**Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining (CIDM 2007)**

# Using Homomorphic Encryption For Privacy-Preserving Collaborative Decision Tree Classification

Justin Zhan
Carnegie Mellon University
justinzh@andrew.cmu.edu

## Abstract

*To conduct data mining, we often need to collect data from various parties. Privacy concerns may prevent the parties from directly sharing the data. A challenging problem is how multiple parties collaboratively conduct data mining without breaching data privacy. The goal of this paper is to provide solutions for privacy-preserving decision tree classification which is one of data mining tasks. Our goal is to obtain accurate data mining results without disclosing private data.*

**Key Words:** Privacy, Data Mining, Decision Tree Classification.

## 1. Introduction

Recent advances in data collection, data dissemination and related technologies have inaugurated a new era of research where existing data mining algorithms should be reconsidered from the point of view of privacy preservation [20]. The need for privacy is sometimes due to law (e.g., for medical databases) or can be motivated by business interests. However, there are situations where the sharing of data can lead to mutual benefit. For example, there are several biomedical labs involved into a multi-site medical study. Each lab has its own data set containing medical records. These labs would like to conduct data mining over the data sets from all of labs with the goal of more valuable information would be obtained via mining the joint data set. Because of legal privacy rules, one lab cannot disclose their private records to other labs. How can these labs achieve their goal? In this paper, we present technologies to solve privacy-preserving collaborative decision tree classification problem over large data sets with reasonable efficiency.

The paper is organized as follows: We present our privacy definition in section 2. Our technologies are provided in section 3 and 4. Thereafter, we describe our problem and privacy-oriented protocols for privacy-preserving collaborative decision tree classification in section 5, section 6 and section 7. We present some related works and our conclusions in section 8.

## 2 Definition of Privacy

We presented a definition of privacy in [19, 20]. In this paper, we follow the same concept of privacy which is described as follows:

**Definition 1** *A privacy-oriented scheme $S$ preserves data privacy if for any private data $T$, the following is held:*

$$|Pr(T|PPDMS) - Pr(T)| \leq \epsilon$$

*where*

- $PPDMS$*: Privacy-preserving data mining scheme.*
- $\epsilon$*: A probability parameter.*
- $Pr(T|PPDMS)$*: The probability that the private data T is disclosed after privacy-preserving data mining schemes being applied.*
- $Pr(T)$*: The probability that the private data T is disclosed without any privacy-preserving data mining scheme being applied.*
- $Pr(T|PPDMS) - Pr(T)$*: The probability that private data T is disclosed with and without privacy-preserving data mining schemes being applied.*

*We call $\epsilon$ the privacy level that the privacy-oriented scheme $S$ can achieve. The goal is to make $\epsilon$ as small as possible.*

We have defined privacy for data mining algorithms. However, often time, we need to reduce the whole privacy-preserving data mining algorithm to a set of component privacy-oriented protocols. We say the privacy-preserving data mining algorithm preserves privacy if each component protocol preserves privacy and the combination of the component protocols do not disclose private data. In the secure

multiparty computation literature, a composition theorem [6] describes the similar idea.

**Theorem 1** *Suppose that g is privately reducible to f and that there exists a protocol for privately computing f. Then there exists a protocol for privately computing g.*

**Proof 1** *Refer to [6].*

We now formally define privacy for a component protocol.

**Definition 2** *A privacy-oriented component protocol $CP$ preserves data privacy if for any private data $T$, the following is held:*

$$|Pr(T|CP) - Pr(T)| \leq \epsilon$$

*where*

- $CP$*: Component protocol.*
- $Pr(T|CP)$*: The probability that the private data T is disclosed after a privacy-preserving component protocol being applied.*
- $Pr(T|CP) - Pr(T)$*: The probability that private data T is disclosed with and without privacy-preserving component protocol.*

*We call $\epsilon$ the privacy level that the privacy-oriented component protocol $CP$ can achieve. The goal is to make $\epsilon$ as small as possible.*

Next, we introduce homomorphic encryption and digital envelope technique.

## 3 Homomorphic Encryption

The concept of homomorphic encryption was originally proposed in [14]. Since then, many such systems have been proposed. In this paper, we base our privacy-oriented protocols on [10] which is semantically secure. In our privacy-oriented protocols, we use additive homomorphism offered by [10] in which Paillier proposed a new trapdoor mechanism based on the idea that it is hard to factor number n = pq where p and q are two large prime numbers. In the performance evaluation, Paillier compares the proposed encryption scheme with existing public-key cryptosystems. The results show that the encryption process is comparable with encryption process of RSA in terms of the computation cost; the decryption process is faster than the decryption process of RSA.

In this paper, we utilize the following property of the homomorphic encryption functions: $e(m_1) \times e(m_2) = e(m_1 + m_2)$ where $m_1$ and $m_2$ are the data to be encrypted. Because of the property of associativity, $e(m_1 + m_2 + .. + m_n)$ can be computed as $e(m_1) \times e(m_2) \times \cdots \times e(m_n)$ where $e(m_i) \neq 0$. That is

$$e(m_1 + m_2 + \cdots + m_n) = e(m_1) \times e(m_2) \times \cdots \times e(m_n). \quad (1)$$

$$d(e(m_1)^{m_2}) = d(e(m_1 \times m_2)), \quad (2)$$

where $\times$ denotes multiplication.

## 4 Digital Envelope

A digital envelope [2] is a random number (or a set of random numbers) only known by the owner of private data. To hide the private data in a digital envelope, we conduct a set of mathematical operations between a random number (or a set of random numbers) and the private data. The mathematical operations could be addition, subtraction, multiplication, etc. For example, assume the private data value is $v$. There is a random number R which is only known by the owner of $v$. The owner can hide $v$ by adding this random number, e.g., $v + R$.

## 5 Privacy-Preserving Collaborative Decision Tree Classification Problems

In this paper, we consider the privacy-preserving collaborative decision tree classification problem which is defined as follows:

**Problem 1** *$P_1$ has a private data set $DS_1$, $P_2$ has a private data set $DS_2$, $\cdots$ and $P_n$ has a private data set $DS_n$. The data set $[DS_1 \cup DS_2 \cup \cdots \cup DS_n]$ forms a dataset, which is actually the concatenation of $DS_1$, $DS_2$, $\cdots$ and $DS_n$. The n parties want to conduct decision tree classification over $[DS_1 \cup DS_2 \cup \cdots \cup DS_n]$ to obtain the mining results satisfying the given constraints.*

There are two types of collaborative models. In the vertical collaboration, diverse features of the same set of data are collected by different parties. In the horizontal collaboration, diverse sets of data, all sharing the same features, are gathered by different parties. The collaborative model that we consider is the *vertical collaboration*.

## 6. Privacy-Preserving Decision Tree Classification

The decision tree is one of the classifiers. The induction of decision trees [11, 12, 13] from attribute vectors is an

important and fairly explored machine learning paradigm. The decision tree representation is the most widely used method. There is a large number of decision tree induction algorithms described in the machine learning and applied statistics literature. In general, the decision tree is built in a top-down style, using a greedy strategy to choose, based on the instances corresponding to the sub-tree in construction, the root of this sub-tree.

A decision tree is a class discriminator that recursively partitions the training set until each partition entirely or dominantly consists of examples from one class. A well known algorithm for building decision tree classifiers is ID3 [11]. We describe the algorithm below where $S$ represents the training instances and $AL$ represents the attribute list:

**Algorithm 1** . **ID3**($S$, $AL$)

1. Create a node V.
2. **If** $S$ consists of instances with all the same class C **then** return V as a leaf node labelled with class C.
3. **If** $AL$ is empty, **then** return V as a leaf-node with the majority class in $S$.
4. Select test attribute ($TA$) among the $AL$ with the highest information gain.
5. Label node V with $TA$.
6. **For** each known value $a_i$ of $TA$
   (a) Grow a branch from node V for the condition $TA = a_i$.
   (b) Let $s_i$ be the set of instances in $S$ for which $TA = a_i$.
   (c) **If** $s_i$ is empty **then** attach a leaf labeled with the majority class in $S$.
   (d) **Else** attach the node returned by **ID3**($s_i$, $AL - TA$).

According to ID3 algorithm, each non-leaf node of the tree contains a splitting point, and the main task for building a decision tree is to identify an attribute for the splitting point based on the information gain. Information gain can be computed using *entropy*. In the following, we assume there are $nc$ number of classes in the whole training data set. $Entropy(S)$ is defined as follows:

$$Entropy(S) = -\sum_{j=1}^{nc} Q_j \log Q_j, \tag{3}$$

where $Q_j$ is the relative frequency of class $j$ in $S$. Based on the entropy, we can compute the information gain for any candidate attribute A if it is used to partition $S$:

$$Gain(S, A) = Entropy(S) - \sum_{v \in A} (\frac{|S_v|}{|S|} Entropy(S_v)), \tag{4}$$

where $v$ represents any possible values of attribute A; $S_v$ is the subset of $S$ for which attribute A has value v; $|S_v|$ is the number of elements in $S_v$; $|S|$ is the number of elements in S. To find the best split for a tree node, we compute information gain for each attribute. We then use the attribute with the largest information gain to split the node.

To build a decision tree classifier, we have to decide and assign an attribute for each node. In order to determine each node for the tree, we need to conduct the following steps:

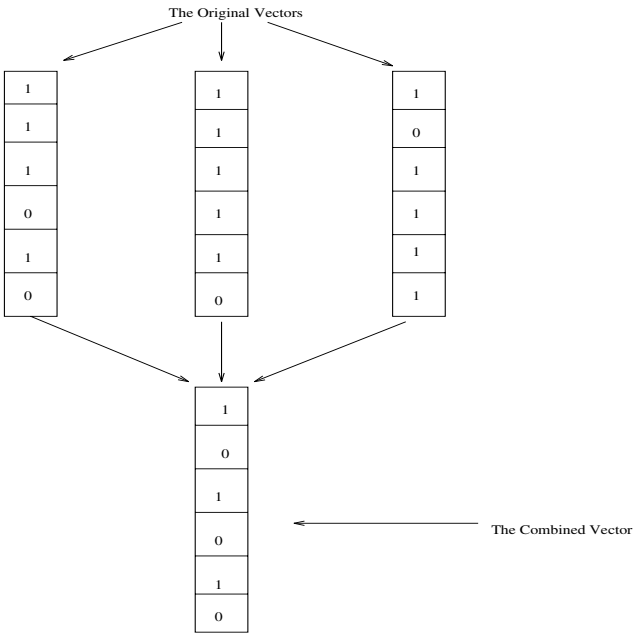
1. To compute $Entropy(S_v)$.
2. To compute $\frac{|S_v|}{|S|}$.
3. To compute $\frac{|S_v|}{|S|} Entropy(S_v)$.
4. To compute information gain for each candidate attribute.
5. To compute the attribute with the largest information gain.

Next, we will provide privacy-oriented protocols to conduct each step in the scenarios of vertical collaboration.

## 7 Privacy-Preserving Protocols

Each party has a private data set. The data set normally contains many attributes. For example, $P_1$ has $DS_1$ which includes five attributes denoted by $A_{11}, A_{12}, \cdots, A_{15}$ respectively. To compute $Q_j$, we need to compute the frequency count involving all the private data. Each party can first join their own attribute vectors and obtain a single vector. For example, in Figure 1, there are three attribute vectors which are denoted by $A_{i1}$, $A_{i2}$ and $A_{i3}$, after being combined, they reduce to a single vector which is denoted by $A_i$. The combination process is as follows: if $A_{i1} = 1$, $A_{i2} = 1$ and $A_{i3} = 1$, then $A_i = 1$; otherwise, $A_i = 0$. The cross-parties computation will solely use this combined vector (Figure 1).

We first select a key generator who produces the encryption and decryption key pairs. Let us assume that $P_n$ is the key generator who generates a homomorphic encryption key pair (e, d). Next, we will show how to conduct each step.




$e(Entropy(S_v))$

**Highlight of Protocol 1:** There are three steps in the protocol. In step I, $P_{n-1}$ obtains $e(Q_j)$. In this step, $P_n$ sends $e(A_{ni})$ to $P_1$ who computes $e(A_{n1}A_{11})$ then sends it to $P_2$. Continue this process until $P_{n-1}$ obtains $e(A_{1i}A_{2i}\cdots A_{ni})$, $i \in [1, N]$. $P_{n-1}$ then computes $e(Q_j) = e(A_{11}A_{21}\cdots A_{n1} + A_{12}A_{22}\cdots A_{n2} + \cdots + A_{1N}A_{2N}\cdots A_{nN})^{\frac{1}{N}} = e(A_{11}A_{21}\cdots A_{n1}) \times e(A_{12}A_{22}\cdots A_{n2}) \times \cdots \times e(A_{1N}A_{2N}\cdots A_{nN})^{\frac{1}{N}}$.

In step II, $P_{n-1}$ computes $e(Q_j log(Q_j))$. In this step, $P_{n-1}$ first generates set of random numbers $R_1, R_2, \cdots$, and $R_t$, then sends the sequence of $e(Q_j)$, $e(R_1)$, $e(R_2)$, $\cdots$, $e(R_t)$ to $P_n$ in a random order. $P_n$ decrypts each element in the sequence, computes, then sends $log(Q_j)$, $log(R_1)$, $log(R_2)$, $\cdots$, $log(R_t)$ to $P_1$ in the same order as $P_{n-1}$ did. $P_1$ and $P_{n-1}$ compute $e(Q_j log(Q_j))$. Step I and step II are repeated to obtain $e(Q_j log(Q_j))$ for all j's. Finally, $P_{n-1}$ computes $e(Entropy(S_v))$.

We present the formal protocol as follows:

**Protocol 1** .

1. *Step I: To compute* $e(Q_j)$.
   - *(a)* $P_n$ *sends* $e(A_{n1})$ *to* $P_1$.
   - *(b)* $P_1$ *computes* $e(A_{n1})^{A_{11}} = e(A_{n1}A_{11})$, *then sends it to* $P_2$.
   - *(c)* $P_2$ *computes* $e(A_{n1}A_{11})^{A_{21}} = e(A_{n1}A_{11}A_{21})$, *then sends it to* $P_3$.
   - *(d) Continue until* $P_{n-1}$ *obtains* $e(A_{11}A_{21}\cdots A_{n1})$.
   - *(e) Repeat all the above steps for* $A_{1i}$, $A_{2i}$, $\cdots$, *and* $A_{ni}$ *until* $P_{n-1}$ *gets* $e(A_{1i}A_{2i}\cdots A_{ni})$ *for all* $i \in [1, N]$.
   - *(f)* $P_{n-1}$ *computes* $e(A_{11}A_{21}\cdots A_{n1}) \times e(A_{12}A_{22}\cdots A_{n2}) \times \cdots \times e(A_{1N}A_{2N}\cdots A_{nN}) = e(A_{11}A_{21}\cdots A_{n1} + A_{12}A_{22}\cdots A_{n2} + \cdots + A_{1N}A_{2N}\cdots A_{nN})$.
   - *(g)* $P_{n-1}$ *computes* $e(A_{11}A_{21}\cdots A_{n1} + A_{12}A_{22}\cdots A_{n2} + \cdots + A_{1N}A_{2N}\cdots A_{nN})^{\frac{1}{N}} = e(Q_j)$.
2. *Step II: To compute* $e(Q_j log(Q_j))$.
   - *(a)* $P_{n-1}$ *generates a set of random numbers* $R_1$, $R_2$, $\cdots$, *and* $R_t$.
   - *(b)* $P_{n-1}$ *sends the sequence of* $e(Q_j)$, $e(R_1)$, $e(R_2)$, $\cdots$, $e(R_t)$ *to* $P_n$ *in a random order.*
   - *(c)* $P_n$ *decrypts each element in the sequence, and sends* $log(Q_j)$, $log(R_1)$, $log(R_2)$, $\cdots$, $log(R_t)$ *to* $P_1$ *in the same order as* $P_{n-1}$ *did.*
   - *(d)* $P_1$ *adds a random number* $R$ *to each of the elements, then sends them to* $P_{n-1}$.
   - *(e)* $P_{n-1}$ *obtains* $log(Q_j) + R$ *and computes* $e(Q_j)^{(log(Q_j)+R)} = e(Q_j log(Q_j) + RQ_j)$.
   - *(f)* $P_{n-1}$ *sends* $e(Q_j)$ *to* $P_1$.
   - *(g)* $P_1$ *computes* $e(Q_j)^{-R} = e(-RQ_j)$ *and sends it to* $P_{n-1}$.
   - *(h)* $P_{n-1}$ *computes* $e(Q_j log(Q_j) + RQ_j) \times e(-RQ_j) = e(Q_j log(Q_j))$.
3. *Step III: To compute* $e(\sum_j Q_j log(Q_j))$.
   - *(a) Repeat Step I and Step II to compute* $e(Q_j log(Q_j))$ *for all j's.*
   - *(b)* $P_{n-1}$ *computes* $e(Entropy(S_v)) = \prod_j e(Q_j log(Q_j)) = e(\sum_j Q_j log(Q_j))$.

**The Correctness Analysis of Protocol 1:** In step I, $P_{n-1}$ obtains $e(Q_j)$. In step II, $P_{n-1}$ gets $e(Q_j log(Q_j))$. These two protocols are repeatedly used until $P_{n-1}$ obtains $e(Q_j log(Q_j))$ for all $j$'s. In step III, $P_{n-1}$ computes the entropy by all the terms previously obtained. Notice that although we use $Entropy(S_v)$ to illustrate the protocol, $Entropy(S)$ can be computed following the above protocols with different input attributes.

**The Complexity Analysis of Protocol 1:**

To calculate the computation cost for each component protocol, we utilize the total number of primary operations

such as addition, subtraction, multiplication, modulo, etc [1]. The generation of a cryptographic key pair [10] is constant. We use $g_1$ to denote it. The encryption involves 6 primary operations. The decryption involves 13 primary operations. A permutation of N numbers needs $g_2 N$ operations where $g_2$ is a constant. To sort n numbers, the computation cost is denoted by $g_3 nlog(n)$. We denote the cost for generating N random numbers as $g_4 N$.

The bit-wise communication cost has the upper bound of $2\alpha(n+t+2)nc$ consisting of (1) the cost of $\alpha(n-1)$ from step I; (2) the cost of $\alpha(t+1)+2\beta(t+1)+2\alpha$ where $\beta$ denotes the number of bits for each plaintext and is assumed that $\beta < \alpha$; (3) $(nc-1)$ times of the total cost of step I and step II since the step I and step II are repeated $(nc-1)$ times.

The following contributes to the computational cost: (1) The generation of a cryptographic key pair. (2)The generation of t random numbers. (3)The total number of $nN-N+3$ exponentiations. (4)The total number of $n+nc$ multiplications where $nc$ is the total number of classes. (5)One summation.

Therefore, the total computation cost is $g_1+g_4 t+nN-N+3+n+nc+1=(n-1)N+n+g_4 t+nc+g_1+4$.

**Theorem 2** *Protocol 1 preserve data privacy at a level equal to $ADV_{P_n}$.*

For the purpose of proof, let us introduce the following notations.

- In the component protocol involving multiple parties, we use $ADV_{P_i}$ to denote the $P_i$'s advantage to gain access to the private data of any other party via the component protocol.
- We use $ADV_S$ to denote the advantage of one party to gain the other party's private data via the component protocol by knowing the semantic secure encryptions.
- We use $ADV_{P_i}$ to denote $P_i$'s advantage to gain access to the private data of any other party via the component protocol.
- We use $VIEW_{P_i}$ to denote the extra information that $P_i$ obtains via a privacy-oriented protocol.

**Proof 2** *We will identify the value of $\epsilon$ such that*

$$|Pr(T|CP) - Pr(T)| \leq \epsilon$$

*holds for $T = T_{P_i}$, $i \in [1,n]$, and CP = Protocol 1.*

*According to our notation,*

$$ADV_{P_n} = Pr(T_{P_i}|VIEW_{P_n}, Protocol1) - Pr(T_{P_i}|VIEW_{P_n}), \quad i \neq n,$$

*and*

$$ADV_{P_i} = Pr(T_{P_j}|VIEW_{P_i}, Protocol1) - Pr(T_{P_j}|VIEW_{P_i}), \quad i \neq n, i \neq j.$$

*The information that $P_i$ for $i \neq n$ obtains from other parties is encrypted by $e$ which is semantically secure, therefore,*

$$ADV_{P_i} = ADV_S.$$

*In order to show that privacy is preserved according to Definition 2, we need to know the value of the privacy level $\epsilon$. We set*

$$\epsilon = max(ADV_{P_n}, ADV_{P_i}) = max(ADV_{P_n}, ADV_S) = ADV_{P_n}.$$

*Then*

$$Pr(T_{P_i}|VIEW_{P_n}, Protocol1) - Pr(T_{P_i}|VIEW_{P_n}) \leq ADV_{P_n}, \quad i \neq n,$$

*and*

$$Pr(T_{P_j}|VIEW_{P_i}, Protocol1) - Pr(T_{P_j}|VIEW_{P_i}) \leq ADV_{P_n}, \quad i \neq n, i \neq j.$$

*which completes the proof* [2].

In the next section, we present how to compute $\frac{|S_v|}{S} Entropy(S_v)$.

$\frac{|S_v|}{S} Entropy(S_v)$

**Highlight of Protocol 2:** There are three steps in the protocol. In step I, $P_{n-1}$ obtains $e(|S_v|)^{\frac{1}{|S|}} = e(\frac{|S_v|}{|S|})$. In this step, $P_{n-1}$ sends $e(|S_v|)$ to the party (e.g., $P_i$) who holds the parent node, then $P_i$ computes $e(|S_v|)^{\frac{1}{|S|}} = e(\frac{|S_v|}{|S|})$, and sends it to $P_{n-1}$. In step II, $P_{n-1}$ obtains $\frac{|S_v|}{|S|} Entropy(S_v)$. First, $P_{n-1}$ sends $e(\frac{|S_v|}{|S|})$ to $P_1$ who computes $e(\frac{|S_v|}{|S|}) \times e(R') = e(\frac{|S_v|}{|S|} + R')$ where $R'$ is a random number from the real domain and only known by $P_1$, then sends $e(\frac{|S_v|}{|S|} + R')$ to

[1] We will follow this convention for the whole thesis.

[2] The information that $P_n$ obtains from $P_{n-1}$ is $e(Q_j), e(R_1), e(R_2), \cdots, e(R_t)$ in a random order.

$P_n$. $P_n$ decrypts it and sends $\frac{|S_v|}{|S|} + R'$ to $P_{n-1}$ who $P_{n-1}$ computes $e(Entropy(S_v))^{(\frac{|S_v|}{|S|}+R')} = e(\frac{|S_v|}{|S|}Entropy(S_v) + R'Entropy(S_v))$. Next, $P_{n-1}$ sends $e(Entropy(S_v))$ to $P_1$ who computes $e(Entropy(S_v))^{-R'} = e(-R'Entropy(S_v))$, and sends it to $P_{n-1}$. Then $P_{n-1}$ computes $e(\frac{|S_v|}{|S|}Entropy(S_v) + R'Entropy(S_v)) \times e(-R'Entropy(S_v)) = e(\frac{|S_v|}{|S|}Entropy(S_v))$. Finally, $P_{n-1}$ obtains $e(Gain(S, A))$.

We present the formal protocol as follows:

**Protocol 2** .

1. *Step I: To Compute $\frac{|S_v|}{|S|}$*
   (a) *$P_{n-1}$ sends $e(|S_v|)$ to the party (e.g., $P_i$) who holds the parent node.*
   (b) *$P_i$ computes $e(|S_v|)^{\frac{1}{|S|}} = e(\frac{|S_v|}{|S|})$, then sends it to $P_{n-1}$.*
2. *Step II: To Compute $\frac{|S_v|}{|S|}Entropy(S_v)$.*
   (a) *$P_{n-1}$ sends $e(\frac{|S_v|}{|S|})$ to $P_1$.*
   (b) *$P_1$ computes $e(\frac{|S_v|}{|S|}) \times e(R') = e(\frac{|S_v|}{|S|} + R')$ where $R'$ is a random number from the real domain and only known by $P_1$, then sends $e(\frac{|S_v|}{|S|} + R')$ to $P_n$.*
   (c) *$P_n$ decrypts it and sends $\frac{|S_v|}{|S|} + R'$ to $P_{n-1}$.*
   (d) *$P_{n-1}$ computes $e(Entropy(S_v))^{(\frac{|S_v|}{|S|}+R')} = e(\frac{|S_v|}{|S|}Entropy(S_v) + R'Entropy(S_v))$.*
   (e) *$P_{n-1}$ sends $e(Entropy(S_v))$ to $P_1$.*
   (f) *$P_1$ computes $e(Entropy(S_v))^{-R'} = e(-R'Entropy(S_v))$, and sends it to $P_{n-1}$.*
   (g) *$P_{n-1}$ computes $e(\frac{|S_v|}{|S|}Entropy(S_v) + R'Entropy(S_v)) \times e(-R'Entropy(S_v)) = e(\frac{|S_v|}{|S|}Entropy(S_v))$.*
3. *Step III: To Compute Information Gain for An Attribute.*
   (a) *$P_{n-1}$ computes $\prod_{v\in A} e(\frac{|S_v|}{|S|}Entropy(S_v)) = \sum_{v\in A}\frac{|S_v|}{|S|}Entropy(S_v)$.*
   (b) *He computes $e(\sum_{v\in A}\frac{|S_v|}{|S|}Entropy(S_v))^{-1} = e(-\sum_{v\in A}\frac{|S_v|}{|S|}Entropy(S_v))$.*
   (c) *He computes $e(Gain(S, A)) = e(Entropy(S)) \times e(-\sum_{v\in A}\frac{|S_v|}{|S|}Entropy(S_v))$.*

**The Correctness Analysis of Protocol 2:** In step I, $P_{n-1}$ obtains $e(\frac{|S_v|}{|S|})$. In step II, $P_{n-1}$ gets $\frac{|S_v|}{|S|}Entropy(S_v)$. In step III, $P_{n-1}$ gets $e(Gain(S, A))$. The computation uses the both properties of homomorphic encryption.

**The Complexity Analysis of Protocol 2:** The bit-wise communication cost of this protocol is $7\alpha$ consisting of (1) the cost of $\alpha$ from step 1; (2) the cost of $6\alpha$ from step 2.

The following contributes to the computational cost: (1)The generation of a cryptographic key pair. (2) 4 exponentiations. (3) The total number of $3 + nc$ multiplications where $nc$ is the total number of classes.

Therefore, the total computation cost is $g_1+4+3+nc = nc + g_1 + 7$.

**Theorem 3** *Protocol 2 preserve data privacy at a level equal to $ADV_{P_n}$.*

**Proof 3** *We will identify the value of $\epsilon$ such that*

$$|Pr(T|CP) - Pr(T)| \leq \epsilon$$

*holds for $T = T_{P_i}$, $i \in [1, n]$, and CP = Protocol 2.*

*According to our notation,*

$$ADV_{P_n} = Pr(T_{P_i}|VIEW_{P_n}, Protocol2) - Pr(T_{P_i}|VIEW_{P_n}), \quad i \neq n,$$

*and*

$$ADV_{P_i} = Pr(T_{P_j}|VIEW_{P_i}, Protocol2) - Pr(T_{P_j}|VIEW_{P_i}), \quad i \neq n, i \neq j.$$

*The information that $P_i$ for $i \neq n$ obtains from other parties is encrypted by $e$ which is semantically secure, therefore,*

$$ADV_{P_i} = ADV_S.$$

*In order to show that privacy is preserved according to Definition 2, we need to know the value of the privacy level $\epsilon$. We set*

$$\epsilon = max(ADV_{P_n}, ADV_{P_i}) = max(ADV_{P_n}, ADV_S) = ADV_{P_n}.$$

*Then*

$$Pr(T_{P_i}|VIEW_{P_n}, Protocol2) - Pr(T_{P_i}|VIEW_{P_n}) \leq ADV_{P_n}, \quad i \neq n,$$

*and*

$$Pr(T_{P_j}|VIEW_{P_i}, Protocol2) - Pr(T_{P_j}|VIEW_{P_i}) \leq ADV_{P_n}, i \neq n, i \neq j,$$

*which completes the proof.*

Once we compute the information gain for each candidate attribute, we then compute the attribute with the largest information gain. Without loss of generality, Let us assume there are k information gains: $e(g_1)$, $e(g_2)$, $\cdots$, and $e(g_k)$, with each corresponding to a particular attribute. We then apply privacy preserving sorting protocol, which is presented in next section, to compute the attribute with the largest information gain.

**Problem 2** *Assume that $P_1$ has a private number $t_1$, $P_2$ has a private number $t_2$, $\cdots$, and $P_n$ has a private number $t_n$. The goal is to sort $t_i$, $i \in [1, n]$ without disclosing $t_i$ to $P_j$ where $i \neq j$.*

**Highlight of Protocol ??:** In our protocol, we randomly select a key generator, e.g., $P_n$, who generates a cryptographic key pair ($e$, $d$) of a homomorphic encryption scheme. Each party encrypts their number using $e$, then sends it to $P_{n-1}$. $P_{n-1}$ computes the encryption difference of two numbers and obtains a sequence $\varphi$ of $n2$ elements. $P_{n-1}$ randomly permutes this sequence and sends the permuted sequence to $P_n$ who decrypts each element in the permuted sequence and obtains a $+1/-1$ sequence according to the decrypted results. $P_n$ sends this $+1/-1$ sequence to $P_{n-1}$ who determines the sorting result.

We present the formal protocol as follows:

**Protocol 3** .

1. *$P_n$ generates a cryptographic key pair ($e$, $d$) of a semantically secure homomorphic encryption scheme.*
2. *$P_i$ computes $e(t_i)$, for $i = 1, 2, \cdots, n-2, n$, and sends it to $P_{n-1}$.*
3. *$P_{n-1}$ computes $e(t_i) \times e(t_j)^{-1} = e(t_i - t_j)$ for all $i, j \in [1, n], i < j$, and sends the sequence denoted by $\varphi$, which is randomly permuted, to $P_n$.*
4. *$P_n$ decrypts each element in the sequence $\varphi$. He assigns the element $+1$ if the result of decryption is not less than 0, and $-1$, otherwise. Finally, he obtains a $+1/-1$ sequence denoted by $\varphi'$.*
5. *$P_n$ sends the $+1/-1$ sequence $\varphi'$ to $P_{n-1}$.*

| | $t_1$ | $t_2$ | $t_3$ | $\cdots$ | $t_n$ |
|---|---|---|---|---|---|
| $t_1$ | +1 | +1 | -1 | $\cdots$ | -1 |
| $t_2$ | -1 | +1 | -1 | $\cdots$ | -1 |
| $t_3$ | +1 | +1 | +1 | $\cdots$ | +1 |
| $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ | $\cdots$ |
| $t_n$ | +1 | +1 | -1 | $\cdots$ | +1 |

6. *$P_{n-1}$ sorts the numbers $t_i$, $i \in [1, n]$.*

**The Correctness Analysis of Protocol 3:** $P_{n-1}$ is able to remove permutation effects from $\varphi'$ (the resultant sequence is denoted by $\varphi''$) since she has the permutation function that she used to permute $\varphi$, so that the elements in $\varphi$ and $\varphi''$ have the same order. It means that if the $q$th position in sequence $\varphi$ denotes $e(t_i - t_j)$, then the $q$th position in sequence $\varphi''$ denotes the result of $t_i - t_j$. We encode it as +1 if $t_i \geq t_j$, and as -1 otherwise. $P_{n-1}$ has two sequences: one is $\varphi$, the sequence of $e(t_i - t_j)$, for $i, j \in [1, n](i > j)$, and the other is $\varphi''$, the sequence of $+1/-1$. The two sequences have the same number of elements. $P_{n-1}$ knows whether or not $t_i$ is larger than $t_j$ by checking the corresponding value in the $\varphi''$ sequence. For example, if the first element $\varphi''$ is $-1$, $P_{n-1}$ concludes $t_i < t_j$. $P_{n-1}$ examines the two sequences and constructs the index table (Table 3.1) to sort $t_i$, $i \in [1, n]$.

In Table 3.1, +1 in entry $ij$ indicates that the value of the row (e.g., $t_i$ of the $i$th row) is not less than the value of a column (e.g., $t_j$ of the $j$th column); -1, otherwise. $P_{n-1}$ sums the index values of each row and uses this number as the weight of that row. She then sorts the sequence according the weight.

To make it clearer, Let us illustrate it by an example. Assume that: (1) there are 4 elements with $t_1 < t_4 < t_2 < t_3$; (2) the sequence $\varphi$ is $[e(t_1 - t_2), e(t_1 - t_3), e(t_1 - t_4), e(t_2 - t_3), e(t_2 - t_4), e(t_3 - t_4)]$. The sequence $\varphi''$ will be $[-1, -1, -1, -1, +1, +1]$. According to $\varphi$ and $\varphi''$, $P_{n-1}$ builds the Table 3.2. From the table, $P_{n-1}$ knows $t_3 > t_2 > t_4 > t_1$ since $t_3$ has the largest weight, $t_2$ has the second largest weight, $t_4$ has the third largest weight, $t_1$ has the smallest weight.

**The Complexity Analysis of Protocol 3:** The total communication cost is (1) The cost of $\alpha(n-1)$ from step 2. (2) The cost of $\frac{1}{2}\alpha n^2$ from step 3. (3) The cost of $\frac{1}{2}\beta n^2$ from step 4 where $\beta$ denotes the number of bits for $+1$ and $-1$. Note that normally $\beta \ll \alpha$ (4) The cost of $\frac{1}{2}\beta n^2$ from step 5. Therefore, the total communication overhead is upper bounded by $\frac{3}{2}\alpha n^2 + \alpha(n-1)$.

The following contributes to the computational cost: (1)The generation of one cryptographic key pair. (2) The total number of $n$ encryptions. (3)The total number of $n^2$ multiplications. (4) The total number of $n^2$ decryptions.

| | $t_1$ | $t_2$ | $t_3$ | $t_4$ | Weight |
|---|---|---|---|---|---|
| $t_1$ | +1 | -1 | -1 | -1 | -2 |
| $t_2$ | +1 | +1 | -1 | +1 | +2 |
| $t_3$ | +1 | +1 | +1 | +1 | +4 |
| $t_4$ | +1 | -1 | -1 | +1 | 0 |

(5)The total number of $n^2$ assignments. (6) $n^2 - n$ additions. (7)$g_3 nlog(n)$ for sorting n numbers.

Therefore, the total computation overhead is $g_1 + 6n + n^2 + 13n^2 + n^2 + n^2 - n + g_3 nlong(n) = 16n^2 + 5n + g_3 nlog(n) + g_1$.

**Theorem 4** *Protocol 3 preserves data privacy at a level equal to* $ADV_{P_n}$.

**Proof 4** *We will identify the value of* $\epsilon$ *such that*

$$|Pr(T|CP) - Pr(T)| \leq \epsilon$$

*holds for* $T = T_{P_i}$, $i \in [1, n]$, *and CP = Protocol 3.*

*According to our notation,*

$$ADV_{P_{n-1}} = Pr(T_{P_i}|View_{P_{n-1}}, Protocol3) - Pr(T_{P_i}|View_{P_{n-1}}), i \neq n-1,$$

*and*

$$ADV_{P_n} = Pr(T_{P_j}|View_{P_n}, Protocol3) - Pr(T_{P_j}|View_{P_n}), j \neq n.$$

*All the information that* $P_{n-1}$ *obtains from other parties is* $e(t_i)$ *for* $1 \leq i \leq n$, $i \neq n-1$, *and the sequence* $\varphi'$.

*Since* $e$ *is semantic secure,*

$$ADV_{P_{n-1}} = ADV_S,$$

*which is negligible.*

*In order to show that privacy is preserved according to Definition 2, we need to know the value of the privacy level* $\epsilon$. *We set*

$$\epsilon = max(ADV_{P_n}, ADV_{P_{n-1}}) = max(ADV_{P_n}, ADV_S) = ADV_{P_n}.$$

*Then*

$$Pr(T_{P_i}|View_{P_{n-1}}, Protocol3) - Pr(T_{P_i}|View_{P_{n-1}}) \leq ADV_{P_n}, i \neq n-1,$$

*and*

$$Pr(T_{P_j}|View_{P_n}, Protocol3) - Pr(T_{P_j}|View_{P_n}) \leq ADV_{P_n}, j \neq n.$$

*which completes the proof.*

## 8. Discussion

To protect actual data from being disclosed, one approach is to alter the data in a way that actual individual data values cannot be recovered, while certain computations can still be applied to the data. Due to the fact that the actual data are not provided for the mining, the privacy of data is preserved. This is the core idea of randomization-based techniques. Randomization approaches were first proposed by Agrawal and Srikant [1] to solve the privacy-preserving data mining problem. Evfimievski et. al. [5] presented a framework for mining association rules from transactions consisting of categorical items where the data has been randomized to preserve privacy of individual transactions. While it is feasible to recover association rules and preserve privacy using a straightforward *uniform* randomization, the discovered rules can unfortunately be exploited to find privacy breaches. They analyzed the nature of privacy breaches and proposed a class of randomization operators that are much more effective than uniform randomization in limiting the breaches. Du and Zhan [4] proposed a technique for building decision trees using randomized response techniques [17] which were developed in the statistics community for the purpose of protecting surveyees privacy. The randomization-based methods have the benefits of efficiency. However, the drawbacks are that post-randomization data mining results are only an approximation of pre-randomization results. There are some randomization level control parameters.

Following the idea of secure multiparty computation, people designed privacy-oriented protocols for the problem of the privacy-preserving collaborative data mining. Lindell and Pinkas [9] first introduced a secure multi-party computation technique for classification using the ID3 algorithm, over horizontally partitioned data. Specifically, they consider a scenario in which two parties owning confidential databases wish to run a data mining algorithm on the union of their databases, without revealing any unnecessary information. Du and Zhan [3] proposed a protocol for making the ID3 algorithm privacy-preserving over vertically partitioned data. Lin and Clifton [8] proposed a secure way for clustering using the EM algorithm over horizontally partitioned data. Kantarcioglu and Clifton [7] described protocols for the privacy-preserving distributed data mining of association rules on horizontally partitioned data. Vaidya and Clifton presented protocols for privacy-preserving association rule mining over vertically partitioned data [15] and provided a solution for building a decision tree without compromising data privacy [16].

Encryption is a well-known technique for preserving the confidentiality of sensitive information. Comparing with other techniques described, a strong encryption scheme can be more effective in protecting the data privacy. An encryp-

tion system normally requires that the encrypted data should be decrypted before making any operations on it. For example, if the value is hidden by a randomization-based technique, the original value will be disclosed with certain probability. If the value is encrypted using a semantic secure encryption scheme [10], the encrypted value provide no help for attacker to find the original value. One of the schemes is the homomorphic encryption which was originally proposed in [14] with the aim of allowing certain computations performed on encrypted data without preliminary decryption operations. To date, there are many such systems. Homomorphic encryption is a very powerful cryptographic tool and has been applied in several research areas such as electronic voting, on-line auction, etc. [18] is mainly based on homomorphic encryption where Wright and Yang applied homomorphic encryption to the Bayesian networks induction for the case of *two* parties. Zhan et. al. [21] proposed a cryptographic approach to tackle collaborative association rule mining among multiple parties. In this paper, we will apply homomorphic encryption [10] and digital envelope techniques [2] to the privacy-preserving data mining and use them to design privacy-oriented protocols for privacy-preserving k-nearest neighbor classification problem.

In this paper, we have proposed to use homomorphic encryption and digital envelope technique to achieve collaborative decision tree classification without sharing the private data among the collaborative parties. Specifically, we provide a solution for decision tree classification with vertical collaboration in this paper. We show that our solution preserves data privacy under our definition. We provide efficient analysis for our solution. In the future, we would like to examine other privacy-preserving collaborative data mining tasks. Some, e.g., the horizontal collaboration for the decision tree classification, are discussed in [19].

## References


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

[2] D. Chaum. Security without identification. In *Communication of the ACM, 28(10): 1030–1044, October*, 1985.

[3] W. Du and Z. Zhan. Building decision tree classifier on private data. In *IEEE Intertional Workshop on Privacy, Security, and Data Mining, Maebashi City, Japan, December 9*, 2002.

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

[5] A. Evfmievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 217-228, Edmonton, Alberta, Canada, July 23-26*, 2002.

[6] O. Goldreich. The foundations of cryptography. In *Vol. 2, Cambridge University Press*, 2004.

[7] M. Kantarcioglu and C. Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. In *The ACM SIGMOD Work- shop on Research Issues on Data Mining and Knowledge Discovery (DMKD'02), pages 24-31, Madison, WI, June*, 2002.

[8] X. Lin, C. Clifton, and M. Zhu. Privacy preserving clustering with distributed em mixture modeling. In *Knowledge and Information Systems*, 2004.

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

[10] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptography - EUROCRYPT '99, pp 223-238, Prague, Czech Republic*, 1999.

[11] J. Quinlan. Induction of decision trees. In *Machine Learning, 1:81 106*, 1986.

[12] J. Quinlan. C4.5 programs for machine learning. In *San Francisco: Morgan Kaufmann Publishers*, 1993.

[13] J. Quinlan. Learning decision tree classifiers. In *ACM Computing Surveys, 28(1)*, 1996.

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

[15] J. Vaidya and C. Clifton. Privacy preserving association rule mining in vertically partitioned data. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, July 23-26*, 2002.

[16] J. Vaidya and C. Clifton. Privacy-preserving decision trees over vertically partitioned data. In *19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security Nathan Hale Inn, University of Connecticut, Storrs, CT, U.S.A., August 7-10*, 2005.

[17] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *The American Statistical Association*, 60(309):63 – 69, March 1965.

[18] R. Wright and Z. Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2004.

[19] J. Zhan. *Privacy Preserving Collaborative Data Mining*. PhD thesis, Department of Computer Science, University of Ottawa, 2006.

[20] J. Zhan and S. Matwin. A crypto-based approach to privacy preserving collaborative data mining. In *Workshop on Privacy Aspect of Data Mining (PADM'06) in conjunction with the IEEE International Conference on Data Mining (ICDM'06), HongKong, December 1*, 2006.

[21] J. Zhan, S. Matwin, and L. Chang. Privacy-preserving collaborative association rule mining. In *19th Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Nathan Hale Inn, University of Connecticut, Storrs, CT, U.S.A., August 7-10*, 2005.