Electronic Research Archive

http://www.aimspress.com/journal/ERA

ERA, 32(4): 2865–2879.
DOI: 10.3934/era.2024130
Received: 31 December 2023
Revised: 13 March 2024
Accepted: 26 March 2024
Published: 12 April 2024

Research article

A differentially private distributed collaborative XGBoost method

Xiongfei Li, Shuyu Li*, Hao Xu and Yixuan Zhang

School of Computer Science, Shaanxi Normal University, Xi'an 710119, China

* **Correspondence:** Email: lishuyu@snnu.edu.cn; Tel: +862985310161.

Abstract: With the rapid progress of artificial intelligence (AI) technology in medical scenarios, it becomes a trend for medical services to adopt various AI algorithms for auxiliary diagnosis and health care of patients. However, medical data is often sensitive and possibly owned by multiple participants without the willingness of data sharing. To solve this problem under the vertical partition scenario of medical data, a differentially private distributed collaborative XGBoost method named DP-DCXGBoost was proposed and applied for disease classification in the paper. Initially, a reputation-based participant selection algorithm was designed, which evaluated the contribution of participants to the global model and used it for reputation calculation to select proper participants. Then, in the collaborative training phase, the proposed method utilized the local vertical dataset of each participant to calculate feature buckets and splitting gains in order to collaboratively construct a differentially private global XGBoost classification model. Finally, the experimental analysis for two real disease datasets showed that the proposed method had good classification accuracy on the basis of preserving participants' data privacy.

Keywords: disease classification; gradient boosting; collaborative training; differential privacy; reputation evaluation

1. Introduction

With the rapid development of information and communication technologies including AI, Internet of Things (IoT) [1–3], and bioinformatics, mining and analyzing valuable information from real-world medical data has been successfully applied in various healthcare scenarios, such as auxiliary diagnosis, health monitoring, disease classification and prediction [4], and so on.

For multiple healthcare participants, there is a strong need to collaboratively build accurate models for disease classification due to limited local data of each participant. However, healthcare data often contains sensitive information of users and improper sharing may discourse users' privacy. Currently, for analysis of large scale medical data [5–7], distributed ensemble learning provides a possible solution for different participants to collaborate in building high-quality learning models in condition of preserving users' privacy. Therefore, the development of privacy-preserving distributed ensemble learning algorithms is in high demand.

In distributed ensemble learning scenarios, the participants in each task are usually fixed. The initial assumption behind is that all the participants equally contribute to the training process [8] and provide the local models with same level of quality. However, this assumption does not correspond to reality. In practice, participants with low quality local models can seriously affect the accuracy of the global model. Correctly selecting participants with high quality local models is crucial for the stability of the learning scenario and the fast convergence of the global model. Reputation based participant selection schemes can be applied for solving this problem. However, many existing reputation-based participant selection schemes adopt multi-weight subjective logic models to compute reputation [9], which may introduce the unfair subjective judgment and lead to inaccurate reputation evaluation.

Based on the above analysis, our motivation of this paper is to design a trustworthy reputation evaluation scheme for selecting proper participants and then build a privacy preserving ensemble learning algorithm for disease classification.

Leveraging the XGBoost (eXtreme Gradient Boosting) algorithm, a DP-DCXGBoost method is proposed in the paper, which is under a distributed collaborative learning scenario consisting of a central server and a set of healthcare participants. The contributions of this paper can be summarized as follows:

(1) A reputation based participant selection algorithm is designed, which evaluates the contribution of a participant's final local model to the global model to alleviate the negative effects of participants with low quality local model and improve the accuracy of the global model.

(2) The central server and the selected participants adopts the vertical learning process to collaboratively train a global classification model based on XGBoost. First-order gradients and second-order gradients are interchanged. Local samples of each participant are divided into buckets and local differential privacy is applied to preserve the data privacy of buckets.

(3) Experiments were conducted on two real datasets, and the findings exhibit the efficacy of the proposed method.

The rest of the paper is structured as follows: In Section 2, related work is discussed. Section 3 provides a review of preliminary knowledge about local differential privacy (LDP) and XGBoost algorithm. The design of the DP-DCXGBoost method is presented in Section 4. In Section 5, the results of experiments conducted on two real datasets are analyzed. Section 6 concludes the paper.

2. Related work

2.1. Ensemble learning based disease classification and prediction

By analyzing large amounts of medical data, machine learning algorithms can identify potential disease patterns and risk factors, for auxiliary disease classification and prediction. This improves not only the probability of early diagnosis for better personalized treatment, but also the survival rate and the life quality of patients.

Wibawa et al. [10] used correlation based feature selection strategy to select appropriate features, and applied AdaBoost to improve the diagnosis of chronic kidney disease. KNN, Naive Bayes, and Support Vector Machine algorithms are chosen as the base classifiers in the designed ensemble learning framework. Yang et al. [11] constructed a two-stage learning model based on stacking ensemble learning, which combines several complementary base classifiers to achieve better classification performance. Mahesh et al. [12] adopted six supervised machine learning algorithms as base classifiers to construct ensemble learning models on the breast cancer dataset. They compared the final ensemble classifier with those base classifiers and observed a significant improvement of the ensemble classifier with the evaluation metrics of accuracy and recall. Kannan [13] introduced a novel ensemble model for early prediction of heart disease using random subspace and the RSS-KNN algorithm. An outlier removal mechanism is first designed to eliminated noise and outliers in the dataset. Subsequently, the extracted features recognized by RSS are input into the KNN algorithm for accurate classification of heart disease.

2.2. Differentially private ensemble learning

In recent years, researchers have applied various privacy preservation methods for privacy preserving ensemble learning, including data anonymization, data encryption, and data perturbation. Differential privacy [14] is an important method of data perturbation with strong mathematic guarantee. The basic principle of differential privacy is to add a certain amount of noise to the original data without affecting the data statistics, so that attackers cannot infer personal information from the noisy data.

Li et al. [15] proposed a privacy-preserving gradient boosting decision tree (GBDT) training algorithm, which achieved tighter sensitivity bound and efficient privacy budget allocation. Specifically, they designed gradient-based data filtering to ensure sensitivity bound and further introduced geometric leaf clipping to achieve tighter sensitivity bound. Chaudhary et al. [16] presented a differentially private ensemble learning classification model, while maintaining its utility and accuracy, which evaluated the medical health record and classified the patient’s medical disease from the given data in a way that required very little human involvement. Li et al. [17] proposed a differentially private ensemble learning algorithm for classification. The bag of little bootstrap technique and the Jaccard similarity coefficient are first adopted to generate training datasets. And then the differentially private base classifiers are constructed and selected based on some criterion functions with corresponding weights. The final result of the classification is obtained by a weighted voting scheme. Tian et al. [18] proposed a framework named FederBoost for private federated learning of GBDT. The designed framework supported both horizontally and vertically partitioned data. The whole training process of GBDT relies on the ordering of the data instead of the values. Zhao et al. [19] designed a privacy preserving GBDT schema that allows differentially private regression trees trained by different data owners, to be sequentially integrated into a GBDT model. However, both the internal nodes and the leaf nodes in the decision tree were perturbed with equal noise, which seriously degraded the performance of the model. Besides, the training order of the participants in the proposed schema was fixed and did not consider the impact of computational resources of the participants.

3. Preliminaries

3.1. Local differential privacy

Centralized differential privacy requires a trusted third-party data collector to collect user data and perform privacy processing on it. LDP, on the other hand, assumes that the server is untrustworthy and that each user can process individual data independently. That is, privacy processing is shifted from the data collector to the user side, without the intervention of a trusted third party.

Definition 1 (ε-LDP) Given a random function f, for any two input tuples $t, t' \in Dom(f)$, and any possible output t^* of function f, if:

$$\Pr\left[f(t)=t^*\right] \le e^{\varepsilon} \times \Pr\left[f(t')=t^*\right]. \tag{1}$$

Then, the random function f is said to satisfy ε-LDP, where ε is the parameter of privacy budget. It can be seen from above definition, ε-LDP controls the similarity of the output results of any two records, so as to ensure that the random function f satisfies ε-LDP.

The two combination theorems of ε-LDP are given as follows:

Theorem 1 (Serial combination theorem). Suppose there is a set of random algorithms $\{M_1, M_2, ..., M_n\}$, each of $M_i\,(1 \le i \le n)$ satisfies ε_i-LDP on the dataset D. Then the set of M_i sequence privacy mechanisms provides $\left(\sum_{i=1}^{n} \varepsilon_i\right)$-LDP.

Theorem 2 (Parallel combination theorem). Suppose dataset D can be divided into a series of independent and non-overlapped subsets $\{D_1, D_2, ..., D_n\}$ and there is a set of random algorithms $\{M_1, M_2, ..., M_n\}$. If each privacy mechanism $M_i\,(1 \le i \le n)$ satisfies ε_i-LDP on $D_i\,(1 \le i \le n)$, then the set of randomized algorithm can achieve $\max\{\varepsilon_i\}$-LDP on D.

3.2. XGBoost algorithm

XGBoost is a machine learning algorithm used for regression, classification, and ranking. It is an efficient implementation of GBDT that can operate on large-scale datasets and exhibits strong generalization capabilities. The core of the algorithm lies in using multiple weak learners to construct a strong learner by gradually optimizing the loss function. Specifically, each weak learner is a decision tree model, and XGBoost employs a custom loss function that allows simultaneous consideration of error magnitude and complexity when building each tree. Additionally, XGBoost utilizes a regularization technique, namely L1 and L2 regularization, to prevent overfitting.

At each iteration, XGBoost computes the gradient and Hessian matrix for each sample, which are used to build the decision tree. Then, based on the gradient and Hessian matrix of the loss function, it calculates the split gain for each node to determine which feature and threshold can minimize the loss function. Finally, a new decision tree is generated by greedy selecting the split points. After a certain number of iterations, XGBoost combines multiple decision trees to form a strong learner.

4. The DP-DCXGBoost method

4.1. Framework

In this paper, we consider a distributed collaborative learning scenario consisting of a central server and a set of healthcare participants. The central server is honest but curious. While each healthcare participant owns an unshared private local dataset, the dataset is vertically partitioned according to features. The goal of the DP-DCXGBoost method is twofold: (1) To allow the participants to collaboratively train the disease classification model without disclosing the privacy information of each participant; and (2) participants with more positive contributions are more likely selected for collaborative learning. The overall framework of the proposed method is shown in Figure 1.

For the first time of distributed collaborative model learning task, the central server randomly selects some of the participants to release the initialized global model parameters, and the selected participants receive the model parameters and perform the local model training based on the corresponding private data. When one round of local model training is completed, each participant uploads the local model parameters to the central server. The central server searches for the global optimal split point according to the aggregation of local model parameters uploaded by the participants. The optimal split point is then sent down to the participants for the next round of iterations, and the model training task is completed when the global model parameters have converged or the number of iterations have reached the maximum number. After the task is completed, the central server evaluates the final local model quality of each participant and adjusts the reputation of each participant. In subsequent model training tasks, the central server will prioritize the participants with higher reputation for better collaborative learning.

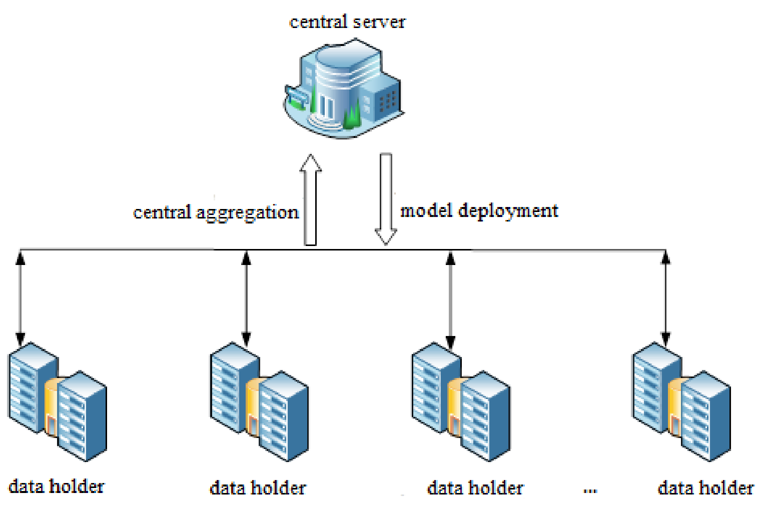


Figure 1. The framework of the DP-DCXGBoost method.

4.2. Reputation based participant selection

Aiming at the problem that the performance of the global model suffers due to the differences in model quality of each participant in distributed learning, the DP-DCXGBoost method designs a participant selection algorithm based on the reputation of participant, which is calculated according to the local model quality of the participant.

Supposing there is a central server and n participants, a collaborative training task requires $m(m<n)$ participants to take part in, and each task trains up to T decision trees to obtain the final XGBoost global model. After a task is completed, the central server assigns different reputation values to the participants based on the quality of their final local models. Before the start of the next task, the central server selects suitable participants to join the task based on their reputation. Since there is no a priori knowledge about the local model quality of the participants, the central server initializes the reputation of each participant with the same constant value.

The central server also maintains a test dataset locally, and scores participants based on the final models it receives. This scoring is done by comparing the difference of the two global models whether a certain participant's final local model is aggregated or not. The scoring function is defined as follows:

$$E_{i,k}=G_k(D_s)-G_k^{-i}(D_s)\,, \tag{2}$$

where D_s is the test dataset, $G_k(D_s)$ represents the prediction accuracy of the global model (aggregating the participant $i's$ final local model) on the test dataset in the kth task, while $G_k^{-i}(D_s)$ denotes the prediction accuracy of the global model without aggregating the participant $i's$ final local model, and $E_{i,k}$ is score value. Additionally, a quality metric is introduced to normalize the participant's score into quality value, which takes the reciprocal of $E_{i,k}$ and normalizes it to obtain the participant's model quality metric, denoted by $l_{i,k}$:

$$l_{i,k}=\frac{1/e^{l_{i,k}}}{\sum_{i=1}^{m}1/e^{l_{i,k}}}\,, \tag{3}$$

where m is the number of candidate participants. It can be noted that $l_{i,k}$ is a real number belonging to (0,1) and $\sum_{i=1}^{m}l_{i,k}=1$. After evaluating the final models of the participants, the selection of participants is based on their historical reputation. The reputation of participants gradually increases if they consistently provide high quality models. However, if a participant contributes a poor model in a single training task, their reputation will rapidly decrease. The evaluation strategy also provides incentives for each participant to offer high quality local model in each training task.

The DP-DCXGBoost method adopts the Richard's Curve to update the participant's reputation, which is defined as follows:

$$R_{i,k}(l_{i,k}^{'})=A+\frac{K-A}{(C+Qe^{-B(l_{i,k}^{'}-D)})^{\frac{1}{v}}}\,, \tag{4}$$

where A represents the lower asymptotic bound of the curve, K is the upper asymptotic bound of the curve, B is the exponential growth rate of the curve, D determines the maximum growth of the curve, Q is related to the initial value $R_{i,k}(0)$ and typically in [0.1,2], C is a constant value and typically set to 1, and v determines the asymptotic maximum growth point.

At the end of each task, after the central server has evaluated the quality of the participant's local model, the participant's reputation is updated according to formula 4, where the output $R_{i,k}(l'_{i,k}) \in (0,1)$ is the updated reputation value of the ith participant after completing the kth task. The input variable $l'_{i,k}$ of the Richard curve is calculated according to the following formula:

$$l'_{i,k} = \begin{cases} \sum_{j=1}^{k} x_{i,j}(1-l_{i,k})^{k-j}(l_{i,k}-1/n) & \text{if } l_{i,k} \le 1/n, \\ \sum_{j=1}^{k} x_{i,j}(l_{i,k})^{k-j}(l_{i,k}-1/n) & \text{if } l_{i,k} > 1/n, \end{cases} \tag{5}$$

where n is the number of participants, $x_{i,j} = 1$ indicates that participant i takes part in the jth model training task, otherwise $x_{i,j} = 0$. When evaluating the reputation of participant i in the kth task, a basic idea is to sum the indicators of the local model quality that this participant owns in the previous $k-1$ tasks. It is important to note that the increase gradient of reputation and the decrease gradient of reputation should not be measured in the same way. In particular, the decreased gradient of reputation for participants should be greater than the increased gradient of reputation.

After considering the reputation of the participants, the central server selects the participants with higher reputation for the next training task. The pseudo code of reputation based participant selection algorithm is shown in Algorithm 1.

Algorithm 1: reputation based participant selection algorithm

Input: k is the number of completed tasks, n is the number of participants, $m(m<n)$ is the number of candidate participants
Output: the set of candidate participants for the $(k+1)th$ task

```
1. the central server initializes the reputation values for all the participants as $R_{i,0} = 1/n(1 \le i \le n)$;
2. if $k > 1$ then
3.   calculate quality metric $l_{i,k}$ for participant $i$ in the $kth$ task according to formula (3);
4.   calculate the input variable $l'_{i,k}$ for participant $i$ according to formula (5);
5.   update the reputation value $R_{i,k}$ for participant $i$ according to formula (4);
6.   sort participants according to the descent order of reputation value of each participant;
7.   select top $m$ particpants;
8. else
9.   randomly select $m$ participants; // the first training task
10. return the set of candidate participants.
```

4.3. Model training and prediction

For each model training task, the central server first selects a set of candidate participants, denoted by $Par = \{P_1, P_2, ..., P_m\}$ using algorithm 1. In each task, both the central server and participants owns different feature sets $\{X_c, X_1, X_2, ..., X_m\}$. Each feature set X_i consists of K features denoted by $X_i = [x^1, x^2, ...x^K]$, and each feature x^k contains all r data samples $x^k = [x_1^k, x_2^k, ..., x_r^k]$. However, only the central server has the class labels $y = [y_1, y_2, ...y_r]$ for these r samples.

During the model training phase of the DP-DCXGBoost method, the goal is to obtain the final XGBoost ensemble model. To preserve data privacy during the vertically federated training process, both the central server and the participants follow the learning strategies:

(1) First-order gradient g_i and second-order gradient h_i for samples are calculated based on the sample class labels y_i. When seeking the global optimal splitting point in the training process, only the central server possesses the sample class label values, while participants cannot have access. If the central server was to directly transmit g_i and h_i to the participants, the participants could calculate the optimal splitting point for their local data using the information gain formula. However, if participants have knowledge of g_i and h_i, they could easily infer the class label values, thereby compromising data privacy. Therefore, the central server should send g_i and h_i in a privacy preserving way. Initially, the central server independently constructs the first decision tree, with no involvement of the participants. Then, participants can only access to the gradients g_i and h_i after residual update in the first tree. This ensures that malicious participants cannot infer the class label information.

(2) For each feature, participants sort the samples based on their feature values and place these samples into l buckets. After receiving gradient aggregation information from the participants regarding the buckets, the central server knows only the relative order of bucket sizes and has no knowledge about the order of samples within each bucket. However, it is possible for information to leak between two samples in different buckets. The DP-DCXGBoost method uses LDP to add noise for each bucket, ensuring the data privacy of participants.

The training process of the DP-DCXGBoost method is described as follows:

1. The central server initializes parameters, calculates the first-order gradient g_i and second-order gradient h_i for each sample, and uses the approximate split-finding algorithm to independently generate the first decision tree;

2. Based on the first decision tree's leaf node weights, the central server updates the values of y_i, the gradients g_i and h_i for the samples. Then, g_i and h_i are sent to m participants.

3. Each participant receives g_i and h_i, divides samples into buckets based on candidate quantile points for each feature, adds differential privacy noise to each bucket, and calculates the G_{kv} and H_{kv}, which are the sum of g_i and h_i for each bucket, respectively. These two sum values are then sent back to the central server.

4. The central server traverses all the features and thresholds, computes the feature k_{opt} and the threshold v_{opt} corresponding to the globally optimal split point, and sends them to the participants. A participant splits the current sample space based on the values of k_{opt} and v_{opt}. Due to the separation on features, a participant is required to store specific information for each split, generates a lookup table of node split information [feature k, threshold v] with a unique sample number ID as an index of the table based on k_{opt} and v_{opt}, in order to predict new samples later. A participant then sends the sample number ID and $\{G_k, H_k\}_{k=1}^{K}$ to the central server.

5. The central server splits the current node based on the received $\{G_k, H_k\}_{k=1}^{K}$ from the participants and calculates the leaf node weight. Afterward, it associates the current node with [participant ID, sample ID].

6. Repeat this process until the global model converges or the maximum number of decision trees for iteration is reached. Stop splitting and obtain the final model.

7. The central server evaluates the quality of the local models provided by each participant and updates the reputation values.

From the perspective of a participant, upon receiving g_i and h_i, a participant sorts the samples and places them into l buckets based on candidate quantile points for each feature. Then, differential privacy noise is added to each bucket. Specifically, for a sample initially assigned to the a bucket, there is a probability $p=\frac{e^{\varepsilon}}{e^{\varepsilon}+l-1}$ that the sample remains in the same bucket, and a probability $q=\frac{1}{e^{\varepsilon}+l-1}$ that it is moved to another bucket. Subsequently, they calculate the sums G_{kv}^{m} and H_{kv}^{m} of g_i and h_i within each bucket and send them to the central server. The participant's execution process is illustrated in Algorithm 2.

Algorithm 2: participant i training process

```
Input: Sample space $I$ of the current node, feature set $X_i=[x^1,x^2,...x^K]$, $\{g_i,h_i\}_{i\in I}$
Output: $\{G_k,H_k\}_{k=1}^{K}$
1. for each $x^j \in X_i$ // Iterate over each feature of the current node
2.    sort $x^j$ and calculate the candidate quantile points $S_k=\{s_{k1},s_2,...,s_{kl}\}$;
3.    divide $x^j$ into $l$ buckets based on the quantile points;
4. end for
5. for each $x^k \in x^K$
6.    for each $x_i^k \in x^k$ // iterate through each sample within the feature
7.       move the value $x_i^k$ to another bucket with a probability of $\frac{1}{e^{\varepsilon}+l-1}$;
8.    end for
9.    $G_{kv}=\sum_{i\in\{i|s_{k,v}\geq x_i^k\geq s_{k,v-1}\}} g_i$
10.   $H_{kv}=\sum_{i\in\{i|s_{k,v}\geq x_i^k\geq s_{k,v-1}\}} h_i$
11. end for
12. send $\{G_k,H_k\}_{k=1}^{K}$ to the central server.
```

From the central server's perspective, the central server independently generates a decision tree and calculates g_i and h_i for each sample. First, locally compute aggregation information G_{kv} and H_{kv} for different bins based on each feature's candidate quantile points. Then, send g_i and h_i to the participants. After receiving the gradient aggregation information from the participants, the central server calculates the local optimal split points for the participants. Next, it compares the information gain of the global optimal split point with the local optimal split points and selects the global optimal split point. The process is illustrated in Algorithm 3.

Algorithm 3 central server training process

```
Input: the sample space $I$ of the current node, the participants' aggregated gradient $\{G^j,H^j\}_{j=1}^{m}$
Output: partition the current sample space according to the optimal split point
1. $g \leftarrow \sum_{i\in I} g_i$ , $h \leftarrow \sum_{i\in I} h_i$ ;
2. for each $P_j \in Par$ // Iterate through each participant
3.    for each $x^k \in x^K$ // Iterate over each feature
4.       $g_l \leftarrow 0$ , $h_l \leftarrow 0$;
5.       for each $v \in l_k$ // Iterate over each bucket
6.          $g_l \leftarrow g_l + G_{kv}^{j}$, $h_l \leftarrow h_l + H_{kv}^{j}$
7.          $g_r \leftarrow g - g_l$ , $h_r \leftarrow h - h_l$;
```

```
8.          $Gain \leftarrow \max(Gain, \frac{g_l^2}{h_l+\lambda} + \frac{g_r^2}{h_l+\lambda} - \frac{g^2}{h+\lambda})$
9.      end for
10.   end for
11. end for
12. Obtain the best $Gain$, record the corresponding participant $P_s$, and send $k_{opt}$ and $v_{opt}$ to the participant $P_s$;
// step 13–14 are executed on participant side
13. participant $P_s$ divide the sample space based on $k_{opt}$ and $v_{opt}$;
14. participant $P_s$ make samples in the split information lookup table and send the sample ID and $I_L$ to the central server;
15.The central server splits the sample space based on $I_L$, calculates the leaf node weights, and associates the current node with [participant ID, sample ID].
```

In the prediction process, each participant has only complete information about the local feature split points, and multiple participants must work together to determine the leaf nodes of the sample in the prediction process. Initially, when a sample enters the root node of the first decision tree, the central server queries the [participant ID, sample ID] associated with the current node, and then sends a message to the corresponding participant to ask whether this sample should enter the left subtree or the right subtree. After receiving the message, the corresponding participant compares the sample value with the threshold value in the local split information lookup table, and sends the decision information (whether the sample should enter the left subtree or not) to the central server. The central server then allows the sample to enter the related subtree. The above steps are repeated until the sample reaches a leaf node. After determining the leaf node of the sample, the central server calculates the predicted value of the sample based on the weights of the leaf nodes of each tree.

Supposing d is the max depth of decision trees, T is the number of decision trees, $\|x\|_0$ is the number of non-missing items in the whole training dataset, since quantile sketch strategy is applied in the training process, the time complexity of training process is $O(dT\|x\|_0 + \|x\|_0 \log Kmr)$, where K is the number of features, m is the number of selected participants, and r is the number of samples on each participant.

4.4. Privacy analysis

In this section, we briefly prove that the DP-DCXGBoost method satisfies ε-differential privacy.

Initially, we give the definition of neighbor data of two local samples $x = [x_1, x_2, ..., x_k]$ and $x' = [x_1^{'}, x_2^{'}, ..., x_k^{i}]$, respectively. If there is only one element difference between x and x', then x and x' is called neighbor data.

In the distributed collaborative training phase, after each participant receives g_i and h_i, the samples are sorted and put into l buckets according to the candidate quantile points of each feature, and differential privacy noise is added for each bucket. For any two samples x, x' and a bucket B, it can be obtained:

$$\frac{\Pr[bucketize(x) \in B]}{\Pr[bucketize(x') \in B]} \le \frac{p}{q} = \frac{e^{\varepsilon} / (e^{\varepsilon} + q - 1)}{1/(e^{\varepsilon} + q - 1)} = e^{\varepsilon} . \tag{6}$$

Therefore, the proposed method satisfies ε-differential privacy, where $\Pr[bucketize(x) \in B]$

denotes the probability that sample x is placed in the bucket B.

5. Experiments

5.1. Experimental settings

We implemented the proposed method in Python programming language (version 3.9), and conducted experiments on a machine with Intel (R) i7-11700 CPU @ 2.50GHz, 16G RAM, 512 SSD + 1T hardware configuration and a 64-bit Windows 10 operating system.

We utilized two real datasets, Cardiovascular diseases dataset from Kaggle public database [20] and Diabetes 130-US hospitals for years 1999–2008 dataset from UCI public database [21]. Table 1 outlines the statistics of the two datasets.

Table 1. Statistics of datasets.

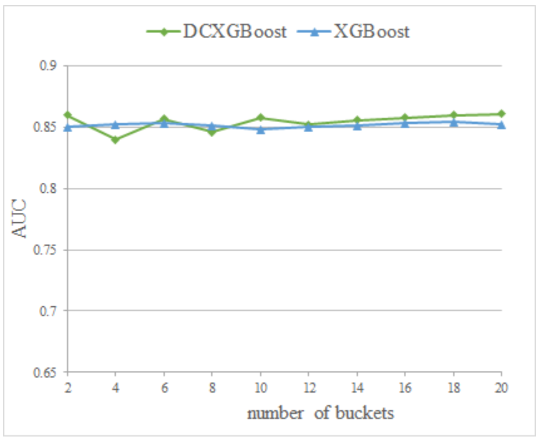
Data Set	Sample Size	Number of features
Cardiovascular disease	68783	12
Diabetes disease	100000	55

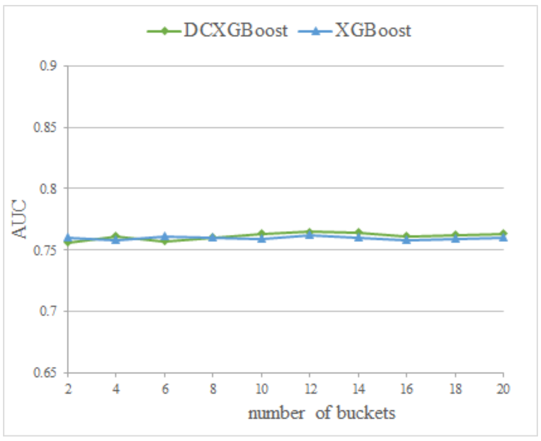
5.2. Experimental results

In order to test the performance of the proposed method, three groups of comparative experiments are carried out. For each dataset above, it is divided into two parts for training and testing. The training part contains two thirds of the samples and the testing part contains the remaining.

Figure 2(a),(b) show the area under curve (AUC) of the feature values trained using the Cardiovascular diseases dataset and Diabetes diseases dataset under different number of buckets, respectively. DCXGBoost is the non-privacy-preserving version of the proposed method, which removes the differential privacy noise. Moreover, XGBoost is run in a centralized environment using the same dataset and the result is adopted as a baseline for comparison with DCXGBoost. The results show that the change in AUC varies little as the number of buckets increases. Furthermore, DCXGBoost achieves almost the same AUC value as that of XGBoost with the same number of buckets. Vibrations can be observed with DCXGBoost in Figure 2 when the bucket number is small. One possible reason is that buckets are used for samples division and sort, the efficiency of parallel computing is not high or stable when the bucket number is small.

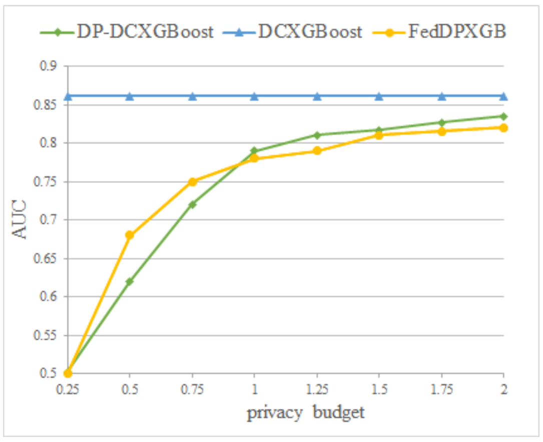
In order to verify the effect of privacy budget ε to the proposed method, the number of buckets with the best AUC is fixed. The DP-DCXGBoost method is compared with FedDPXGB (Federal Differential Privacy XGBoost) [22] under different privacy budgets, and the result of DCXGBoost is used as a base line. The experimental results on the cardiovascular disease dataset and the diabetes disease dataset are shown in Figure 3(a),(b), respectively. It can be seen from Figure 3 that with the increase of privacy budget, the AUC value of both DP-DCXGBoost and FedDPXGB display an increasing trend. This result is expected because the error generated by differential privacy decreases as privacy budget increases. Furthermore, the AUC value of both DP-DCXGBoost and FedDPXGB become stabilize when privacy budget is greater than 1.25. When the privacy budget is greater than 0.9, the performance of the proposed method achieves better than that of the FedDPXGB method.

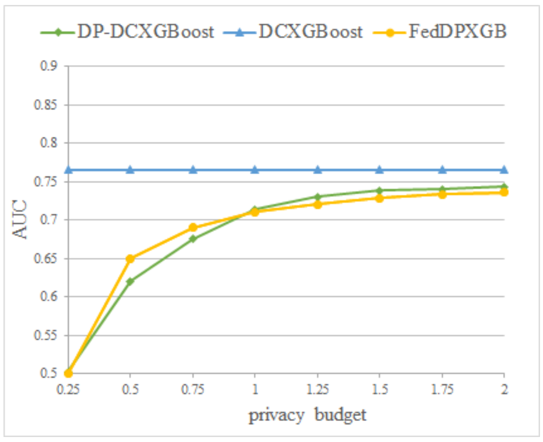




(a) AUC comparison (b)AUC comparison

Figure 2. AUC values under different number of buckets.

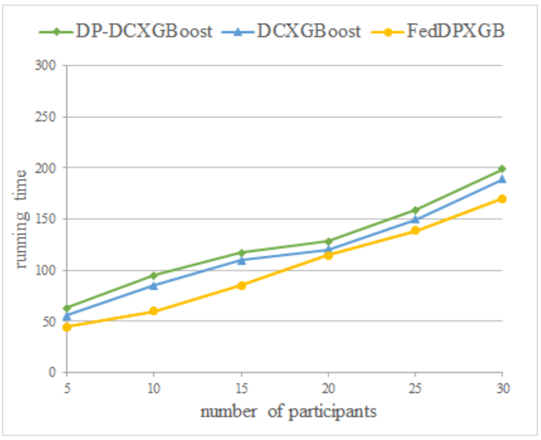


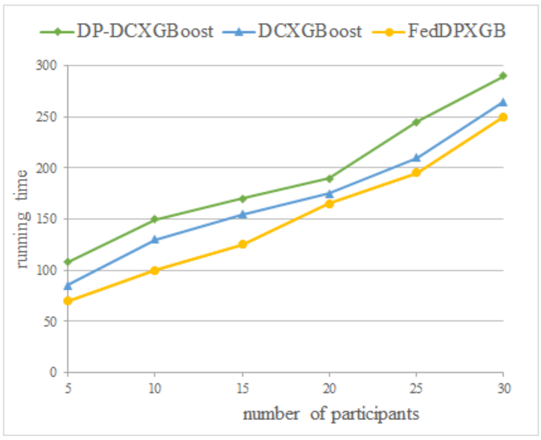


(a) AUC comparison (b)AUC comparison

Figure 3. AUC values under different privacy budgets.

The efficiency of the DP-DCXGBoost method can also be influenced by the indicator of running time. To investigate running time under different numbers of participants, the DP-DCXGBoost method is compared with DCXGBoost and FedDPXGB. The experimental results for the Cardiovascular Diseases dataset and the Diabetes Diseases dataset are shown in Figure 4(a),(b), respectively. The results show that the running time of all three methods increases as the number of participants increases. The running time of the DCXGboost method is always lower than that of the DP-DCXGboost method. Besides, the running tiem of the FedDPXGB method is better than that of DP-DCXGboost under the same conditions, which is due to the fact that the proposed method spends some time on the reputation based participant selection before model training. As a whole, the proposed method meets the design expectations.





(a) running time comparison (b) running time comparison

Figure 4. Running time under different numbers of participants.

6. Conclusions

In this paper, we propose the DP-DCXGBoost method to address the privacy preserving collaborative learning problem in the scenario where participants' medical data is vertically partitioned. To alleviate the negative effects of participants with low quality local model and improve the accuracy of the global model, a reputation based participant selection algorithm is first designed, which evaluates the contribution of a participant's final local model to the global model. Then, the central server and the selected participants, collaboratively train a global classification model based on XGBoost. Local samples of each participant are divided into buckets and local differential privacy is applied preserve the data privacy of buckets. Experimental results for two real disease datasets validate the effectiveness of the proposed method.

In the future, we plan to consider the selection of participants based on their communication conditions and computational resources and improve the efficiency of participant selection algorithm. In addition, the proposed method does not introduce noise when the gradients of the participants is uploaded to the central server. We will try to apply mixed differential privacy technology to the proposed method.

Use of AI tools declaration

The authors declare that they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This research was funded by "Key Research and Development Program of Shaanxi Province", grant number 2021GY-090.

Conflict of interest

The authors declare there is no conflict of interest.

References

1. C. Wang, C. Jiang, J. Wang, S. Shen, S. Guo, P. Zhang, Blockchain-aided network resource orchestration in intelligent internet of things, *IEEE Int. Things J.*, **10** (2022), 6151–6163. https://doi.org/10.1109/JIOT.2022.3222911
2. J. Lu, H. Liu, R. Jia, J. Wang, L. Sun, S. Wan, Towards personalized federated learning via group collaboration in IIoT, *IEEE Trans. Ind. Inform.*, **19** (2023), 8923–8932. https://doi.org/10.1109/TII.2022.3223234
3. G. Wu, L. Xie, H. Zhang, J. Wang, S. Shen, S. Yu, STSIR: An individual-group game-based model for disclosing virus spread in Social Internet of Things, *J. Netw. Comput. Appl.*, **214** (2023), 103608. https://doi.org/10.1016/j.jnca.2023.103608
4. A. Yaqoob, R. M. Aziz, N. K. Verma, P. Lalwani, A. Makrariya, P. Kumar, A review on nature-inspired algorithms for cancer disease prediction and classification, *Mathematics*, **11** (2023), 1081. https://doi.org/10.3390/math11051081
5. B. Dou, Z. Zhu, E. Merkurjev, L. Ke, L. Chen, J. Jiang, et al., Machine learning methods for small data challenges in molecular science, *Chem. Rev.*, **123** (2023), 8736–8780. https://doi.org/10.1021/acs.chemrev.3c00189
6. N. Liu, X. Li, E. Qi, M. Xu, L. Li, B. Gao, A novel ensemble learning paradigm for medical diagnosis with imbalanced data, *IEEE Access*, **8** (2020), 171263–171280. https://doi.org/10.1109/ACCESS.2020.3014362
7. Y. Yang, H. Lv, N. Chen, A survey on ensemble learning under the era of deep learning, *Artif. Intell. Rev.*, **56** (2023), 5545–5589. https://doi.org/10.1007/s10462-022-10283-5
8. T. Nishio, R. Yonetani, Client selection for federated learning with heterogeneous resources in mobile edge, in *ICC 2019-2019 IEEE international conference on communications (ICC)*, IEEE, (2019), 1–7. https://doi.org/10.1109/ICC.2019.8761315
9. J. Kang, Z. Xiong, D. Niyato, S. Xie, J. Zhang, Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory, *IEEE Int. Things J.*, **6** (2019), 10700–10714. https://doi.org/10.1109/JIOT.2019.2940820
10. M. S. Wibawa, I. M. D. Maysanjaya, I. M. A. W. Putra, Boosted classifier and features selection for enhancing chronic kidney disease diagnose, in *2017 5th International Conference on Cyber and IT Service Management*, IEEE, (2017), 1–6. https://doi.org/10.1109/CITSM.2017.8089245
11. Y. Yang, L. Wei, Y. Hu, Y. Wu, L. Hu, S. Nie, Classification of Parkinson's disease based on multi-modal features and stacking ensemble learning, *J. Neurosci. Meth.*, **350** (2021), 109019. https://doi.org/10.1016/j.jneumeth.2020.109019
12. T. R. Mahesh, V. Vinoth Kumar, V. Vivek, K. M. Karthick Raghunath, G. Sindhu Madhuri, Early predictive model for breast cancer classification using blended ensemble learning, *Int. J. Syst. Assur. Eng. Manag.*, **15** (2024), 188–197. https://doi.org/10.1007/s13198-022-01696-0
13. S. Kannan, An automated clinical decision support system for predicting cardiovascular disease using ensemble learning approach, *Concurr. Comp.—Pract. E.*, **34** (2022), e7007. https://doi.org/10.1002/cpe.7007
14. C. Dwork, Differential privacy, in *International Colloquium on Automata, Languages, and Programming*, Springer, (2006), 1–12. https://doi.org/10.1007/11787006_1

15. Q. Li, Z. Wu, Z. Wen, B. He, Privacy-preserving gradient boosting decision trees, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **34** (2020), 784–791. https://doi.org/10.1609/aaai.v34i01.5422
16. N. Chaudhary, V. Gupta, K. Sandhir, R. Gupta, S. Chhabra, A. K. Singh, Privacy preserving ensemble learning classification model for mental healthcare, in *2022 Seventh International Conference on Parallel, Distributed and Grid Computing (PDGC)*, IEEE, (2022), 513–518. https://doi.org/10.1109/PDGC56933.2022.10053268
17. X. Li, J. Liu, S. Liu, J. Wang, Differentially private ensemble learning for classification, *Neurocomputing*, **430** (2021), 34–46. https://doi.org/10.1016/j.neucom.2020.12.051
18. Z. Tian, R. Zhang, X. Hou, J. Liu, K. Ren, Federboost: Private federated learning for gbdt, preprint, arxiv:2011.02796.
19. L. Zhao, L. Ni, S. Hu, Y. Chen, P. Zhou, F. Xiao, et al., Inprivate digging: Enabling tree-based distributed data mining with differential privacy, in *IEEE INFOCOM 2018—IEEE Conference on Computer Communications*, IEEE, (2018), 2087–2095. https://doi.org/10.1109/INFOCOM.2018.8486352
20. Cardiovascular Diseases Dataset (clean). Available from: https://www.kaggle.com/datasets/aiaiaidavid/cardio-data-dv13032020.
21. Diabetes 130-US Hospitals for Years 1999-2008, 2014, Available from: https://archive.ics.uci.edu/ml/datasets/Diabetes+130-US+Hospitals+For+Years+1999-2008.
22. X. Zhu, *Research and Implementation of Differential Privacy Protection Technology under Federated Learning*, Master thesis, Nanjing University of Posts and Telecommunications in Nanjing, 2021. https://doi.org/10.27251/d.cnki.gnjdc.2021.000896