InPrivate Digging: Enabling Tree-based Distributed Data Mining with Differential Privacy

Lingchen Zhao[†‡], Lihao Ni[†], Shengshan Hu[†§], Yanjiao Chen[†], Pan Zhou[¶], Fu Xiao[‖] and Libing Wu[†]
[†]School of Cyber Science and Engineering, School of Computer Science, Wuhan University, P. R. China
[‡]Collaborative Innovation Center of Geospatial Technology, P. R. China
[§]Department of Computer Science, City University of Hong Kong, P. R. China
[¶]Huazhong University of Science and Technology, P. R. China
[‖]Nanjing University of Posts and Telecommunications, P.R. China

***Abstract*—Data mining has heralded the major breakthrough in data analysis, serving as a "super cruncher" to discover hidden information and valuable knowledge in big data systems. For many applications, the collection of big data usually involves various parties who are interested in pooling their private data sets together to jointly train machine-learning models that yield more accurate prediction results. However, data owners may not be willing to disclose their own data due to privacy concerns, making it imperative to provide privacy guarantee in collaborative data mining over distributed data sets.**

In this paper, we focus on tree-based data mining. To begin with, we design novel privacy-preserving schemes for two most common tasks: *regression* and *binary classification*, where individual data owners can perform training locally in a differentially private manner. Then, for the first time, we design and implement a privacy-preserving system for gradient boosting decision tree (GBDT), where different regression trees trained by multiple data owners can be securely aggregated into an *ensemble*. We conduct extensive experiments to evaluate the performance of our system on multiple real-world data sets. The results demonstrate that our system can provide a strong privacy protection for individual data owners while maintaining the prediction accuracy of the original trained model.

I. Introduction

During the past few years, data mining has gained an incredible popularity and influenced every aspect of our daily lives. Massive data has been generated by a large number of institutions such as governments, enterprises, hospitals and so on. Data mining, a cutting-edge technique, can automatically and intelligently assist us in discovering hidden information and valuable knowledge in the vast amounts of data.

In many applications, the collection of big data often involves multiple parties who have the incentive to pool their private data sets together to train a more precise prediction model. As shown in Fig. 1, a larger data set contributes to a more accurate model, which motivates data owners to cooperate, especially when the size of individual data sets is small. For example, multiple hospitals may share the data of patients to train a model that can be more effective in diagnosing rare diseases. Unfortunately, the private data set of each data owner may contain sensitive information that can not be disclosed. A direct exposure may even violate various privacy policies such as the HIPPA privacy rule [1]. Therefore, it is imperative to resolve the privacy issue to enable collaborative data mining over distributed data sets in real-world applications.

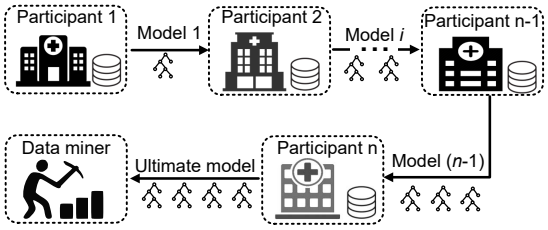


Fig. 1. A system overview of distributed data mining.

Existing works on privacy-preserving data mining have leveraged advanced cryptographic tools. A naive approach is to encrypt data locally and then release the ciphertexts, on which a handful of operations can be performed. As a typical example, many works have adopted fully homomorphic encryption [2] to secure private data in various data mining tasks such as matrix factorization [3] and linear regression analysis [4]. However, these solutions usually introduce high computation overheads, which hinders their real applications to very large data sets. Along another research line, differential privacy [5], a new privacy model, has attracted much attention owing to its efficient perturbation-based implementation. So far, extensive research efforts have been devoted to designing differentially private schemes for classic data mining tasks such as linear regression [6], support vector machines [7], decision tree [8], and neural networks [9].

Recent years have witnessed the success of tree-based data mining that can deal with non-binary labels and non-linear relationships [10], [11]. However, few state-of-art works have addressed the privacy issues raised in a distributed computing environment. Most of the existing schemes concentrate on how to enable a single data owner to securely publish her decision tree by using differential privacy [12], [13], [14]. Nevertheless, the typical application scenario where the data scatter among multiple data owners is rarely considered and investigated. Only a recent work [15] proposed a solution to construct random trees in such a distributed scenario.

Motivated by the above observations, in this paper, we address a more complicated and commonly-used tree-based data mining technique, *i.e.*, gradient boosting decision tree (GBDT) in distributed environments. GBDT is an orthogonal approach

to random trees [15] in ensemble learning. In particular, we focus on two representative data mining tasks: *regression* and *binary classification*. We design two new privacy-preserving schemes where differentially private regression trees can be learned separately by each data owner by injecting calibrated noises. We use the exponential mechanism to find the split and utilize parallel composition to allocate the privacy budget. On top of this, for the first time, we design and implement a privacy-preserving system for GBDT such that different trees trained by multiple data owners can be securely aggregated into an *ensemble* without a third party, and the data miner do not need to do any additional computations. Our experimental evaluations verify that our system can provide a strong privacy protection while achieving a high prediction accuracy comparable to the original GBDT model.

In summary, we make the following key contributions:

- We design two novel privacy-preserving schemes that enable individual data owners to learn differentially private regression trees for two representative data mining tasks, *i.e.* regression and binary classification.
- We make the first attempt to construct a privacy-preserving system for gradient boosting decision tree, which can securely aggregate distributed regression trees from different data owners into an *ensemble*.
- We extensively evaluate our system on multiple real-world data sets, confirming that our system can achieve the prediction accuracy of the original GBDT with guaranteed privacy.

II. Related Work

Privacy-preserving data mining has attracted much attention over the last decades, and many techniques have been developed to enable privacy preservation of data, such as k-anonymity [16], perturbation-based schemes [17] and cryptographic tools. Differential privacy was first proposed in [5] as a new privacy-preserving paradigm that is provably secure with a rigorous mathematical framework. It has been widely used to design privacy-preserving schemes for data publishing [18] and data mining tasks such as linear regression [6], support vector machines [7], and neural networks [9].

Decision tree is a well-known learning technique for classification and regression tasks. Many variants of decision trees have been proposed for different application scenarios, such as ID3 [19], C4.5 [10], and CART [11]. The concept of *ensemble* was proposed to aggregate multiple trees to one tree, which can be realized by two representative algorithms: random forest [20] and gradient boosting tree [21]. White-box model is ideal for decision trees, but may cause private information leakage. To address this problem, several differentially private decision tree algorithms have been proposed. Blum *et al.* [22] proposed a simple algorithm based on ID3 that breaks the splitting function down into two queries for each feature to satisfy differential privacy. However, in this early work, too much redundant noise is added to the tree, affecting its prediction accuracy. Friedman *et al.* [12] improved the previous works by optimizing the queries based on the parallel composition theory. Different splitting criteria are compared and an effort has been made to split continuous features. Zhu *et al.* [23] proposed a privacy-preserving data releasing algorithm for decision trees. Rana *et al.* [13] and Fletcher *et al.* [14] proposed differentially-private random forest to aggregate trees, which can improve the generalization performance. However, all these works focus on a single data owner who wants to publish her decision tree without revealing private information. So far, there is no consideration for the distributed scenario where multiple data owners conduct collaborative data mining.

To address the privacy issues in distributed environments, Lindell *et al.* [24] proposed a solution that uses garbled circuits [25] to choose the split features in ID3, which is the simplest binary tree. Emecki *et al.* [26] implemented a multi-party ID3 algorithm by secret sharing to utilize each participant's information gain. These solutions usually incur high communication and computation costs. Moreover, they are unable to handle more complicated decision tree algorithms that have more features than just binary features. Du *et al.* [27] partitioned one database into two parts and introduced an untrusted server to construct the binary tree. Vadiya *et al.* [15] proposed to construct privacy-preserving random trees for both horizontally and vertically partitioned data sets. However, none of the approaches in existing works are applicable to the gradient boosting decision tree (GBDT), which is the main focus of this work.

III. Preliminaries

A. Differential Privacy

Differential privacy has become a standard privacy model for statistic analysis with provable privacy guarantee [5]. It ensures that the output of a query remains (almost) unchanged no matter whether a single record in the database is tampered or not. The formal definition of differential privacy is as follows.

Definition 1 (ϵ-differential Privacy)**.** Given any two data sets D and $D^{'}$ with at most one different tuple, a randomized function $\mathcal{F}$ guarantees ϵ-differential privacy if

$$\Pr[\mathcal{F}(D) \in S] \leq \exp(\epsilon) \cdot \Pr[\mathcal{F}(D^{'}) \in S], \tag{1}$$

where ϵ is a privacy budget. A smaller ϵ realizes a higher privacy level by introducing more distortions to the protected data.

Definition 2 (Sensitivity)**.** For any function $f : \mathcal{D} \rightarrow \mathcal{R}^d$, the sensitivity of f w.r.t. $\mathcal{D}$ is

$$\Delta(f) = \max_{D,D^{'} \in \mathcal{D}} ||f(D) - f(D^{'})||_1, \tag{2}$$

where D and $D^{'}$ have at most one different record.

For numerical queries, the Laplace Mechanism is a classic method to achieve ϵ-differential privacy [5]. The main idea is to add random noises drawn from Laplace distribution to the output.

Theorem 3 (Laplace Mechanism)**.** *For any function $f : \mathcal{D} \to \mathcal{R}^d$, the Laplace Mechanism $\mathcal{M}$ for any data set $D \in \mathcal{D}$*

$$\mathcal{M}(D) = f(D) + \mathrm{Lap}(\Delta(f)/\epsilon), \tag{3}$$

where the noise $\mathrm{Lap}(\Delta(f)/\epsilon)$ is drawn from a Laplace distribution with mean zero and scale $\Delta(f)/\epsilon$, provides ϵ-differential privacy.

For non-numerical queries, the exponential mechanism is often adopted to randomize an output r in the output domain $\mathcal{R}$. Given a utility function $u(D, r)$ that calculates a score for each output r, the exponential function assigns an exponentially larger probability of being selected to r that has a higher score, such that the final output is approximately optimum with respect to u.

Theorem 4 (Exponential Mechanism)**.** *Let Δu be the sensitivity of the utility function u: $(\mathcal{D} \times \mathcal{R}) \to \mathbb{R}$, the Exponential Mechanism $\mathcal{M}$ for any data set $D \in \mathcal{D}$,*

$$\mathcal{M}(D, u) = \textit{choose } r \in \mathcal{R} \textit{ with probability} \propto \exp(\frac{\epsilon u(D, r)}{2\Delta u}) \tag{4}$$

gives ϵ-differential privacy.

For a complex mechanism with multiple queries, two privacy budget composition theorems are widely used [28].

Definition 5 (Sequential Composition)**.** If a series of privacy queries $\mathcal{Q} = \{\mathcal{Q}_1, \ldots, \mathcal{Q}_m\}$, in which $\mathcal{Q}_i$ provides ϵ_i-differential privacy, are performed sequentially on a data set, $\mathcal{Q}$ will provide $\sum_i \epsilon_i$-differential privacy.

Definition 6 (Parallel Composition)**.** If a series of privacy queries $\mathcal{Q} = \{\mathcal{Q}_1, \ldots, \mathcal{Q}_m\}$, in which $\mathcal{Q}_i$ provides ϵ_i-differential privacy, are performed separately on disjointed subsets of the entire data set, $\mathcal{Q}$ will provide $\{max(\epsilon_1, \ldots, \epsilon_m)\}$-differential privacy.

B. Gradient Boosting Decision Tree

Ensemble learning is to process a learning task by training and aggregating multiple *weak learners*. Based on the aggregation method of weak learners, ensemble learning can be classified into two categories. One assumes that the weak learners are independent and do not interact with each other, *e.g.*, random trees [15]; the other assumes that the weak learners have strong dependencies on each other, *e.g.*, gradient boosting decision tree (GBDT) [21]. GBDT consists of two main steps: training a regression tree (*i.e.*, weak learners) on each data set locally and then boosting them into a set of trees in a specific order.

The regression tree of GBDT is constructed in a top-down manner like most of the other decision tree algorithms. Its goal is to find an estimation function $f^*(x)$ that maps vector x to label y to minimize the expected value of a certain loss function $L(y, f(X))$. Starting from the root node, each internal node (non-leaf node) finds the best split to partition the set of records into two parts, each of which will be further partitioned in the child node. Splitting aims at dividing records of different classes into different parts while keeping records from the same class in the same part. In other words, the goal of splitting is to get a high purity gain that can be numerically measured, *e.g.*, by square error [21]. Assume there are n records $(x_i, y_i), i \in [1, n]$ in the training set, and $\overline{y}_i$ is the residual value, *i.e.*, the difference between the estimated value and the real value of y_i. In the root node, the residual is initiated to y_i. The square error E can be computed as $E = \sum_{i=1}^{n_c}(\overline{y}_i - \mathrm{mean}(\overline{y}_i))^2$, where n_c is the number of records in the current node, and $\mathrm{mean}(\overline{y}_i)$ is the mean residual of all n_c records. Let F_p denote a feature and $V_{p,q}$ denote a feasible value of F_p. All the records in the current node can be partitioned into two parts such that the records that satisfy $F_p < v_{p,q}$ go to the left subtree, and the others go to the right subtree. The gain from splitting the current node can be defined as $\mathrm{gain} = E - E_{\mathcal{L}} - E_{\mathcal{R}}$, in which $E_{\mathcal{L}}$ and $E_{\mathcal{R}}$ are the square errors of residuals of the two subtrees. For any feature, the best split is the one that maximizes the gain, and can be computed by finding the optimal value $v^*_{p,q}$ from the set of feasible values.

The essence of boosting is to approximate the residual $\overline{y}_i$ by the negative gradient of the loss function, *i.e.*, $-[\frac{\partial L(y, f(x_i))}{\partial f(x_i)}]_{f(x)=f_{m-1}(x)}$. We can regard the training of one of the ordered weak learners as an iteration. In each iteration, we first estimate the residual by the above loss function, then use the approximated residual to train the regression tree. For each partition of a leaf node, we calculate an optimal coefficient, which represents the step and the steepest-descent step direction. The final result of boosting is an accumulation of the updated predicted values in an apposite direction and the steps in each iteration.

IV. Problem Statement

In this paper, we consider the problem of privacy-preserving collaborative data mining using gradient boosting decision tree (GBDT) over distributed data. Fig. 1 presents an overview of our system, where multiple data owners agree on sharing data sets to extract common knowledge, but do not want to reveal any private information about their own data.

A. System Model

We assume that there are N participants (*i.e.*, data owners), and each of them has a sensitive data set for local training. A data miner submits a request to train a model using m features in all data sets, then all participants execute the learning algorithm cooperatively by various queries without data exchange and finally return the model $\mathcal{M}$ to the data miner. Note that the data miner can be one of the participants, or an external third party.

B. Threat Model

Without loss of generality, we assume that the communication security in the whole process is guaranteed by using authenticated secure channels (which is orthogonal to this work), and all participants are semi-honest, which means all participants follow the protocol but may try to infer more

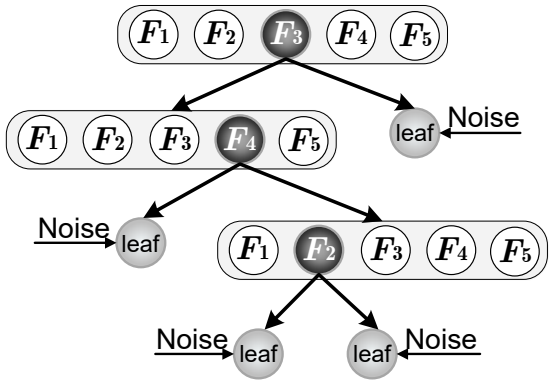


Fig. 2. A differentially private regression tree.

information. We consider participant collusion as a potential risk in our system, where a group of participants may collude to infer information of other participants. Furthermore, the data miner may also try to obtain additional statistic information from the results.

Given the above security requirements, our goal is to design a robust distributed computing system to jointly learn an accurate model (*i.e.*, GBDT) with multiple participating data owners, while preserving privacy of each participant's data against semi-honest and (potentially) collusive participants.

V. Privacy-Preserving Regression Tree

In this section, we introduce the details of our proposed privacy-preserving schemes for two classical data mining tasks: *binary classification* and *regression*. The key idea is to leverage differential privacy to provide rigorous privacy guarantee by using exponential mechanism to find the split and parallel composition to allocate the privacy budget.

A. Differentially Private Regression Tree in Binary Classification

Binary classification, one of the most common prediction tasks, outputs a binary discrete value. Suppose that the training set D has n tuples $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. For each tuple, x_i contains d features $(x_{i1}, x_{i2}, \ldots, x_{id})$. Without loss of generality, we assume that y_i lies in the scope of $[0, 1]$. Following the original work of GBDT [15], we choose the negative binomial log-likelihood as the loss function:

$$L(y, f(x)) = \log(1 + \exp(-2yf(x))),\ y \in [-1, 1], \quad (5)$$

where the prediction function $f(x)$ is a mapping from x to y. Unlike a single decision tree, a tree in gradient boosting needs to evaluate the residual of each record to improve the performance of the next round of training. Therefore, we use steepest-descent method to minimize the loss function. The negative gradient of the loss function can be viewed as the descent direction [21]. In each iteration, the residual $\overline{y}_i$ can be approximated as the negative gradient of the loss function:

$$\overline{y}_i = -[\frac{\partial L(y, f(x_i))}{\partial f(x_i)}] = \frac{2y_i}{1 + \exp(2y_i f_{m-1}(x_i))}. \quad (6)$$

The residual will be used to measure the gains of split in each internal node. Note that each query will consume some privacy budget. When the limit of total budget is hit, the data miner is no longer allowed to submit any query. Therefore, it is necessary to analyze how many queries are (at least) needed to accomplish a task in order to avoid budget exhaustion.

We first consider a simple case where all features are discrete. For each internal node including the root node, it is necessary to find an optimal feature as a split to segment the current node into subtrees. In our scheme, we obtain the best split by utilizing the square error function to compute each feature's gain based on residual $\overline{y}_i$. With the exponential mechanism, we can evaluate the gains of all features simultaneously in one query, and the best split can be derived with a high probability. Furthermore, in each depth of the tree, all sets of records in different nodes are disjointed, satisfying the condition of parallel composition. The total number of queries that will consume the privacy budget in internal nodes is $(d-1)$, where d denotes the depth of the tree.

For the leaf nodes, we don't need to split the records any more. In conventional decision tree algorithms, only one counting query is needed to return the class counts in all leaf nodes simultaneously [12]. In the regression tree, however, we need to find the optimal coefficient to update the descent direction and the step in leaf nodes. We first estimate the optimal coefficient γ as:

$$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} \log(1 + \exp(-2y_i(f_{m-1}(x_i) + \gamma))), \quad (7)$$

which can be approximated by a single Newton-Raphson step as:

$$\gamma_{jm} = \frac{\sum_{x_i \in R_{jm}} \overline{y}_i}{\sum_{x_i \in R_{jm}} |\overline{y}_i|(2 - |\overline{y}_i|)}. \quad (8)$$

For the binary classification task, to obtain the optimal coefficient in Eq. (7), we need to process two queries in each region produced by the tree: (i) $\sum_{x_i \in R_{jm}} \overline{y}_i$, and (ii) $\sum_{x_i \in R_{jm}} |\overline{y}_i|(2 - |\overline{y}_i|)$. Since the regions represented by leaf nodes are disjointed, the condition of parallel composition is also satisfied, resulting in a reduction in the privacy budget consumption. In summary, the total number of queries that will consume the privacy budget is $(d-1) + 2 = d + 1$.

Based on the prediction function $f(x)$, the prediction result is a probability distribution that specifies the class that the record belongs to:

$$\begin{aligned} P_+(x) &= p = \frac{\exp(2f(x))}{1 + \exp(2f(x))} = \frac{1}{1 + \exp(-2f(x))}, \\ P_-(x) &= 1 - p = \frac{1}{1 + \exp(2f(x))}. \end{aligned} \quad (9)$$

Now, we consider a more complicated case where features are continuous values. Using a specific value as a split will reveal the private information of the record that bears this value. In the previous work [12], the domain of continuous features is partitioned into multiple subsets, and exponential mechanism is applied to choose one subset as the split. Nevertheless, this method will result in a re-allocation of privacy budget and will affect the partition. The reason is that, during the process

Algorithm 1 Generating differentially private regression tree

Input: The training set $\mathcal{D}$, the label $\mathbf{y}$, the tree's depth d, and the total privacy budget ϵ.

Output: A privacy-preserving decision tree $t(x)$ and an optimal coefficient γ.

```
Allocate privacy budget ε1 for internal nodes.
Allocate privacy budget ε2 = ε − ε1 for leaf nodes.
Execute down-sampling on continuous features.
List all feasible values for all features in set S.
for current depth i = 1 to d − 1 do
    for all feasible values v in set S do
        Split records in value v into two parts L and R.
        Calculate gain = E_S − E_L − E_R.
    end for
    Sample the split by exponential mechanism such that
    Pr[Selecting value v] ∝ exp(ε1/(2Δ) gain).
end for
Calculate optimal coefficient γ with privacy budget ε2.
Return γ and the structure of the tree.
```

of choosing the best split for a node, GBDT will conduct an exhaustive search of all possible values for each feature to minimize $(E_{\mathcal{L}} + E_{\mathcal{R}})$, and then find the feature to maximize the gain. Such a discretization step will disclose specific values of these features. Therefore, we propose a trick of down-sampling to partition the feasible domain. Assume there are n concrete values in the domain d. We sort the n values in a non-descending order, and calculate the average of every three values, *i.e.*, $\frac{a_1+a_2+a_3}{3}, \frac{a_4+a_5+a_6}{3}, \ldots, \frac{a_{n-2}+a_{n-1}+a_n}{3}$. This operation reduces the size of domain d from $|d|$ to $\frac{|d|}{3}$. Then we use $\frac{|d|}{3}$ to find the split. This method will cause a certain degree of loss in finding the split, but can reduce the risk of privacy leakage.

To fully achieve differential privacy, an important issue is how much noise should be injected. Apart from the above privacy budget ϵ, sensitivity parameter Δ is another important factor that should be carefully considered. For the internal nodes, we need to provide the square error function for the exponential mechanism. The sensitivity of square error is $\Delta = 4$ because the maximum of residual $\overline{y}_i$ is 1 and the minimum of $\overline{y}_i$ approaches to -1. The sensitivity of both queries in terminal nodes can be bounded by 1. If we divide ϵ into ϵ_1 for internal nodes and ϵ_2 for leaf nodes, the probability of selecting $v_{p,q}$ is:

$$\Pr[\text{Selecting split } v_{p,q}] \propto \exp(\frac{1}{2\Delta}\epsilon_1 u(D, \text{gain})), \tag{10}$$

and γ_{jm} will be perturbed as

$$\gamma_{jm} = \frac{\sum_{x_i \in R_{jm}} \overline{y}_i + \text{Lap}(\frac{1}{\epsilon_2/2})}{\sum_{x_i \in R_{jm}} |\overline{y}_i|(2 - |\overline{y}_i|) + \text{Lap}(\frac{1}{\epsilon_2/2})}. \tag{11}$$

In this way, the differentially private regression tree is trained locally and can be published with privacy guaranteed. Fig. 2 shows an example of our design.

B. Differentially Private Regression Tree in Regression

Regression, another classical task, is used to predict continuous values. We will give a brief analysis on how to obtain differentially private regression tree in regression analysis.

Different from the binary classification tree, we choose negative binomial log-likelihood as the loss function:

$$L(y, f(x)) = \frac{(y - f(x))^2}{2}. \tag{12}$$

The residual can be approximated as:

$$\overline{y}_i = -[\frac{\partial L(y, f(x_i))}{\partial f(x_i)}]_{f(x)=f_{m-1}(x)} = y_i - f_{m-1}(x_i). \tag{13}$$

Therefore, the optimal coefficient can be calculated as

$$\begin{aligned} \gamma_{jm} &= \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma) \\ &= \arg\min_{\gamma} \sum_{x_i \in R_{jm}} \frac{1}{2}(y_i - (f_{m-1}(x_i) + \gamma))^2 \\ &= \arg\min_{\gamma} \sum_{x_i \in R_{jm}} \frac{1}{2}(\overline{y}_i - \gamma)^2 = \text{ave}_{x_i \in R_{jm}} \overline{y}_i. \end{aligned} \tag{14}$$

In the last line of Eq. (14), since the goal is to find the best predicted value to minimize the least square function, it is reasonable to use the average of $\overline{y}_i$ to approximate γ. Then, we can use γ to estimate the predicted value.

The difference between the query for binary classification and that for regression lies in the calculation of the optimal coefficient. In Eq. (14), two queries are needed: (i) sum all the residuals $\overline{y}_i$ in this node; (ii) count the number of records in this node. The sensitivities of both queries are 1, and the privacy budget ϵ_2 for leaf nodes should be further partitioned. γ_{jm} can be perturbed as:

$$\gamma_{jm} = \frac{\sum_{x_i \in R_{jm}} \overline{y}_i + \text{Lap}(\frac{1}{\epsilon_2/2})}{\text{count}(\overline{y}_i) + \text{Lap}(\frac{1}{\epsilon_2/2})}. \tag{15}$$

The whole process of generating differentially private regression tree is summarized in Alg. 1, which can accommodate both the regression and the binary classification.

Theorem 7. *Algorithm 1 satisfies ϵ-differential privacy.*

Proof. We divide the privacy budget into two parts ϵ_1 and ϵ_2 for the internal and the leaf nodes respectively. As for the internal nodes, all the nodes in one depth only consume the privacy budget once due to the parallel composition. Thanks to the exponential mechanism, there is only one query in each node to find the split, thus the total number of queries concerned for the budget consumption is $(d-1)$. For the leaf nodes, since the records are partitioned into disjointed subsets, the parallel composition can be applied. At each leaf node, two queries are needed to evaluate the average of residuals, thus a total of two queries should be considered for budget consumption. By dividing ϵ_1 and ϵ_2 equally for $(d-1)$ depths and 2 queries, the total consumed privacy budget is $\frac{\epsilon_1}{d-1}(d-1) + \frac{\epsilon_2}{2}2 = \epsilon$. □

VI. Privacy-Preserving GBDT: Our Construction

In this section, we formally present our privacy-preserving system for gradient boosting decision tree (GBDT). We first show the process of boosting, *i.e.*, aggregating the above differentially private regression tree into an *ensemble*. Then, we discuss two important practical issues for our system. Finally, we show how to speed up the boosting process to make it more practical.

A. A Basic Approach for Boosting

We first observe that differentially private regression trees have two properties: (i) releasing the structure of the tree to other participants is convenient for both the releaser and the adopter; (ii) regarding the tree as an independent decision tree will not leak any privacy information. These two important properties enable us to aggregate distributed trees by directly publishing the tree's structure.

In GBDT, the trees generated from different participants are correlated, making the boosting more difficult. To begin with, in the local training process, since the residual has a significant influence on subsequent tree construction, all the trees should be organized in a specific order before being sent to the data miner. Our basic idea is that each participant maintains a set of trees, and trains a new tree with privacy preservation based on her own data set. Then she updates the set of trees and sends them to the next participant in a predefined order. For example, the first participant trains the first tree t_1 and sends the structure of t_1 to the second participant, who uses t_1 to compute predictions on her own data and trains the second tree t_2 based on the results, then sends the two trees t_1 and t_2 to the third participant. A prediction result can be obtained from a trained tree by feeding a record as input and retrieving the corresponding value from a leaf node. This process continues until all participants are traversed or the number of iterations reaches a predefined threshold. During the whole process, each participant only needs to transmit the set of trees for once, and the regression tree trained by each participant preserves the privacy of her own data set. Alg. 2 gives a high-level description of our design. For each participant, the process can be represented as:

$$f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J} \gamma_{jm} I(x \in R_{jm}), \tag{16}$$

where R_{jm} is the region represented by leaf nodes.

Fig. 3 gives an illustrative example of our design. Trees are transmitted through different data owners. Each data owner trains a new tree with her own data based on previous trees and tests its performance. The last data owner outputs the set of all trees to the data miner.

B. Practical Considerations for Boosting

We consider two important practical issues that need to be addressed for our proposed boosting scheme.

The first concern is the quality of the participant's data. Though we expect all participants to be equally capable to train a tree with the same effectiveness, it is obviously impractical in real world. A malicious participant may tamper with the steepest-descent step descent direction by providing a low-quality tree. Therefore, a control should be imposed on each participant. In our scheme, we adopt prediction accuracy to measure a tree's performance. Specifically, for the classification task, the accuracy is the fraction of records that are successfully predicted; for the regression task, accuracy is measured by *mean absolute error* (MAE), defined as:

$$\mathsf{MAE} = \frac{1}{n} \sum_{i=1}^{n} |y_i - f(x_i)|. \tag{17}$$

Using the prediction accuracy as the performance indicator is reasonable, because data distributions are usually different among participants, and a tree trained on one data set may not be fitful for another data set. In addition, the learning process of tree training may converge without significant improvement after several iterations. Therefore, we can keep track of the prediction accuracy to prevent over-training.

Another problem is when to terminate the iteration. We refer to one round as n iterations from the first participant to the last participant. To avoid omitting participants with high-quality data, one whole round should not be interrupted. We define *advance* as the performance improvement in one round. Assume that a threshold p is predefined to measure the *advance*. If the *advance* exceeds p, we deem that a new round is necessary to further improve the result. Then the data miner will transmit the set of trees obtained from the last participant to the first participant for a new round. The process should be terminated when the *advance* is smaller than p, which indicates that the result is stable. It should be noted that each participant should use disjointed subsets of data in different rounds to meet the condition of parallel composition, otherwise it will cause privacy leakage. However, the number of rounds needed to reach a stable result is not known beforehand, making it

Algorithm 2 Constructing privacy-preserving GBDT

1: The data miner predefines the order of participants.
2: The data miner submits a request to the set of participants $\mathcal{S}$.
3: **for** each participant $p_i \in \mathcal{S}$ **do**
4: **if** i=1 **then**
5: Construct a new differentially private regression tree.
6: Send the trees to the next participant.
7: **else**
8: Use the previous $i-1$ trees to evaluate residuals for her own data.
9: Construct a new tree with her own data set and residuals with perturbation.
10: Send the previous $i-1$ trees along with the new tree to the next participant.
11: **end if**
12: **end for**
13: The last participant returns all trees to the data miner.

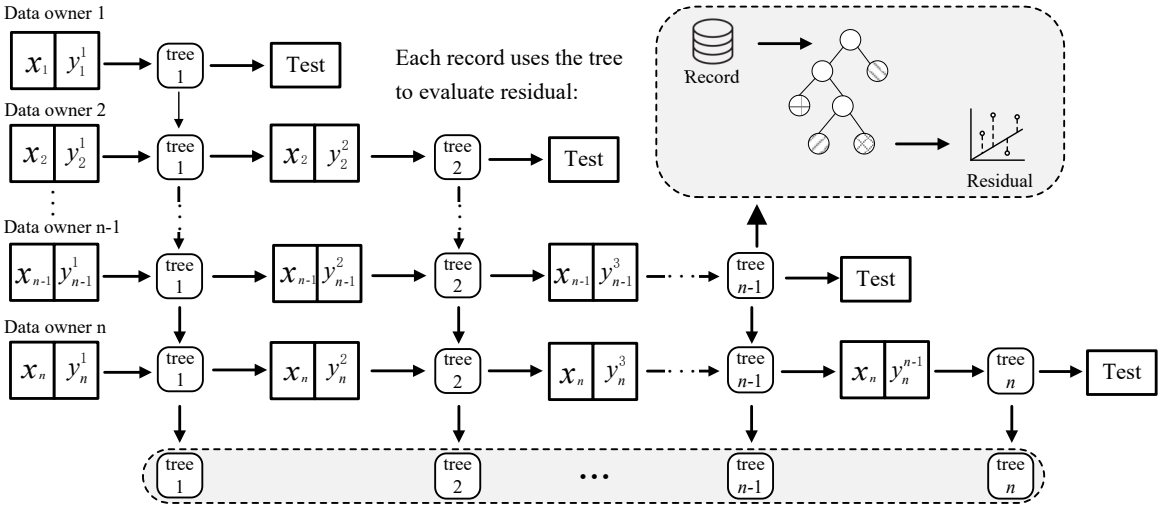


Fig. 3. Privacy-preserving GBDT in Distributed Environment.

difficult for each participant to determine how to divide their data set into subsets for multiple rounds of training. To address this issue, in each round, we utilize half of the current data set and then remove it from the current data set. For example, in the first round, for each participant, we use and remove half of the data set, and in the second round, we further bisect the remaining data set. In this way, we can achieve a faster convergence.

Privacy analysis. Theorem 7 has shown that each tree satisfies ϵ-differential privacy, which means that a direct release of the tree will not reveal private information about the corresponding data set, since no sensitive information can be inferred from the structure of the tree. Furthermore, each tree only depends on one participant's data set, and different trees use disjointed data sets. It should be noted that, different from queries, using the trained tree for calculating residuals is a mapping operation that is data-independent and will not consume the privacy budget. Since each tree satisfies ϵ-differential privacy, the final trained model of all trees satisfies ϵ-differential privacy. In addition, collusion between two or more participants cannot compromise the privacy, because they can only gain knowledge of their own data sets but cannot obtain the information of other participants.

C. Speeding up Boosting

From the above boosting process, we can see that the i-th participant has to wait until all previous $i-1$ participants have completed their training tasks. Without loss of generality, we assume that all participants have the same computing power and each participate only trains one tree on her data. For each participant, let c_1 denote the time of calculating the residual using one previous tree and c_2 denote the time of training the new tree. The total time to complete the boosting for n participants is:

$$\begin{aligned} c(n) &= (0c_1 + c_2) + \cdots + ((n-1)c_1 + c_2) \\ &= \frac{n^2 - n}{2}c_1 + nc_2. \end{aligned} \tag{18}$$

Thus, the time that the n-th participant has to wait for is

$$c(n-1) = \frac{n^2 - 3n + 2}{2}c_1 + (n-1)c_2.$$

With the growth of the number of participants n, this waiting time will become quite long, affecting the performance and the scalability of our system. Therefore, we propose a speed-up mechanism for the boosting process to reduce the waiting time. Our key idea is to achieve a certain degree of parallel computation at the expense of a little additional communication cost.

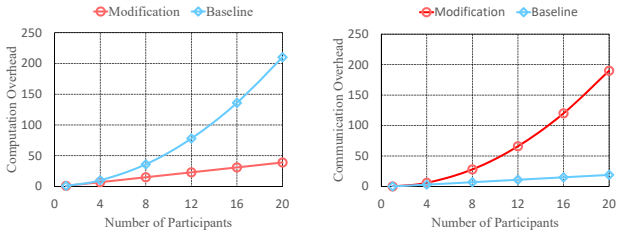
We can see that the total time cost of the i-th participant can be partitioned into three parts: (1) waiting for the previous $i-1$ participants; (2) using the $i-1$ trees to calculate residuals; (3) training a new tree with her own data. The waiting time for the $(i-1)$-th participant to train the $(i-1)$-th tree is indispensable. However, there is no need to wait for the previous $1 \sim i-2$ trees. It is possible for all participants to calculate residuals using the previous $i-2$ trees simultaneously. In order to achieve this goal, after the i-th participant has trained a new tree, she should send the tree to all the remaining $n-i$ participants rather than just the $(i+1)$-th participant. Once the remaining $n-i$ participants receive a tree, they can use the tree to evaluate residuals immediately. As a result, The total time to complete the boosting for n participants becomes:

$$\begin{aligned} c(n) &= (0c_1 + c_2) + (1c_1 + c_2) + \cdots + (c_1 + c_2) \\ &= (n-1)c_1 + nc_2. \end{aligned} \tag{19}$$

The time that the n-th participant has to wait is reduced to

$$c(n-1) = (n-2)c_1 + (n-1)c_2.$$

Note that the communication cost will increase. The number of interactions between all participants will increase from $n-1$ to $\frac{n(n-1)}{2}$. The time cost of sending trees to multiple participants, however, is negligible. This is because each participant still spends most of her time waiting for the new tree trained by the previous participant, which is far more than the time cost of sending trees. Moreover, sending trees



(a) Computation (b) Communication

Fig. 4. Comparison between baseline and modification

to all remaining participants incurs low communication costs (less than several megabits in general) and can be executed in parallel. It is worthwhile to achieve a higher efficiency with a slightly increased communication cost. Fig. 4 shows the tradeoff between computation and communication costs. In Fig. 4, *Baseline* represents the basic scheme in Sec. VI-A, and *Modification* represents the improved approach in Sec. VI-C.

VII. Experiments

In this section, we evaluate the performance of our tree-based distributed data mining system on three real-world data sets for regression and classification analysis. We implement our system on a machine with Intel Core i5-4460S CPU 2.9GHz and 12GB RAM running Windows 10, Python 3.5.

A. Data sets

We conduct experiments on 3 real-world data sets. The first one is General Social Survey (GSS) that contains responses related to marital happiness [29]. It contains 51,020 rows that correspond to individuals with 11 attributes. The classification task is to infer each interviewee's response to the question "Have you watched X-rated movies in the last year?".

The second data set is US, collected by the Integrated Public Use Microdata Series [30], which contains 600,000 census records from the Unites States. There are 15 attributes in the data set. The logistic regression task is to predict the Annual Income of an individual using the rest of the attributes. The two-class classification task is to classify whether or not the individual's annual income is greater than $50,000.

The third data set contains sale price information of houses sold between May 2014 and May 2015 in the King County (HSKC) [31]. There are 19 house features plus the price and the id, along with 21,613 observations. We remove the id and the date as they may not help with the classification task. We also remove some other attributes, which have mostly empty or discontinuous values, to reduce computation cost. The regression task of this data set is to infer house sale prices.

We use *mean absolute error* (MAE) to measure the utility of the regression analysis, and the fraction of successfully predicted samples (accuracy) to measure the performance of binary classification. For each data set, we normalize the values of labels to the interval [-1,1]. In each experiment, we restrict that the maximum depth of the gradient trees is 10, and perform 5-fold cross-validation for 10 times and report the average result.

B. Results

1) Number of Participants: We divide the three data sets into 20 subsets, each of which has 1,000 records. We vary the number of participants for training, and use the test set to evaluate the final model. We set $\epsilon = 1$. Fig. 5(a) shows the accuracy of the binary classification with the data sets GSS and US in different number of iterations. Fig. 5(b) shows the MAE on the data sets US and HSKC. In both figures, *No-Priv* means that there is no differential privacy noise. It is obvious that the accuracy increases if the number of involved participants increases. If there are more participants, the prediction accuracy will be higher. In addition, the results show that differential privacy has only a slight impact on the accuracy.

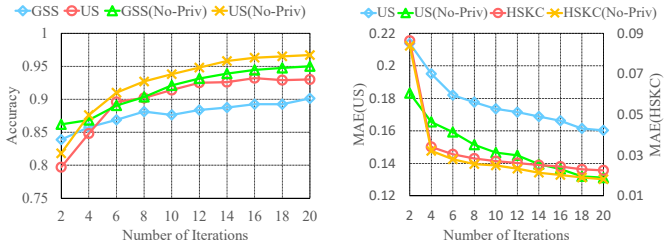
2) Size of Participants' Data: We divide the three data sets into multiple subsets with different sizes, *i.e.* 100, 500, 1,000, 1,500, or 2,000 records. There are 10 subsets for each size, so we can run ten iterations. We set $\epsilon = 1$. As shown in Fig. 6, given the same number of participants, if the size of each participant's data is larger, the final results in binary classification and logistic regression will be more accurate.

3) The Selection of Rounds: As mentioned above, the number of participants and the size of participants' data have a significant impact on the performance of our scheme. We randomly select 10,000 records from GSS and from US, then divide them into 5 subsets of 2,000 records. We set $\epsilon = 1$. We can observe that an increase in the number of rounds can improve the performance, as shown in Fig. 7. Therefore, a data miner who wants to work with a certain number of participants should conduct more rounds, which will lead to more iterations, but also a higher time cost.

4) Privacy Budgets: Sensitive data should have a smaller privacy budget, which means more noise should be added. However, the accuracy of the trained model will be reduced if more noise is added to the data. We analyze the impact of the privacy budget on the performance of our proposed scheme. Figure. 8(a) shows the accuracy of the binary classification on data sets GSS and US with different budgets, and Fig. 8(b) shows the MAE on the data sets US and HSKC. As expected, the performance deteriorates as the privacy budget becomes smaller.

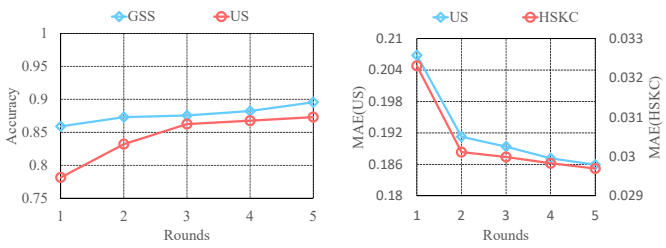
VIII. Conclusion

In this paper, we make the first attempt to design a privacy-preserving system for distributed collaborative data mining among multiple data owners without a third party. Focusing on the gradient boosting decision tree, we carefully analyze the tree construction process, and tailor the injected noise to realize ϵ-differential privacy for all data owners. Then, we propose a novel approach to securely aggregate the trees constructed by different participants to realize the distributed data mining system. The experimental results verify that our system can achieve highly accurate predictions while providing rigorous privacy guarantee. We believe that the design rationale and the solutions developed in this paper will motivate more research endeavors on privacy-preserving distributed data mining.



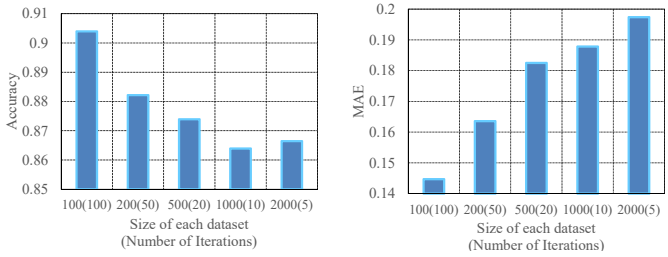
(a) Classification (b) Regression

Fig. 5. Number of iterations



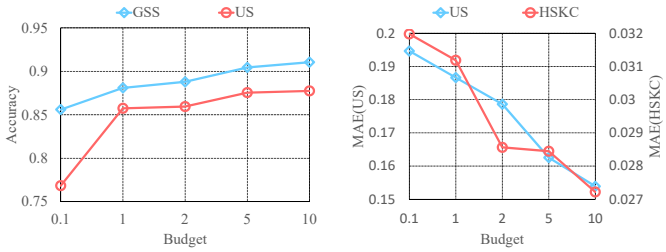
(a) Classification (b) Regression

Fig. 6. Size of participants' data



(a) Classification (b) Regression

Fig. 7. The selection of the participants



(a) Classification (b) Regression

Fig. 8. Differential privacy budgets

ACKNOWLEDGMENT

This work was supported in part by National Natural Science Foundation of China (Grant Nos. U1636219, 61373167, 61702380, U1636101), the Key Program of Natural Science Foundation of Hubei Province (Grant Nos. 2017CFA047, 2017CFA007, 2017CFB134, 2017AAA125) and Jiangsu Province (Grant No. BK20170039). Libing Wu is the corresponding author.

REFERENCES

[1] C. for Disease Control, Prevention *et al.*, "Hipaa privacy rule and public health. guidance from cdc and the us department of health and human services," *MMWR: Morbidity and mortality weekly report*, vol. 52, no. Suppl. 1, pp. 1–17, 2003.

[2] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *STOC'09*. Citeseer.

[3] S. Kim, J. Kim, D. Koo, Y. Kim, H. Yoon, and J. Shin, "Efficient privacy-preserving matrix factorization via fully homomorphic encryption," in *AsiaCCS'16*. ACM, 2016, pp. 617–628.

[4] W. jie Lu, S. Kawasaki, and J. Sakuma, "Using fully homomorphic encryption for statistical analysis of categorical, ordinal and numerical data," p. Accepted to appear, 2017.

[5] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC'06*. Springer, 2006, pp. 265–284.

[6] Z. Ji, X. Jiang, S. Wang, L. Xiong, and L. Ohno-Machado, "Differentially private distributed logistic regression using private and public data," *BMC medical genomics*, vol. 7, no. 1, p. S14, 2014.

[7] H. Li, L. Xiong, L. Ohno-Machado, and X. Jiang, "Privacy preserving rbf kernel support vector machine," *BioMed research international*, vol. 2014, 2014.

[8] M. Bojarski, A. Choromanska, K. Choromanski, and Y. LeCun, "Differentially-and non-differentially-private random decision trees," *arXiv preprint arXiv:1410.6973*, 2014.

[9] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *CCS'15*. ACM.

[10] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[11] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification and regression trees*. CRC press, 1984.

[12] A. Friedman and A. Schuster, "Data mining with differential privacy," in *SIGKDD'10*. ACM.

[13] S. Rana, S. K. Gupta, and S. Venkatesh, "Differentially private random forest with high utility," in *ICDM'15*. IEEE, 2015, pp. 955–960.

[14] S. Fletcher and M. Z. Islam, "Differentially private random decision forests using smooth sensitivity," *Expert Systems with Applications*, vol. 78, pp. 16–31, 2017.

[15] J. Vaidya, B. Shafiq, W. Fan, D. Mehmood, and D. Lorenzi, "A random decision tree framework for privacy-preserving data mining," *IEEE Transactions on Dependable and Secure Computing*, vol. 11, no. 5, pp. 399–411, 2014.

[16] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[17] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 439–450.

[18] Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren, "Real-time and spatio-temporal crowd-sourced social network data publishing with differential privacy," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, pp. 1–1, DOI: 10.1109/TDSC.2016.2 599 873, 2016.

[19] J. R. Quinlan, "Induction of decision trees," *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[20] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[21] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[22] A. Blum, C. Dwork, F. McSherry, and K. Nissim, "Practical privacy: the sulq framework," in *PODS'05*. ACM, 2005, pp. 128–138.

[23] T. Zhu, P. Xiong, Y. Xiang, and W. Zhou, "An effective deferentially private data releasing algorithm for decision tree," in *TrustCom'13*. IEEE, 2013, pp. 388–395.

[24] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *CRYPTO'00*. Springer, 2000, pp. 36–54.

[25] Y. Huang, D. Evans, J. Katz, and L. Malka, "Faster secure two-party computation using garbled circuits," in *USENIX Security Symposium*, vol. 201, no. 1, 2011.

[26] F. Emekçi, O. D. Sahin, D. Agrawal, and A. El Abbadi, "Privacy preserving decision tree learning over multiple parties," *Data & Knowledge Engineering*, vol. 63, no. 2, pp. 348–361, 2007.

[27] W. Du and Z. Zhan, "Building decision tree classifier on private data," in *CRPIT'02*. Australian Computer Society, Inc., 2002, pp. 1–8.

[28] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *FOCS'07*. IEEE.

[29] J. Prince, *Social science research on pornography*, http://byuresearch.org/ssrp/downloads/GSShappiness.pdf.

[30] *Minnesota Population Center. Integrated public use microdata series-international: Version 5.0. 2009*, https://international.ipums.org.

[31] harlfoxem, *House Sales in King County*, https://www.kaggle.com/harlfoxem/housesalesprediction.