perturbation guidance component of the data miner computes $k^*$ and transmits it back to the data provider. If $k^*$ is not acceptable by the data provider (i.e., $k^* > k_i$), the data provider can keep trying. As we will show in Section 6, $k^*$ decreases rapidly when the number of data tuples received by the data miner increases. Since the levels of privacy concerns vary among different data providers [6], the system disclosure level will be acceptable by all data providers eventually.

If $k^*$ is accepted by a data provider (i.e., $k^* \leq k_i$), the data provider inquires for the current system perturbation guidance $V_k^*$. The data miner computes the perturbation guidance $V_k^*$ based on the system disclosure level $k^*$ and dispatches $V_k^*$ to the data provider. Roughly speaking, $V_k^*$ is the vector that projects the original data tuple into a $k^*$-dimensional subspace where the data tuples from different classes are most different. As such, the private information divulged by the perturbed data tuple is the most valuable information for constructing accurate classifiers. This complies to our standard of disclosing only the minimum necessary information to the data miner.

Once $V_k^*$ is received, the data provider checks the validity of $V_k^*$, computes the perturbed data tuple $R(t_i)$ from its private data tuple $t_i$, and transmits $R(t_i)$ along with its class label to the data miner. After all data providers send their data to the data miner, the perturbed data tuples received by the data miner are directly used as the training data tuples to build the classifier.

As we can see, our scheme requires two rounds of message exchange to dispatch the perturbation guidance: the round to inquire $k^*$ and the round to inquire $V_k^*$. Another possible approach is to let a data provider transmit its maximum acceptable disclosure level $k_i$ to the data miner. If $k_i \geq k^*$, the data miner transmits $V_k^*$ back to the data provider. This approach only requires one round of message exchange. However, privacy breach may occur if this approach is used because when $k_i > k^*$, a malicious data miner can manipulate a disclosure level $\hat{k}$ such that $k_i \geq \hat{k} > k^*$ and generate a perturbation guidance based on $\hat{k}$. As such, the data miner may compromise private data which are unnecessary to build an accurate classifier.

**Algorithm 1** Our new scheme

For the data miner:

1: Upon receiving an inquiry message from a data provider, the perturbation guidance (PG) component computes the current system disclosure level $k^*$ and sends it to the data provider;
2: Upon receiving a ready message from a data provider, the PG component computes $V_k^*$ and sends it to the data provider;
3: Upon receiving a perturbed data tuple $R(t_i)$ and its class label $a_0$ from a data provider, sends $R(t_i)$ and $a_0$ to the PG component.

For a data provider:

*Input*: $k_i$, the maximum acceptable disclosure level of the data provider.

1: Sends an inquiry message to the data miner to obtain the current system disclosure level $k^*$.
2: **if** the received $k^*$ is less than or equal to $k_i$ **then**
3: Sends a ready message to the data miner.
4: **else**
5: Goto 1;
6: **end if**
7: Upon receiving $V_k^*$ from the data miner, sends $V_k^*$ to the perturbation component to check its validity. If $V_k^*$ is valid, the perturbation component perturbs the private data tuple based on $V_k^*$ and sends the perturbed data tuple along with its class label to the data miner.

Compared to the randomization approach, our scheme does not have the distribution recovery component. Instead, the classifier construction procedure is performed on the perturbed data tuples directly. Our scheme has two key components, which are the *perturbation guidance* (PG) component of the data miner and the *perturbation* component of the data providers. We will introduce these two components in details in the next section.

# 4. BASIC COMPONENTS

The basic components of our scheme are: a) the PG component of the data miner which computes the current system disclosure level $k^*$ and the perturbation guidance $V_k^*$, and b) the perturbation component of the data providers which checks the validity of $V_k^*$ and perturbs the data tuple. Before presenting the details of these components, we first introduce some notions of the training data set.

Let there be $m$ data providers in the system, each of which holds a private data tuple $t_i (i \in [1, m])$ and its class label attribute $a_0$. The private data tuple consists of $n$ attributes $a_1, \ldots, a_n$. The class label attribute is not sensitive and indicates which predefined class the data tuple belongs to. All other attributes are private information. The data miner has no external knowledge about the private information of the data providers.

In this paper, we assume there be two classes $C_0$ and $C_1$. As such, the class label attribute has two distinct values 0 and 1, corresponding to classes $C_0$ and $C_1$, respectively.

We first consider the case where all attributes are categorical (i.e., discrete-valued). If an attribute is continuous valued, it must be discretized first. An example of such discretion is provided in Section 6. Let the number of distinct values of $a_j$ be $s_j$. Without loss of generality, let $a_j \in \{0, \ldots, s_j - 1\}$. We denote a private data

tuple $t_i$ by an $(s_1 + \ldots + s_n)$-dimensional binary vector as follows.

$$t_i = \overbrace{0, \ldots, 1, \ldots, 0}^{s_1 \text{ bits for } a_1}, \ldots, \overbrace{0, \ldots, 1, \ldots, 0}^{s_n \text{ bits for } a_n} \quad (3)$$

In the $s_j$ bits for $a_j$, the $h$-th bit is 1 if and only if $a_j = h - 1$.

Although our scheme applies to all categorical attributes (with arbitrary $s_j$), for the simplicity of discussion, we assume that all attributes $a_1, \ldots, a_n$ are binary. That is, $s_1 = \cdots = s_n = 2$. As such, each private data tuple can be represented by a $2n$-dimensional vector. We represent the private part of the training data set by an $m \times 2n$ matrix $T = [t_1; \ldots; t_m]$.[3] We denote the transpose of $T$ by $T'$. We use $\langle T \rangle_{ij}$ to denote the element of $T$ with indices $i$ and $j$. Let $T_0$ and $T_1$ be the matrices that represent the private data tuples in class $C_0$ and $C_1$, respectively. We denote the number of data tuples in $T_i$ by $|T_i|$. An example of $T$ is shown in Table 1. As we can see from the matrix, data tuple $t_1$ belongs to class $C_1$ and has three attributes $[a_1, a_2, a_3] = [1, 0, 0]$. For the sake of completeness, we list the notions used in this paper in Appendix A as references.

**Table 1: An Example of the Training Data Set**

Class label:

| | $a_0$ |
|---|---|
| $t_1$ | 1 |
| $\vdots$ | $\vdots$ |
| $t_m$ | 0 |

$T$:

| | $a_1$ | $a_2$ | $a_3$ |
|---|---|---|---|
| $t_1$ | 0, 1 | 1, 0 | 1, 0 |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $t_m$ | 1, 0 | 0, 1 | 1, 0 |

## 4.1 Perturbation Guidance

As we are considering the case where data tuples are iteratively fed to the data miner, the data miner keeps a copy of all received data tuples and updates it when a new data tuple is received. Let the current matrix of received data tuples be $T^*$. When a new data tuple $R(t_i)$ is received by the data miner, $R(t_i)$ is appended to the bottom of $T^*$. Without loss of generality, we assume that data tuple $R(t_i)$ is the $i$-th data tuple that is received by the data miner. As such, when the data miner receives $m^*$ data tuples, $T^*$ is an $m^* \times 2n$ matrix $[t_1; \ldots; t_{m^*}]$. In order to compute the perturbation guidance for the first-come data provider, we assume that before the data collection process begins, the data miner already has $m_0$ ($n \leq m_0 \ll m$) data tuples in $T^*$. These data tuples can either be collected from privacy unconcerned data providers, or be randomly generated.

Besides the received data tuples $T^*$, the data miner also keeps track of two additional $2n \times 2n$ matrices: $A_0^* = T_0^{*\prime} T_0^*$ and $A_1^* = T_1^{*\prime} T_1^*$ where $T_0^*$ and $T_1^*$ are the matrices of received data tuples that belong to class $C_0$ and $C_1$, respectively. Note that the update of $A_0^*$ and $A_1^*$ (after $R(t_i)$ is received) does not need access to any data tuple other than the recently received $R(t_i)$. Thus, we do not require the matrix of received data tuples (i.e., $T^*$) to remain in main memory. If the class label attribute received with $R(t_i)$ satisfies $a_0 = c$ ($c \in \{0, 1\}$), $A_c^*$ is updated as follows.

$$A_c^* = A_c^* + R(t_i)' R(t_i). \quad (4)$$

Given $A_0^*$ and $A_1^*$, using eigen decomposition, we can decompose symmetric matrix $A^* = A_0^* - A_1^*$ as

$$A^* = V^* \Sigma^* V^{*\prime}, \quad (5)$$

where $\Sigma^* = \text{diag}(\sigma_1^*, \ldots, \sigma_{2n}^*)$ is a diagonal matrix with $\sigma_1^* \geq \cdots \geq \sigma_{2n}^*$, $\sigma_i^*$ is the $i$-th eigenvalue of $A^*$, and $V^*$ is an $2n \times 2n$ unitary matrix composed of the eigenvectors of $A^*$.

The perturbation guidance component has two objectives: to determine $k^*$, and to compute $V_k$ based on $k^*$. We address the computation of $k^*$ first. Roughly speaking, an appropriate choice of $k^*$ should be the minimum degree of freedom of $R(t_i)$ that maintains an accurate estimation of the eigenstructure of $A = T_0' T_0 - T_1' T_1$. Based on the eigen decomposition of $A^*$, we can compute $k^*$ as the minimum number that satisfies

$$\sigma_{k^*+1}^* \leq \mu \sigma_1^*, \quad (6)$$

where $\mu$ is a parameter predetermined by the data miner. A data miner that desires a highly accurate classifier can choose a small $\mu$ to ensure a stable estimation of $A$. A data miner that can tolerate a relatively lower level of accuracy can choose a large $\mu$ to help protecting data providers' privacy. In order to choose a good cutoff $k^*$ to retain enough information for building an accurate classifier, a simple textbook heuristic is to set $\mu = 15\%$.

Given $k^*$, $V_k^*$ is an $2n \times k^*$ matrix that is composed of the first $k^*$ eigenvectors of $A^*$ (i.e., the first $k^*$ column vectors of $V^*$, which correspond to the $k^*$ largest eigenvalues of $A^*$). In particular, if $V^* = [v_1, \ldots, v_{2n}]$, then $V_k^* = [v_1, \ldots, v_{k^*}]$. Since $V^*$ is a unitary matrix, we have $V_k^{*\prime} V_k^* = I$, where $I$ is the identity matrix.

We note that due to efficiency and privacy concern, the data miner only updates $k^*$ and $V_k^*$ once several data tuples are received. The efficiency concern is the overhead of computing $k^*$ and $V_k^*$. The privacy concern is that if $V_k^*$ is updated once every data tuple is received, a malicious data provider may infer the perturbed data tuple of another data provider from tracking the change of $V_k^*$. Although the victimized data provider is comfortable transmitting the perturbed data tuple to the data miner, it may not be comfortable divulging it to another data provider.

The justification of $k^*$ and $V_k^*$ will be provided in Section 5. The runtime efficiency of computing $k^*$ and $V_k^*$ will be addressed in Section 7. The communication overhead of transmitting $V_k^*$ will also be addressed in Section 7.

## 4.2 Perturbation

The perturbation component has two objectives: to check the validity of a received $V_k^*$, and to perturb $t_i$ based on $V_k^*$. Once a data provider receives $V_k^*$ from the data miner, the perturbation component first checks if $V_k^*$ is a $2n \times k^*$ matrix which satisfies $V_k^{*\prime} V_k^* = I$, where $I$ is the identity matrix. If so, the perturbation component perturbs the private data tuple $t_i$ based on $V_k^*$. The result is a perturbed data tuple that will be transmitted to the data miner along with class label attribute $a_0$. In our scheme, the perturbation is a two-step process. Recall that the private data tuple $t_i$ is represented as a $2n$-dimensional row vector. In the first step, $t_i$ is perturbed to be another $2n$-dimensional row vector $\tilde{t}_i$, such that

$$\tilde{t}_i = t_i V_k^* V_k^{*\prime}. \quad (7)$$

Since the elements in $\tilde{t}_i$ may be real values, we need a second step to transform $\tilde{t}_i$ to $R(t_i)$ such that every element in $R(t_i)$ belongs to $\{0, 1\}$. In particular, for any $j \in [1, 2n]$, the data provider generates a real number $r$ which is chosen uniformly at random from $[0, 1]$ and computes $R(t_i)$ as follows.

$$\langle R(t_i) \rangle_j = \begin{cases} 1, & \text{if } r \leq \langle \tilde{t}_i \rangle_j^2, \\ 0, & \text{otherwise,} \end{cases} \quad (8)$$

where $\langle \cdot \rangle_j$ is the $j$-th element of a vector. As we can see, the probability that $\langle R(t_i) \rangle_j = 1$ is equal to $\langle \tilde{t}_i \rangle_j^2$.

[3] In the context of training data set, $t_i$ is a data tuple. In the context of matrix, $t_i$ is the corresponding row vector in $T$.

The communication overhead of transmitting $R(t_i)$ will be addressed in Section 7.

# 5. PERFORMANCE ANALYSIS

In this section, we analyze our new scheme. We will define measures on 1) the error of classifiers built on the perturbed data set, and 2) the amount of privacy disclosure. We will derive bounds on them, in order to provide guidelines for the tradeoff between these two measures and hence help system managers setting parameters in practice.

## 5.1 Error Analysis

Given a testing data tuple $t : a_1, \ldots, a_n$ without class label, the objective of data classification is to identify $C_i$ that maximizes

$$P(C_i|t) = \frac{P(C_i, t)}{P(t)}. \quad (9)$$

As $P(t)$ is constant for all classes, the objective is to find $C_i$ that maximizes $P(C_i, t)$. Since $t$ contains $n$ attributes, the cost of computing $P(C_i, t)$ is too expensive. A common compromise is to compute $P(C_i, t)$ based on $P(C_i, t_\mathrm{s})$ where $t_\mathrm{s}$ is a small subset of $\{a_1, \ldots, a_n\}$. For example, in naïve Bayesian classification, the product of $P(C_i, a_j)$ is used to approximate $P(C_i, t)$. In decision tree classification, $P(C_i, t)$ is approximated by $P(C_i, t_\mathrm{s})$ where $t_\mathrm{s}$ is a set of selected test attributes, which correspond to the nodes in the decision tree.

For any data tuple $t$, given size $h \in [1, n]$, let $t_\mathrm{s}$ be a set of $h$ attributes of $t$. We measure the error of classifiers built on the perturbed data set in our scheme by the maximum estimation error of $P(C_0, t_\mathrm{s}) - P(C_1, t_\mathrm{s})$ after perturbation. Let the value of $P(C_i, t_\mathrm{s})$ estimated by the perturbed training data set be $\tilde{P}(C_i, t_\mathrm{s})$. The error on $P(C_0, t_\mathrm{s}) - P(C_1, t_\mathrm{s})$ is defined as

$$l_\mathrm{e}(h) = \max_{t, t_\mathrm{s}} |(P(C_0, t_\mathrm{s}) - P(C_1, t_\mathrm{s})) - (\tilde{P}(C_0, t_\mathrm{s}) - \tilde{P}(C_1, t_\mathrm{s}))|. \quad (10)$$

Given these notions, we define the degree of error as follows.

DEFINITION 1. *The degree of error $l_\mathrm{e}$ is defined as the maximum of $l_\mathrm{e}(h)$ on all sizes. That is,*

$$l_\mathrm{e} = \max_{h \in [1,n]} l_\mathrm{e}(h). \quad (11)$$

Generally speaking, the degree of error measures the discrepancy between classifiers constructed from the original data set and the perturbed data set.

Recall that $\mu$ is the pre-determined parameter used by the data miner to compute $k^*$. Recall that $A = T_0'T_0 - T_1'T_1$. We now derive an upper bound on the degree of error as follows.

THEOREM 5.1. *In our scheme, when $m$ is sufficiently large, there is*

$$l_\mathrm{e} \leq \frac{2\mu\sigma_1}{m}, \quad (12)$$

*where $\sigma_1$ is the largest eigenvalue of $A$.*

PROOF. (sketch) We note that for all $C_i$, if $t_s \subseteq t_s'$, there always be $P(C_i, t_s) \geq P(C_i, t_s')$. As such, we first prove that in the most difficult cases where $h \in \{1, 2\}$, there is $l_\mathrm{e}(h) \leq \mu\sigma_1/m$. In order to do so, we first show an intuitive explanation of $A$. Consider an element of $A$ with indices $2i$ and $2j$, we have

$$\langle A\rangle_{2i,2j} = \sum_{t \in C_0} \langle t\rangle_{2i}\langle t\rangle_{2j} - \sum_{t \in C_1} \langle t\rangle_{2i}\langle t\rangle_{2j} \quad (13)$$

$$= \#\{C_0, a_i = a_j = 1\} - \#\{C_1, a_i = a_j = 1\}, \quad (14)$$

where $\#\{C_i, a_i = a_j = 1\}$ is the number of data tuples that satisfy $a_i = a_j = 1$ and belong to $C_i$. Note that $\forall b_1, b_2 \in \{0, 1\}$, there is

$$\Pr\{C_i, a_i = b_1, a_j = b_2\} = \frac{\#\{C_i, a_i = b_1, a_j = b_2\}}{m}. \quad (15)$$

Generally, we have

$$\langle A\rangle_{2i-1+b_1,2j-1+b_2} = m \cdot (\Pr\{C_0, a_i = b_1, a_j = b_2\} - \Pr\{C_1, a_i = b_1, a_j = b_2\}). \quad (16)$$

As we can see, $l_\mathrm{e}(1)$ is in proportion to the maximum error on the estimate of the *diagonal* elements of $A$, $l_\mathrm{e}(2)$ is in proportion to the maximum error on the estimate of the other elements of $A$. Let the matrix of the perturbed training data set be $T_\mathrm{R}$. Let the corresponding $A$ derived from $T_\mathrm{R}$ be $A_\mathrm{R}$. We now derive an upper bound on $\max_{ij} |\langle A - A_\mathrm{R}\rangle_{ij}|$.

Recall that in the first step of the perturbation, a data provider computes $\tilde{t}_i = t_i V_k^* V_k^{*\prime}$. Let $\tilde{T}$ be an $m \times 2n$ matrix composed of $\tilde{t}_i$ (i.e., $\tilde{T} = [\tilde{t}_1; \ldots; \tilde{t}_m]$). $\tilde{T}_i$ and $\tilde{A}$ are defined correspondingly. Due to our computation of $\tilde{t}_i$, we have $\tilde{T}_i = T_i V_k^* V_k^{*\prime}$. In our scheme, $V_k^*$ is the first $k^*$ eigenvectors of the current $A^*$. For the simplicity of discussion, we consider $V_k^*$ as the first $k^*$ eigenvectors of $A$. In real cases, the first $k^*$ eigenvectors of $A^*$ converge to those of $A$ fairly quickly.

Let $\Sigma_k^*$ be a $k^* \times k^*$ diagonal matrix in which the diagonal elements are the first $k^*$ eigenvalues of $A$ (i.e., the diagonal of $\Sigma_k^*$ is $[\sigma_1, \ldots, \sigma_{k^*}]$). We have

$$\tilde{A} = \tilde{T}_0'\tilde{T}_0 - \tilde{T}_1'\tilde{T}_1 \quad (17)$$

$$= V_k^* V_k^{*\prime} T_0' T_0 V_k^* V_k^{*\prime} - V_k^* V_k^{*\prime} T_1' T_1 V_k^* V_k^{*\prime} \quad (18)$$

$$= V_k^* V_k^{*\prime} (T_0'T_0 - T_1'T_1) V_k^* V_k^{*\prime} \quad (19)$$

$$= V_k^* V_k^{*\prime} A V_k^* V_k^{*\prime} \quad (20)$$

$$= V_k^* \Sigma_k^* V_k^{*\prime} \quad (21)$$

That is, $\tilde{A}$ is the $k^*$-truncation of $A$ [9]. Thus, $\tilde{A}$ is the optimal rank-$k^*$ approximation of $A$ in the sense that within all rank-$k^*$ matrices, $\tilde{A}$ has the minimum $\|A - \tilde{A}\|_2$. In particular, we have

$$\|A - \tilde{A}\|_2 = \sigma_{k^*+1}. \quad (22)$$

As we can see from the determination on disclosure level $k^*$, our scheme maintains a cutoff $k^*$ such that $\sigma_{k^*+1} \leq \mu\sigma_1$. Thus, we have

$$\|A - \tilde{A}\|_2 \leq \mu\sigma_1. \quad (23)$$

Since the absolute value of every element of a matrix is no larger than the 2-norm of the matrix [9], we have

$$\max_{i,j \in [1,2n]} |\langle A - \tilde{A}\rangle_{ij}| \leq \|A - \tilde{A}\|_2 \leq \mu\sigma_1. \quad (24)$$

As we can see from the computation of $R(t)$, for any $i, j \in [1, 2n]$, we have

$$\langle \tilde{t}\rangle_i^2 = \mathrm{Exp}(\langle R(t)\rangle_i^2), \quad (25)$$

$$\mathrm{Exp}(\langle t\rangle_i\langle t\rangle_j - \langle R(t)\rangle_i\langle R(t)\rangle_j) \leq 2(\langle t\rangle_i\langle t\rangle_j - \langle \tilde{t}\rangle_i\langle \tilde{t}\rangle_j), \quad (26)$$

where $\mathrm{Exp}(\cdot)$ refers to the expected value. Thus, when $m$ is sufficiently large, we have $l_\mathrm{e}(h) \leq 2\mu\sigma_1/m$ for $h \in \{1, 2\}$. This bound can be easily extended to $l_\mathrm{e}(h)$ with $h \geq 3$. We omit the proof here due to space limit. □

## 5.2 Privacy Analysis

In our scheme, we need to guarantee that for any private training data tuple $t$, the data miner cannot deduce the original $t$ from the

perturbed $R(t)$. In particular, we must consider the case when the adversary manipulates $V_k^*$ to compromise the privacy of the data providers.

Recall that our scheme allows different data providers to choose different disclosure levels. Thus, we define privacy disclosure measure on individual data providers. Formally, let the maximum acceptable disclosure level selected by a data provider be $k_i$. For any $2n \times k^*$ matrix $\hat{V}_k^*$, let $\hat{R}(t, \hat{V}_k^*)$ be the output of $R(t_i)$ when $t_i = t$ and $V_k^* = \hat{V}_k^*$. With these notions, we define the degree of privacy disclosure as follows.

DEFINITION 2. *The degree of privacy disclosure, $l_{\rm p}(k_i)$, is defined by the maximum fraction of private information disclosed by the perturbed data tuple when the data miner sends a arbitrary $2n \times k^*$ matrix $\hat{V}_k^*$ as the perturbation guidance. That is,*

$$l_{\rm p}(k_i) = \max_{\hat{V}_k^* | k^* \le k_i} \left( \frac{I(t; \hat{R}(t, \hat{V}_k^*))}{H(t)} \right) \tag{27}$$

$$= \max_{\hat{V}_k^* | k^* \le k_i} \left( 1 - \frac{H(t|\hat{R}(t, \hat{V}_k^*))}{H(t)} \right), \tag{28}$$

*where $I(t; \hat{R}(t, \hat{V}_k^*)$ is the mutual information [5] between $t$ and $\hat{R}(t, \hat{V}_k^*)$, $H(\cdot)$ denotes the information entropy.*

In the definition, $I(t; \hat{R}(t))$ measures the amount of private information about $t$ that is disclosed by $\hat{R}(t, \hat{V}_k^*)$. $H(t)$ measures the amount of information in $t$. Thus, the degree of privacy disclosure measures the percentage of private information that is disclosed by $\hat{R}(t, \hat{V}_k^*)$.

THEOREM 5.2. *In our scheme, we have*

$$l_{\rm p}(k_i) < \frac{\rho_1^2 + \cdots + \rho_{k_i}^2}{mn}, \tag{29}$$

*where $\rho_j$ is the $j$-th singular value of $T$.*

PROOF. (sketch) Consider the matrix $T\hat{V}_k^*\hat{V}_k^{*\prime}$ that consists of $\tilde{t}_i = t_i\hat{V}_k^*\hat{V}_k^{*\prime}$. Given any $2n \times k$ matrix $\hat{V}_k^*$, the rank of $\hat{V}_k^*$ is no larger than $k^*$. Thus, the rank of $T\hat{V}_k^*\hat{V}_k^{*\prime}$ is less than or equal to $k^*$. We have

$$\|T - T\hat{V}_k^*\hat{V}_k^{*\prime}\|_F \ge \sqrt{\rho_{k^*+1}^2 + \cdots + \rho_{2n}^2}. \tag{30}$$

where $\|\cdot\|_F$ is the Frobenius norm of a matrix (i.e., the square root of the sum of the squares of its elements). Note that we have $\hat{V}_k^{*\prime}\hat{V}_k^* = I$ because the data providers checks the validity of $\hat{V}_k^*$ before using it to perturb the private data. As such, almost all $\langle\tilde{t}_i\rangle_j$ are within $[0, 1]$. Given our computation of $R(t_i)$ based on $\tilde{t}_i$, we have

$$\text{if } \langle t_i\rangle_j = 0, \quad \text{Exp}((\langle R(t_i)\rangle_j - \langle t_i\rangle_j)^2)$$
$$= \langle\tilde{t}_i\rangle_j^2 = (\langle\tilde{t}_i\rangle_j - \langle t_i\rangle_j)^2 \tag{31}$$
$$\text{if } \langle t_i\rangle_j = 1, \quad \text{Exp}((\langle R(t_i)\rangle_j - \langle t_i\rangle_j)^2)$$
$$= 1 - \langle\tilde{t}_i\rangle_j^2 > (1 - \langle\tilde{t}_i\rangle_j)^2 = (\langle\tilde{t}_i\rangle_j - \langle t_i\rangle_j)^2, \tag{32}$$

where $\text{Exp}(\cdot)$ refers to the expected value. Consider the transformation of the value of an element in $t_i$ from $\langle t_i\rangle_j$ to $\langle R(t_i)\rangle_j$. Since $k \ll n$ in most cases, the number of transformation from 1 to 0 is much larger than that of transformation from 0 to 1. Recall that $T_{\rm R}$ is the matrix of perturbed data tuples. We have

$$\|T - T_{\rm R}\|_F > \|T - T\hat{V}_k^*\hat{V}_k^{*\prime}\|_F \tag{33}$$

That is, the number of elements in $T_{\rm R}$ that are equal to 1 is less than $\rho_1^2 + \cdots + \rho_{k^*}^2$.

As such, for an attribute $a_j$ of a data tuple $t_i$ in $T\hat{V}_k^*\hat{V}_k^{*\prime}$, the probability that $\langle t_i\rangle_{2j-1} = \langle t_i\rangle_{2j} = 0$ is greater than $1 - (\rho_1^2 + \cdots + \rho_{k^*}^2)/mn$. With some mathematical manipulation, we have

$$I(t; \hat{R}(t, \hat{V}_k^*)) < \frac{\rho_1^2 + \cdots + \rho_{k^*}^2}{mn} H(t). \tag{34}$$

Since $k^* \le k_i$, the degree of privacy disclosure satisfies

$$l_{\rm p}(k_i) \le \frac{\rho_1^2 + \cdots + \rho_{k_i}^2}{mn}. \tag{35}$$

□

# 6. EXPERIMENTAL RESULTS

In this section, we first compare the performance of our scheme with that of the randomization approach. After that, we present the simulation results of our scheme on a real data set.

In order to make a fair comparison between the performance of our scheme and that of the randomization approach, we use the exactly same training and testing data sets as in [2]. Due to space limit, please refer to [2] for a detailed description of the training data set and the classification functions. The training data set consists of $100,000$ data tuples. The testing data set consists of $5,000$ data tuples. Each data tuple has nine attributes including seven continuous attributes and two categorical attributes (i.e., elevel, zipcode). Five widely varied classification functions are used to measure the tradeoff between accuracy and privacy in different circumstances. The randomization approach used is a combination of ByClass distribution reconstruction algorithm with Gaussian randomization operator, which performs the best in our experiment compared to other combinations proposed in [2] (i.e., combination of ByClass or Local algorithm with uniform or Gaussian distribution). We use the same classification algorithm, ID3 decision tree algorithm, as in [2].

Since our scheme assumes that the data set contains only categorical data, we first transform the original continuous data to categorical. We split the value of each continuous attribute into four intervals based on its 1st quartile (i.e., $25\%$ percentile), median, and 3rd quartile (i.e., $75\%$ percentile). As such, each continuous attribute is transformed to a categorical attribute with $4$ distinct values. Since the two categorical attributes have $5$ and $9$ distinct values, respectively, each private data tuple is represented by a $42$-dimensional binary vector ($\sum_j s_j = 4 \times 7 + 5 + 9 = 42$) after preprocessing.

To demonstrate the accuracy of classification results intuitively, we compare the percentage of testing data tuples that are correctly classified by the decision trees built upon the perturbed training data set generated by our scheme and the randomization approach. The comparison of the predictive accuracy while fixing the expected degree of privacy disclosure at $25\%$ is shown in Figure 3. Since different data providers may choose different disclosure levels in our scheme, we compute the expected degree of privacy disclosure of our scheme as the average for all data providers. In our scheme, the data miner updates system disclosure level $k^*$ and perturbation guidance $V_k^*$ once $100$ data tuples are received. As we can see, while both approaches perform perfectly on Function 1, our scheme outperforms the randomization approach on the other four functions.

To demonstrate the transparency of our scheme to the classification algorithms, we simulate our scheme using naïve Bayesian classifier on a real data set. We use the congressional voting records

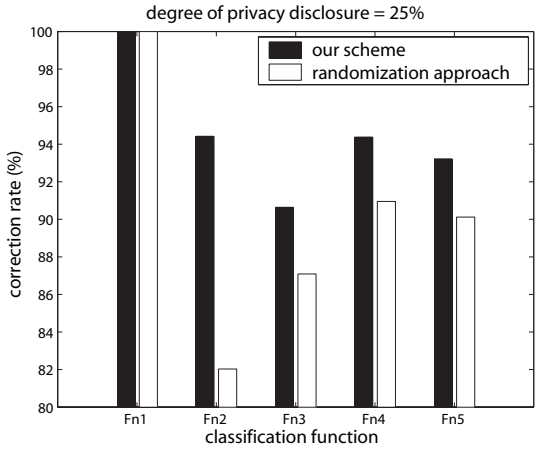


**Figure 3: Comparison of Performance**

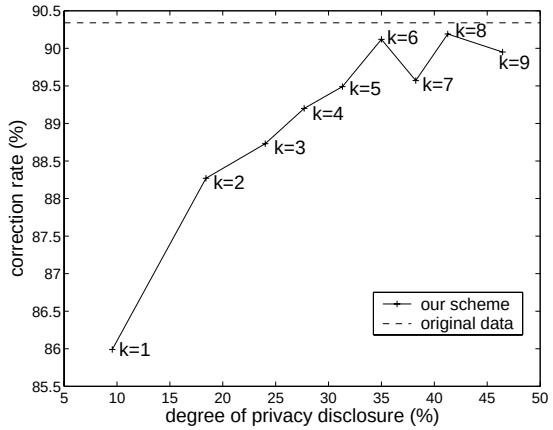


**Figure 4: Naïve Bayesian Classification on Real Data Set**

database from the UCI machine learning repository [4]. The original source of data is Congressional Quarterly Almanac, 98th Congress, 2nd session, 1984. The data set was donated by Jeff Schlimmer in 1987. The data set includes 16 key votes for each of the U.S. House of Representatives congressmen. It includes 435 records with 16 attributes (all of which are binary) and a class label describing whether the congressman is a democrat or republican. There are $61.38\%$ democrats and $38.62\%$ republicans in the data set. The goal of classification is to determine the party affiliation based on the votes. There are 392 missing values in the data set, which we substitute with values chosen uniformly at random from $\{0, 1\}$.

Since each data tuple has 16 binary private attributes, each data tuple is represented by a 32-dimensional binary vector. We first apply naïve Bayesian classification on the original data set to build a naïve Bayesian classifier. We then apply our scheme on 9 different degrees of privacy disclosure and build 9 classifiers on the perturbed data sets. After that, we apply the same testing data set to all 10 classifiers and compare their predictive accuracy. The predictive accuracy of the classifier built on the original data set is $90.34\%$. In order to demonstrate the role of disclosure level $k_i$ in our scheme, we simulate our scheme when all data providers choose the same disclosure level $k_i = k$. The predictive accuracy of classifiers built on perturbed data sets are shown in Figure 4. As we can see from the figure, the naïve Bayesian classifier built on the perturbed data set can predict the class label with correction rate of $85.99\%$ when the degree of privacy disclosure is $9.56\%$. Thus, our classification can effectively preserve privacy while keeping the classifier predictively accurate.

To demonstrate that the system disclosure level $k^*$ decreases rapidly during the collection of data tuples, we perform another simulation while fixing the parameter $\mu$, which is used to compute $k^*$. Recall that generally speaking, the lower $\mu$ is, the more information is retained for building a more accurate classifier. For a given $\mu$, we investigate the change of $k^*$ with the number of data tuples received by the data miner (i.e., $|T^*|$). In most cases, $k^*$ decreases to be very small fairly soon. For example, when $\mu = 15\%$, $k^*$ decreases to be 2 after 50 data tuples are received. Figure 5 shows the change of $k^*$ with $|T^*|$ when the degree of error is required to be very small ($\mu = 2.5\%$). As we can see, even when the error is strictly bounded, $k^*$ still decreases fairly quickly.

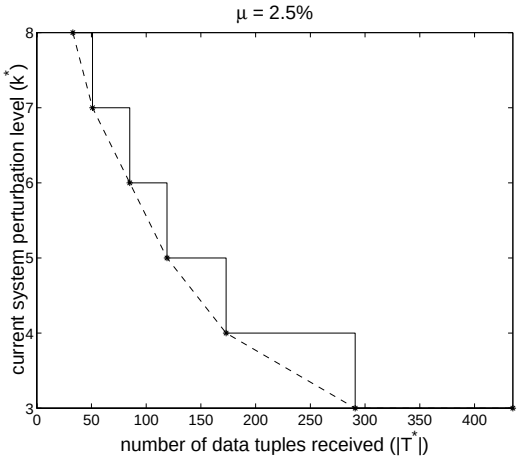


**Figure 5: Change of $k^*$ with $|T^*|$**

## 7. IMPLEMENTATION

A prototypical system for privacy-preserving data classification has been implemented using our new scheme. The goal of the system is to deliver an online survey solution that preserves the privacy of survey respondents. The survey collector/analyzer and the survey respondents are modeled as the data miner and the data providers, respectively. The system consists of a perturbation guidance component on web servers and a data perturbation component on web browsers. Both components are implemented as custom plug-ins that one can easily install to existing systems. The architecture of our system is shown in Figure 6.

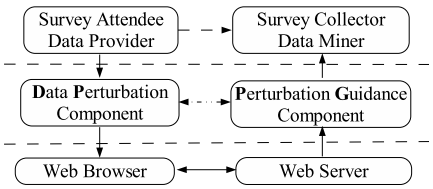


**Figure 6: System Implementation**

As is shown in the figure, there are three separate layers in our system: user interface layer, perturbation layer, and web layer. The top layer, named user interface layer, provides interface to data providers and the data miner. The middle layer, named perturba-

tion layer, realizes our privacy-preserving scheme and exploits the bottom layer to transfer information. In particular, before $V_k^*$ is determined, the data perturbation component encrypts the private data and caches it on the client machine. When $V_k^*$ is received, the data perturbation component decrypts the cached data, perturbs it, and transmits the perturbed data tuple to the data miner. The bottom layer, named web layer, consists of web servers and web browsers. As an important feature of our system, the details of data perturbation on the middle layer are transparent to both data providers and the data miner.

## 7.1 Runtime Efficiency

We now compare the runtime efficiency of our scheme with that of the randomization approach. As we have addressed in Section 2, it is shown in [3] that the cost of mining randomized data set is "*well within an order of magnitude*" in respect to that of mining the original data set. In particular, the randomization approach proposed in [2] requires the original data distribution to be reconstructed before a decision tree classifier can be built on the randomized data set. The distribution reconstruction is a three-step process. We use "ByClass" reconstruction algorithm as an example because, as stated in [2], it is a tradeoff between accuracy and efficiency.

In the first step, split points are determined to partition the domain of each attribute into intervals. There is an estimated number of data points in each interval. The second step partitions data values into different intervals. For each attribute, the values of randomized data are sorted to be associated with an interval. In the third step, for each attribute, the original distribution is reconstructed for each class separately. The main purpose of the first two steps is to accelerate the computation of the third step. The time complexity of the algorithm is $O(mn + nv^2)$ where $m$ is the number of training data tuples, $n$ is the number of private attributes in a data tuple, and $v$ is the number of intervals on each attribute. It is assumed in [2] that $10 \leq v \leq 100$.

Note that the overhead of the randomization approach occurs on the critical time path. Since the distribution reconstruction is not an incremental algorithm, it has to be performed after all data tuples are collected and before the classifier is constructed. Besides, the distribution reconstruction algorithm requires access to the whole training data set, some of which may not be stored in the main memory. This problem may incur even more serious overhead.

In our scheme, the perturbed data tuples are directly used to construct the classifier. The only overhead incurred on the data miner is to update the system disclosure level $k^*$ and perturbation guidance $V_k^*$. Note that the overhead is not on the critical time path. Instead, it occurs during the collection of data. The time complexity of the updating process is $O(n^2)$. As we mentioned in Section 4 and demonstrated in Section 6, the data miner may only need to update $k^*$ and $V_k^*$ once several data tuples are received. Since the number of attributes is much less than the number of data tuples (i.e., $n \ll m$) in data classification, the overhead of our scheme is significantly less than the overhead of the randomization approach.

Our scheme is scalable to very large training data sets. As we can see, the space complexity of computing $k^*$ and $V_k^*$ is $O(n^2)$. That is, the received data tuples need not to remain in the main memory.

Since for most data providers, the disclosure level $k^*$ is a small number (a heuristic average value of $k^*$ is an order less than $n$), the communication overhead ($O(nk^*)$ per data provider) incurred by the two-way communication in our scheme is not significant. There may be concern on the upstream traffic from the data providers to the data miner when there are many distinct values for each attribute of the data tuple. In this case, the sparse nature of $R(t_i)$ provides an efficient way to encode $R(t_i)$ to a list of nonzero elements such that the overhead of transmitting $R(t_i)$ can be substantially reduced.

# 8. FINAL REMARKS

In this paper, we propose a new scheme on privacy-preserving data classification. Compared with previous approaches, we introduce a two-way communication mechanism between the data miner and the data providers with little overhead. In particular, we let the data miner send perturbation guidance to the data providers. Using this intelligence, data providers perturb their data tuples to be transmitted to the data miner. As a result, our scheme has the benefit of a better tradeoff between accuracy and privacy.

Our work is preliminary and many extensions can be made. We are currently investigating how to apply our scheme to clustering problem. We would also like to investigate the integration of our scheme with cryptographic techniques.

# Acknowledgement

We thank the anonymous reviewers for their insightful comments that helped us improve the quality of the paper.

# 9. REFERENCES

[1] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pages 247–255. ACM Press, 2001.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proceedings of the 19th ACM SIGMOD International Conference on Management of Data*, pages 439–450. ACM Press, 2000.

[3] S. Agrawal, V. Krishnan, and J. R. Haritsa. On addressing efficiency concerns in privacy-preserving mining. In *Proceedings of the 9th International Conference on Database Systems for Advanced Applications*, pages 439–450. Springer Verlag, 2004.

[4] C. Blake and C. Merz. UCI repository of machine learning databases, 1998.

[5] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, 1991.

[6] L. Cranor, J. Reagle, and M. S. Ackerman. Beyond concern: Understanding net users' attitudes about online privacy. Technical Report TR 99.4.3, AT&T Labs-Research, 1999.

[7] W. Du and Z. Zhan. Building decision tree classifier on private data. In *Proceedings of the IEEE International Conference on Privacy, Security and Data Mining*, pages 1–8. Australian Computer Society, Inc., 2002.

[8] 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*, pages 505–510. ACM Press, 2003.

[9] G. H. Golub and C. F. V. Loan. *Matrix Computation*. John Hopkins University Press, 1996.

[10] J. Han and M. Kamber. *Data Mining Concepts and Techniques*. Morgan Kaufmann, 2001.

[11] HIPAA. Health insurance portability and accountability act, 2002. available at http://www.hhs.gov/ocr/hipaa/privrulepd.pdf.

[12] M. Kantarcioglu and J. Vaidya. Privacy preserving naïve bayes classifier for horizontally partitioned data. In *Workshop on Privacy Preserving Data Mining held in association with The 3rd IEEE International Conference on Data Mining*, 2003.

[13] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 99–106. IEEE Press, 2003.

[14] Y. Lindell and B. Pinkas. Privacy preserving data mining. In *Proceedings of the 20th Annual International Cryptology Conference on Advances in Cryptology*, pages 36–54. Springer Verlag, 2000.

[15] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[16] J. Vaidya and C. Clifton. Privacy preserving naïve bayes classifier for vertically partitioned data. In *Proceedings of the 4th SIAM Conference on Data Mining*, pages 330–334. SIAM Press, 2004.

[17] N. Zhang, S. Wang, and W. Zhao. A new scheme on privacy preserving association rule mining. In *Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*. Springer Verlag, 2004.

# APPENDIX

## Appendix A. NOTIONS

**Table 2: Notions**

| | |
|---|---|
| $m$ | number of data providers |
| $n$ | number of private attributes |
| $t_i$ | private data tuple |
| $a_0$ | class label attribute |
| $a_1, \ldots, a_n$ | private attributes |
| $s_j$ | number of distinct values of $a_j$ |
| $C_0, C_1$ | classes |
| $T$ | matrix of all private data tuples |
| $T_i$ | matrix of data tuples in $C_i$ |
| $T_R$ | matrix of perturbed data tuples |
| $A$ | $T_0'T_0 - T_1'T_1$ |
| $A_i$ | $T_i'T_i$ |
| $\sigma_i$ | $i$-th largest eigenvalue of $A$ |
| $\rho_i$ | $i$-th singular value of $T$ |
| $k_i$ | maximum acceptable perturbation level of a data provider |
| $k^*$ | current system perturbation level |
| $V_k^*$ | current perturbation guidance |
| $\mu$ | a predetermined parameter on computing $k^*$ |
| superscript $'$ | transpose of a matrix or vector |
| superscript $*$ | current version of a matrix or variable |
| $\langle\cdot\rangle_j$ | $j$-th element of a vector |
| $\langle\cdot\rangle_{ij}$ | the element of a matrix with indices $i$ and $j$ |