applied sciences

MDPI

Article

Secure and Efficient Federated Gradient Boosting Decision Trees

Xue Zhao, Xiaohui Li *, Shuang Sun and Xu Jia

School of Electronics and Information Engineering, Liaoning University of Technology, Jinzhou 121000, China
* Correspondence: dxxylxh@lnut.edu.cn

Abstract: In recent years, federated GBDTs have gradually replaced traditional GBDTs, and become the focus of academic research. They are used to solve the task of structured data mining. Aiming at the problems of information leakage, insufficient model accuracy and high communication cost in the existing schemes of horizontal federated GBDTs, this paper proposes an algorithm of gradient boosting decision trees based on horizontal federated learning, that is, secure and efficient FL for GBDTs (SeFB). The algorithm uses locality sensitive hashing (LSH) to build a tree by collecting similar information of instances without exposing the original data of participants. In the stage of updating the tree, the algorithm aggregates the local gradients of all data participants and calculates the global leaf weights, so as to improve the accuracy of the model and reduce the communication cost. Finally, the experimental analysis shows that the algorithm can protect the privacy of the original data, and the communication cost is low. At the same time, the performance of the unbalanced binary data set is evaluated. The results show that SeFB algorithm compared with the existing schemes of horizontal federated GBDTs, the accuracy is improved by 2.53% on average.

Keywords: horizontal federated learning; data security; gradient boosting decision tree; privacy protection; LSH

Citation: Zhao, X.; Li, X.; Sun, S.; Jia, X. Secure and Efficient Federated Gradient Boosting Decision Trees. *Appl. Sci.* **2023**, *13*, 4283. https://doi.org/10.3390/app13074283

Academic Editor: Luis Javier Garcia Villalba

Received: 17 February 2023
Revised: 25 March 2023
Accepted: 26 March 2023
Published: 28 March 2023

1. Introduction

In traditional machine learning, a well-trained machine learning model relies heavily on the data of a large number of users. However, the security and privacy problems of users' data cause many users' data to be sensitive and cannot be directly uploaded to the central server, which greatly limits the flexibility of data application. Therefore, Mcmahan et al. [1] put forward Federated Learning (FL) technology. The goal of FL is to make the model effect as close as possible to that of traditional machine learning methods under the condition of ensuring the safety of users' private data, and to realize collaborative modeling and improve the model effect under the premise of legal compliance. In recent years, FL has been widely used in joint modeling, which can be divided into horizontal federated learning [2], vertical federated learning [3] and federated transfer learning [4].

However, due to the differences in geographical location and time distribution of clients, the original data of federated learning system is often non-iid, which will adversely affect the accuracy and convergence speed of the model [5], and has been identified as a basic challenge of FL [6]. At present, many federated deep learning algorithms have optimized the problems caused by non-iid data [7–9].

Federated learning is influenced by heterogeneous devices and limited network bandwidth, which leads to its computing and communication efficiency becoming the biggest challenge that hinders its practical application. Recently, several federated deep learning algorithms have been studied to reduce the communication efficiency of the model [10–12].

At present, most of the research on FL focuses on training deep learning models with privacy protection [13–17]. Although the deep learning model is very powerful for a series of real-world tasks in the federated scene, it is easily defeated by the "simpler" model

when faced with tabular datasets, one of which is the gradient boosting decision trees (GBDTs) [18–21]. The model consists of several decision trees. The decision trees are trained by gradient boosting method, and then weak learners are constructed in turn to minimize the loss function. It has many advantages such as fast training speed, high accuracy and strong interpretability. Many GBDTs frameworks, such as XGBoost [22], LightGBM [23] and CatBoost [24], have been used in different fields [25–27] and have high learning efficiency and prediction performance.

Recently, several studies have combined GBDTs with FL, such as GBDTs under secure training in federated setting [28–32]. These methods usually rely on cryptographic techniques, such as homomorphic encryption (HE) or secure multiparty computing (MPC). This dependence on heavyweight cryptography (such as HE or MPC) often makes methods computationally intensive or require a large number of communication rounds, making it difficult for them to extend to more than a few participants.

Through the above analysis, the existing federated learning scheme has the problem that the model accuracy is affected by non-iid data. This paper mainly studies the federated learning scheme that can better fit the non-iid data and improve the model accuracy. Secondly, the existing federated optimization methods are mostly based on deep learning algorithms. This paper mainly studies the optimization algorithm of federated GBDTs, which can make up for the shortcomings of slow training speed and unexplained model of federated deep learning algorithms in structured data mining. Finally, the existing GBDTs schemes under secure training in federated environment has the problems of complicated calculation and high communication cost. This paper mainly studies the federated GBDTs scheme which can protect the original data security while reducing the calculation and communication cost.

2. Related Work

This paper mainly studies horizontal FL schemes for GBDTs. Each party has its own data, and these data have the same features. The existing horizontal FL schemes for GBDTs has some limitations.

Zhao et al. [33] designed a distributed GBDTs scheme (TFL), in which all parties take turns to train a differential privacy decision tree. This method only uses the local datasets of data participants to update the tree, which leads to low model accuracy. Yamamoto et al. [34] put forward a federated GBDTs model (eFL-Boost) which focuses on properly allocating local computing (executed by each data owner alone) and global computing (executed by all owners in cooperation). Although it can improve the prediction performance to a certain extent, the accuracy of the model is greatly affected by non-iid datasets. Tian et al. [35] proposed a GBDTs of private FL (FederBoost) to construct accurate trees through gradient histograms. However, this method brings problems such as information leakage. Li et al. [36] put forward SimFL, which uses a weighted gradient boosting method to model a single tree, which is a new direction of GBDTs algorithm under the FL framework. Unfortunately, when training large-scale datasets, the communication overhead of each iteration is very large. Therefore, the existing horizontal FL schemes of GBDTs has some problems, such as insufficient prediction accuracy, model accuracy greatly affected by non-iid datasets, information leakage and high communication cost.

To sum up, this paper proposes a Secure and efficient horizontal FL schemes for GBDTs (SeFB). The main contributions of this paper are as follows: (1) Aiming at the influence of non-iid datasets on the model accuracy, this paper uses Locality Sensitive Hash (LSH) in the GBDTs of FL, and uses coordinator to count the similar instances of participants, fully absorbing the gradient information of each participant, and measuring the importance of the gradient of each participant to updating the tree model, so as to stimulate and motivate participants with high-quality datasets to make greater contributions. (2) Aiming at the existing GBDTs schemes under secure training in federated environment has the problems of complicated calculation and high communication cost. This paper mainly adopts the master–slave architecture, and the high-performance server is responsible for the global

complex operations, such as calculating the weighted gradients and global leaf weights. Participants are responsible for simple operations, such as calculating hash values, building trees and updating models. Reduce the calculation burden of participants, and at the same time reduce the number of communications. (3) Experimental analysis shows that the algorithm can protect the privacy of the original data, and the communication cost is low. At the same time, the performance of imbalanced binary data sets is evaluated (ROC AUC and F1-score as indicators). The results show that our SeFB algorithm has higher prediction accuracy than the current horizontal federated GBDTs scheme.

3. Preliminaries

3.1. Locality Sensitive Hashing (LSH)

Nearest neighbor search plays an important role in machine learning, but it has some limitations when dealing with high-dimensional data. In order to solve this problem, Gionis et al. [37] proposed locality sensitive hash (LSH), whose main idea is to map the data in the original space through the LSH function family. The probability that the hash values of similar points are equal is large, while the probability that the hash values of dissimilar points are equal is small. In LSH, the same hash value will correspond to different input data. Therefore, LSH is used to protect user data in keyword search [38] and recommendation system [39].

In 2004, Dater et al. [40] proposed LSH based on p-stable distribution, and defined the hash function as

$$F_{a,b}(v) = \left\lfloor \frac{a * v + b}{\gamma} \right\rfloor \tag{1}$$

where a is a randomly selected k-dimensional vector satisfying p-stable distribution. The dot product $a * v$ of vector a with vector v maps each vector onto a straight line; b is a random number with uniform distribution in $[0, \gamma]$ range; γ is the segment length of the segment on the straight line. Hash function divides a straight line into several segments with equal length γ, giving the same hash value to the points mapped to the same segment, and giving different hash values to the points mapped to different segments.

3.2. Federated Learning for GBDTs

3.2.1. GBDTs

Gradient Boosting Decision trees (GBDTs) is a machine learning algorithm to improve decision trees. After multiple weak learners train regression trees through local data sets, they are aggregated into a group of trees in a specific order, thus forming strong learners. As shown in Figure 1, in each tree, whenever the weak learner inputs instance data, its prediction results are divided into a certain leaf node. In this way, the leaf weights of the same prediction results are accumulated to obtain the prediction results of the strong classifier.

$$\hat{y} = \sum_{t=1}^{T} f(x) \tag{2}$$

where T represents the number of decision trees and $f_t(x)$ represents the prediction result of the T decision tree.

Although GBDTs can obtain good prediction results, it cannot make full use of the local hardware resources of weak learners, and it also brings challenges to the data security of weak learners. Therefore, the literature [28] combines GBDTs with federated learning to achieve better results.

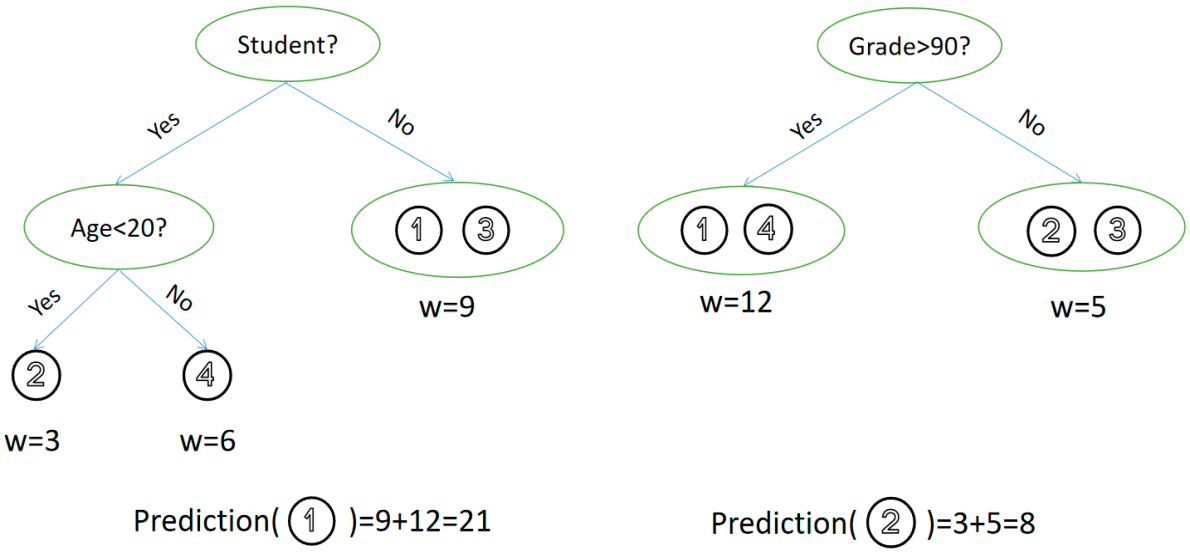


Figure 1. An example of GBDTs.

3.2.2. Horizontal Federated Learning for GBDTs

Although GBDTs is applied to federated learning, the data distribution of weak learners leads to differences in their corresponding training processes. As the data distribution is horizontal, the horizontal federated learning for GBDTs is proposed, and its centralized training process is shown in Figure 2.

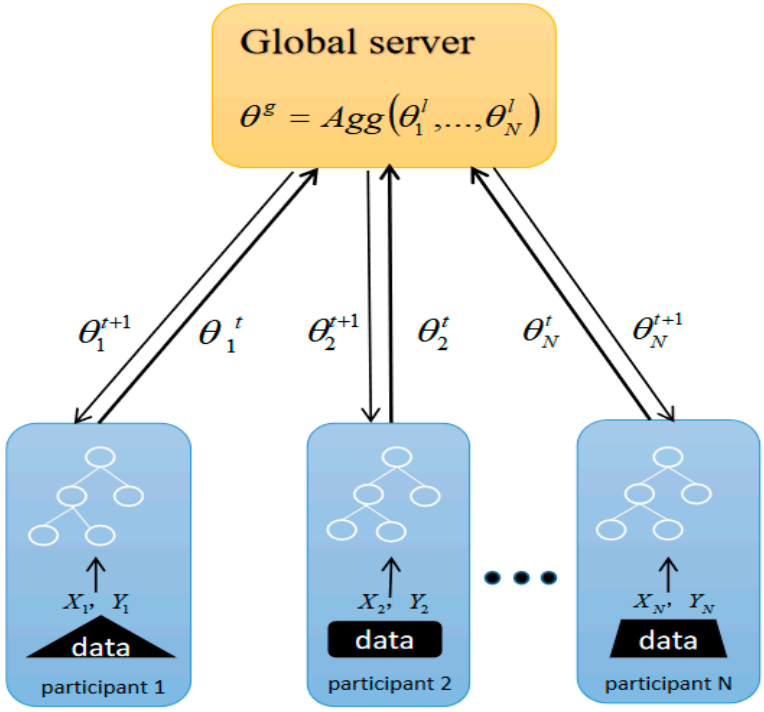


Figure 2. Horizontal federated learning for GBDTs.

Next, we briefly introduce the GBDTs training algorithm for federated learning. Formally, given a loss function l and a dataset with n instances and d features: $D = \{(x_i, y_i)\}(|D| = n, x_i \in R^d, y_i \in R)$, GBDTs minimizes the following objective functions [22]:

$$\widetilde{L} = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \quad (3)$$

where, L is the loss function, y_i represents the ith target value, $\hat{y}_i$ represents the predicted value of the previous tree, $\Omega(f)$ represents the penalty term of tree complexity, T_1 represents the number of leaves and ω represents the weight of leaves. Each f_k corresponds to a decision tree. By training the model additively, GBDTs minimizes the following objective function at the tth iteration:

$$\tilde{L}^{(t)} = \sum_{i=1}^{N}\left[g_i f_t(x_i) + \frac{h_i f_t^2(x_i)}{2}\right] + \Omega(f_t) \tag{4}$$

where, $g_i = \partial_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}^{(t-1)}\right)$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l\left(y_i, \hat{y}^{(t-1)}\right)$ are the first-order and second-order gradient statistics of the loss function. Let $I = I_L \cup I_R$, where I represents the instance set of the parent node. I_L and I_R represent the instance set of the left child node and the right child node after splitting the parent node, respectively. The gain value of each split is expressed as:

$$L_{split} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda}\right] - \gamma \tag{5}$$

Since the loss function is quadratic, the leaf weight $\hat{w}_j$ and the splitting score of a node can be expressed as follows.

$$\hat{\omega_j} = -\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i + \lambda} \tag{6}$$

$$score = \frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{G^2}{H + \lambda} \tag{7}$$

where G_L and G_R (or H_L and H_R) represent the sum of the data g (or h) after splitting the left and right nodes, respectively.

The literature [22] shows that gradient can represent the importance of an instance. For an instance $X_q^b \in I_b$, let $W_{bq}^p = \left\{k \middle| S_{bq}^p = q\right\}$,which contains the IDs of the instances in I_p that are similar with X_q^b. Let g_q^b and h_q^b denote the first order and second order gradients of loss function at X_q^b, respectively. When the builder B is building a new tree at the t-th iteration, WGB minimizes the following objective function [36].

$$\tilde{L}_w^{(t)} = \sum_{x_q^b \in I_b} [G_{bq} f_t(X_q^b) + \frac{1}{2} H_{bq} f_t^2(X_q^b)] + \Omega(f_t) \tag{8}$$

where $G_{bq} = \sum_p \sum_{i\in W_{bq}^p} g_i^p, H_{bq} = \sum_p \sum_{i\in W_{bq}^p} h_i^p$

Compared with the objective in Equation (4), Equation (8) only uses the instances of I_b. Instead of using the gradients g_q^b, h_q^b of the instance X_q^b, we use G_{bq}, H_{bq} which are the sum of the gradients of the instances that are similar with X_q^b (including X_q^b itself). Table 1 shows the list of mathematical notations of variables.

Table 1. Mathematical notations of variables.

Notations	Descriptions
n	Total number of similar instances for all data participants.
n_p	Number of similar instances of the p-th participant
p	Data provided by all participants
X_p	Local dataset of the p-th participant
I_p	Instance set of participant p
G_{bq}^i, H_{bq}^i	Weighted gradient of builder B
G_p, H_p	Sums of the gradients (g_i and h_i in (4)) corresponding to each leaf
t_j	The j-th tree
T_j	The updated j-th tree

4. Secure and Efficient FL for GBDTs

4.1. The SeFB Framework

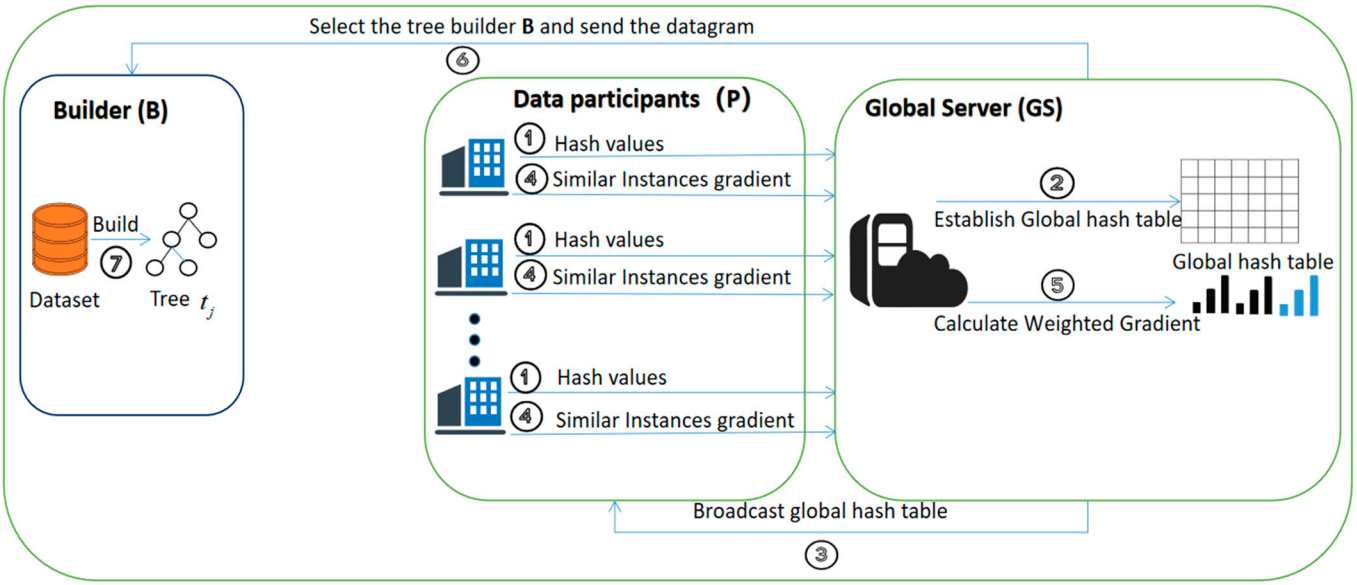
Because the GBDTs model is optimized in function space instead of Euclidean space through gradient descent iterative process, the implementation of federated learning setting of GBDTs algorithm is quite different from that of standard paradigm. Because the weights in this model are specific leaves generated from local data, traditional model exchange technology cannot be applied to combine model parameters at the aggregator. That is to say, it is impossible to train independent weak learner models locality on each side and integrate the tree structures generated by them on the aggregator.

At present, the frontier research of GBDTs in horizontal federated learning probably follows the following ways. Participants train a decision tree and send it to other participants. Participants who receive the decision tree will update their local gradient and train the next decision tree, which will take turns until the final task is completed. However, the problem of this scheme is that when the local data set of a participant is very skewed, the quality of the trained decision tree is low and more seriously, the low-quality decision tree will affect all subsequent training processes (this process is irreversible, because the subsequent decision tree will depend on the prediction residuals of all previous models). Therefore, the determination of the first decision tree is particularly important.

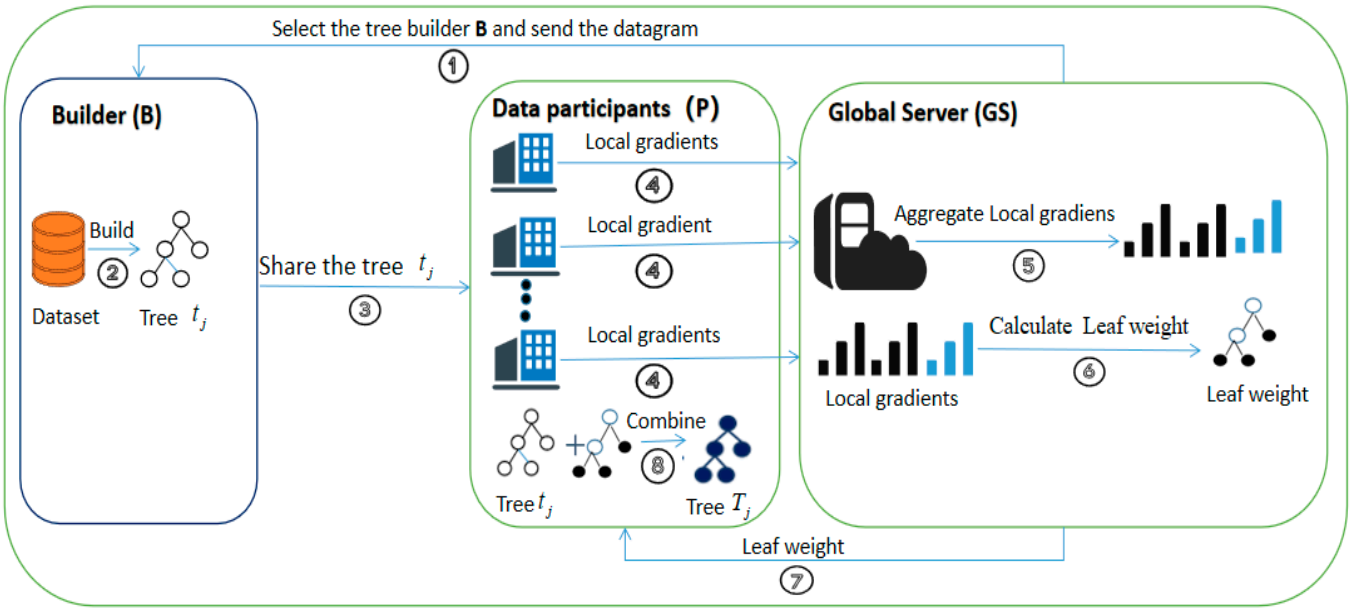
We hope that in the process of building the first tree, we will select the participant with the best quality and distribution of its data set as the builder, and fully absorb the gradient information of other participants to alleviate the influence of the local skewed data set on the decision tree model. In SeFB, we first collect information about similar instances through LSH, and then use gradient aggregation method to build the tree. In addition, in terms of privacy protection, we have not exposed the original data of each data owner.

In the process of subsequent update iteration, if the gradients of similar instances need to be obtained from the participants every time, this means that the communication cost is directly related to the number of participants, which is not suitable for the practical application scenarios of large-scale data sets. Therefore, we adopt the master–slave architecture, and the high-performance server is responsible for the global complex operations, such as calculating the weighted gradients and global leaf weights. Participants are responsible for simple operations, such as calculating hash values, building trees and updating models. The calculation burden of participants is reduced, and at the same time, the communication times are reduced. In addition, in terms of accuracy, because the leaf weights are directly related to the output of the model, them contribute more to the improvement of model accuracy. Therefore, we add global leaf weights to the constructed tree to update the model and improve the accuracy of the model.

A schematic of the SeFB framework is shown in Figure 3. It includes the builder (B), the data participants (P) and the global server (GS). The builder is one of the data participants and is responsible for building the tree. The global server is responsible for establishing the global hash table and aggregating the local gradients to calculate the global leaf weight. Data participants is responsible for computing the hash values and gradients and sending them to the global server.



(**a**)



(**b**)

Figure 3. This is the framework of SeFB. (**a**) The stage of building a tree. (**b**) The stage of updating the tree.

4.2. The Algorithm Design of SeFB

SeFB has two main stages: building the tree and updating the tree. In the stage of building the tree, the data participant first calculates the hash value by using the randomly generated Local Sensitive Hash (LSH) function and uploads it to the GS. Then, the GS builds a global hash table through the collected hash values and broadcasts it to all data participants. Finally, the builder (one of the data participants) can build the tree structure by using the global hash table and similarity information without accessing the original data of other parties. In the stage of updating the tree, the data participants calculate the local gradient according to the tree shared by the builder and send it to the GS. Then, the GS calculates and shares the leaf weights by aggregating the gradients. Finally, each data participant combines the tree and the leaf weights to obtain a new tree, which is added to the global model. It is worth noting that once a tree is built on one side, it will be sent to

all parties to update the gradient. All the decision trees we obtain will serve as the final learning model.

4.3. The Algorithm Description of SeFB

Step 1 Input LSH functions $\{F_k\}_{k=1,2...L,}$, instance set for each participant I_p, number of updates U and data set for each participant X_p,

Step 2 Data participants calculate the hash values of the local instances according to LSH functions, and upload them to coordinator GS.

Step 3 GS constructs a global hash table according to the hash values and broadcasts it. At the same time, GS sends a request to participants to obtain the cumulative gradients of their local similar instances.

Step 4 Data participants construct a matrix of similar instances $s = S^i_{pq}$ according to the global hash table, and calculate the cumulative gradients $G^i_{pq} \leftarrow G^i_{pq} + g^i_q$, $H^i_{pq} \leftarrow G^i_{pq} + h^i_q$ of local similar instances according to the global hash table and upload them to GS.

Step 5 GS counts the similar instance information of each participant, and selects the participant with the largest number of similar instances in turn as the builder B (the active party) of each update iteration.

Step 6 GS calculates the weighted gradients G^i_{bq}, H^i_{bq} of the B according to the similar information between each instance X^b_q in the I_b and I_p.

Step 7 GS sends datagram (weighted gradients and tree-building mark) to B.

Step 8 B uses the previous tree t_{j-1}, data set X_b and weighted gradients G^i_{bq}, H^i_{bq} rewrites the loss function according to Formula (8) and constructs the tree t_j according to Formula (4), and sends t_j to other participants.

Step 9 Participants calculate the sum of gradients corresponding to each leaf according to t_j and data set X_p to obtain G_p and H_p, and upload them to GS.

Step 10 GS aggregates G_p and H_p of each participant to obtain $G = \sum_{p\in P} \frac{n_p}{n} G_p$ and $H = \sum_{p\in P} \frac{n_p}{n} H_p$, and calculates the global leaf weights $\omega_i = -\frac{G}{H+\lambda}$ with G and H, and then returns them to the participants.

Step 11 Participants obtain the leaf weights ω, they combine the tree t_j with ω and to obtain T_j.

Step 12 Repeat step5-step11, and output a decision tree in each cycle until the update iteration ends *(j > U)*.

Step 13 Output global model $\{T_1, \ldots, T_U\}$

In step 2, for each instance, the purpose of using LSH is to obtain similar instance IDs. In order to obtain the similar information between any two instances in the federated data without exposing the original data, the p-stable LSH function is adopted.

In step 5, GS selects the participant with the largest number of similar instances as the builder after obtaining the cumulative gradients of local similar instances of each participant. In other words, the builder at this time is the participant with the best data quality and distribution.

In step 6, the builder changes the local loss function through the weighted gradients. The weighted gradients at this time fully absorb the gradient information of all participants, which can ensure that the constructed model has better generalization.

In step 7, GS can measure the importance of a participant's gradient to updating the tree model through $\frac{n_p}{n}$ ($n = \sum_{p=1}^{U} n_p$), improve the contribution of high-quality data sets, and thus speed up the convergence of the model.

5. Discussion

5.1. Security Analysiss

We verify the security of the SeFB algorithm proposed in this paper through the following three parts. Firstly, the necessity of the security strategy of this algorithm is

analyzed. Secondly, the index to evaluate the security of the algorithm is introduced. Finally, the security of this algorithm is proved.

5.1.1. Necessity Analysis of Security Strategy of SeFB

Assuming that SeFB algorithm does not add any security policy, step 2 shares the original data to coordinator GS in plaintext, which will lead to the risk of data leakage. Therefore, it is necessary to use LSH function to calculate hash value to encrypt the original data. In step 10, if GS obtains the gradient information of each participant and the model structure information at the same time, GS can obtain the original data information through model backstepping attack. Therefore, it is necessary that GS only allows obtaining gradient information.

5.1.2. Evaluation Index of the Security of SeFB

Privacy model. a two-party security model was proposed by Du et al. [41] and adopted by Liu et al. [42] in 2020. We extend the model to multiple parties and obtain the following privacy definition:

Definition 1 (privacy model). *Assume that all participants are semi-city real adversaries. For a protocol C, execution, where* $O_1, O_2, \ldots O_M$ *and* $I_1, I_2, \ldots, I_M$ *are the output and input of participant* $P_1, P_2, \ldots, P_M$*, respectively, is said to be secure for P if there exists an infinite number of tuples satisfying* $(O_1, O_2, \ldots, O_M) = C(I_1, I_2, \ldots, I_M)$.

When potential risks such as inference attacks are encountered, definition 1 has a weaker privacy level compared to the security definition in secure multi-party computation [43]. Due to the literature [36], the corresponding heuristic model is proposed based on this privacy model.

Definition 2 (heuristic model). *If* $L < d$*, where L is the number of hash functions and d is the number of dimensions of the training data. In short, if the number of unknowns (i.e., d) is greater than the number of equations (i.e., L), then there exists an infinite number of solutions.*

5.1.3. Proof of the Security of SeFB

Assume that the global server GS is honest and curious. We prove the security of SeFB by analyzing whether GS can infer the original data of federated participants.

In step 2, participants perform LSH functions to convert data into hash values. Here, the number of hash functions (L) is selected to be less than the number of dimensions of the training data (d). According to Definition 2, an opponent will obtain an infinite number of possible inputs when attacking with background knowledge. Therefore, the honest and curious coordinator GS cannot obtain the original data of the participants.

In step 12, GS only obtained the cumulative gradients G_p and H_p of each leaf of each data participant, but did not obtain the global model. Therefore, the conditions of model inversion cannot be met.

5.2. Communication Costs

In this section, we discuss the communication cost required for each update of each scheme.

In step 5–step 11, SeFB needs four communications for each update. First, GS sends datagram (weighted gradients and tree-building mark) to builder B. Secondly, the builder B shares the constructed tree with all participants, then the participants transmit the local gradients to the GS, and finally the GS calculates the global leaf weights and transmits them.

Therefore, after U update iterations, the total communication cost of SeFB is $O(U)$, which is the same as TFL and eFL-Boost. However, in each update of FederBoost, the gradient histogram corresponding to each node must be communicated several times according to the depth of the tree, and the total communication cost is $O(HU)$. SimFL

needs to obtain instance gradients from other participants every time it is updated, and the total communication cost is $O(|I_m|U)$. Therefore, SeFB can train the model with lower communication cost than FederBoost and simFL. Table 2 shows the communication times required for each scheme to update the global model each time, where H represents the depth of the tree and the I_m represents local instance set.

Table 2. Communication times required for each scheme to update the global model each time.

Schemes	Communication Times
TFL	1
eFL-Boost	3
SeFB	4
FederBoost	2H
simFL	$\|I_m\|$

5.3. Computational Complexity

In step 5–step 11, SeFB needs to be computed four times for each update: (1) In step 6, GS calculates the builder's weighted gradients through simple addition operation, and the computational complexity is $O(N)$. (2) In step 8, builder B builds a tree, and the total number of data and features owned by the p th participant are N_p and N_f, respectively, so the computational complexity of this process is $O\left(N_d N_f\right)$. (3) In step 9, each participant calculates the gradients of each leaf. Assuming that the number of data participants is N, there are M data participants, so the operation can be completed with a computational complexity of $O(MN)$. (4) In step 10, GS calculates the leaf weights, and the computational complexity of this process is $O\left(2^H\right)(2^H \gg NM)$, where H is the depth of the tree. Through the above analysis, it is concluded that the process of building a tree by B is the most important.

The computational complexity required for each update of each scheme is shown in Table 3. In all schemes, all participants share the complete GBDTs, so the computational complexity of each scheme is equal to that of normal GBDTs.

Table 3. Computational complexity required by each scheme for each update.

Schemes	Computational Complexity
TFL	$O\left(N_d N_f\right)$
eFL-Boost	$O\left(N_d N_f\right)$
SeFB	$O\left(N_d N_f\right)$
FederBoost	$O\left(N_d N_f\right)$
simFL	$O\left(N_d N_f\right)$

6. Experiments

6.1. Experimental Setups

To validate the accuracy of the proposed SeFB model, we compare SeFB with two approaches: (1) Non indicates the case that a participant trains a common GBDTs using federated data from all parties, regardless of privacy concerns. (2) Independent is a case where training the common GBDTs is carried out independently by local data for each party. We also compare SeFB with two other federated learning for GBDTs schemes. TFL is the scheme from Zhao et al. [33]. eFL-Boost is the scheme from Yamamoto et al. [34]. Because the training process in FederBoost [35] is the same as that in GBDTs, it has the same accuracy as that in Non, so the experimental results of FederBoost are not shown.

In this experiment, the test runs on AMD Ryzen 5 3600 3.6GHz CPU and 16GB RAM. SeFB algorithm was implemented using the python language. The experimental data

were used, Credit 1 dataset, Credit 2 dataset and Adult dataset. Table 4 lists the basic information of the three datasets used in the experiments. It is worth noting that they are all unbalanced datasets.

Table 4. Experimention on the dataset. The task in the following dataset is binary classification.

Data Set	Data	Features	Positive Data Ratio
Credit 1 [44]	284,805	30	0.2%
Credit 2 [45]	120,269	10	7%
Adult [46]	32,651	14	24%

Here are three evaluation indicators that were originally used for this experiment: (1) F1-score: harmonic mean of precision and recall. It supports model robustness evaluation in experiments with unbalanced data sets. (2) ROC-AUC: area under the ROC curve. It can ignore the influence of threshold selection in classification and evaluate the classification effect of the model. (3) Test errors. For any given problem, a lower test error value means a better prediction.

6.2. Experimental Results

In the practical use of federated learning, the data quality and distribution of participants are often uncontrollable, and it is impossible to require all participants' data to meet independent and identical distribution [47]. Therefore, this experiment divides the training data set according to the method of the literature [48] to simulate the unbalanced data distribution in federated learning.

In the experiment, the datasets are divided into two parts by the unbalanced segmentation method with a ratio of $\theta = 80\%$, which are allocated to both A and B, respectively. The test error is shown in Table 5. The experimental results show that: firstly, the test error of SeFB on data part A and B is always lower than Independent, and the prediction accuracy can be improved by SeFB. Secondly, the test error of SeFB is close to Non; thirdly, compared with TFL and eFL-Boost, SeFB has lower test error. The test errors of TFL are always larger than Independent, which hinders their adoption in practice.

Table 5. Test errors of different scheme.

	Credit 1	Credit 2	Adult
Non	21.8%	16.5%	18.6%
Independent	25.7%	19.3%	21.3%
TFL	27.7%	21.4%	23.5%
eFL-Boost	24.6%	18.2%	20.7%
SeFB	22.4%	17.3%	19.6%

6.2.1. Performance Impact of the Number of Participants

The core idea of federation learning is to enable multiple participants to jointly train a model to improve prediction accuracy. Therefore, to investigate the effectiveness of federation learning, we set up different numbers of participants in this experiment, while fixing the amount of data for each participant, and evaluated the prediction performance for each case.

The results of the experiment are shown in Table 6 and Figure 4. Each participant owns 10% of all the data. We adjusted the number of participants from 2 to 10. The results show that the performance of all models increases when the number of participants increases. Moreover, in all cases, all the indexes of SeFB are better than Independent, TFL and eFL-Boost.

Table 6. Performance impact of the number of data participants.

Dataset	Schemes	F1-Score					ROC-AUC				
		2	4	6	8	10	2	4	6	8	10
Credit 1	Non	0.812	0.826	0.843	0.857	0.861	0.972	0.975	0.975	0.978	0.980
	Independent	0.763	0.769	0.768	0.769	0.751	0.963	0.963	0.965	0.966	0.966
	TFL	0.792	0.797	0.799	0.805	0.814	0.966	0.972	0.974	0.975	0.976
	eFL-Boost	0.801	0.820	0.839	0.841	0.843	0.970	0.972	0.973	0.974	0.975
	SeFB	0.811	0.824	0.841	0.851	0.858	0.971	0.974	0.975	0.977	0.979
Credit 2	Non	0.901	0.904	0.906	0.908	0.910	0.922	0.925	0.927	0.930	0.934
	Independent	0.882	0.891	0.893	0.893	0.894	0.911	0.912	0.913	0.914	0.916
	TFL	0.889	0.891	0.897	0.900	0.902	0.916	0.917	0.919	0.922	0.928
	eFL-Boost	0.900	0.902	0.904	0.906	0.907	0.920	0.923	0.925	0.927	0.931
	SeFB	0.901	0.903	0.906	0.908	0.910	0.922	0.924	0.926	0.930	0.933
Adult	Non	0.722	0.731	0.740	0.748	0.767	0.888	0.900	0.902	0.906	0.911
	Independent	0.662	0.671	0.678	0.683	0.698	0.843	0.844	0.845	0.845	0.847
	TFL	0.689	0.693	0.710	0.718	0.732	0.873	0.875	0.877	0.888	0.896
	eFL-Boost	0.718	0.724	0.733	0.740	0.752	0.883	0.885	0.889	0.896	0.903
	SeFB	0.721	0.729	0.736	0.745	0.763	0.885	0.888	0.897	0.902	0.909

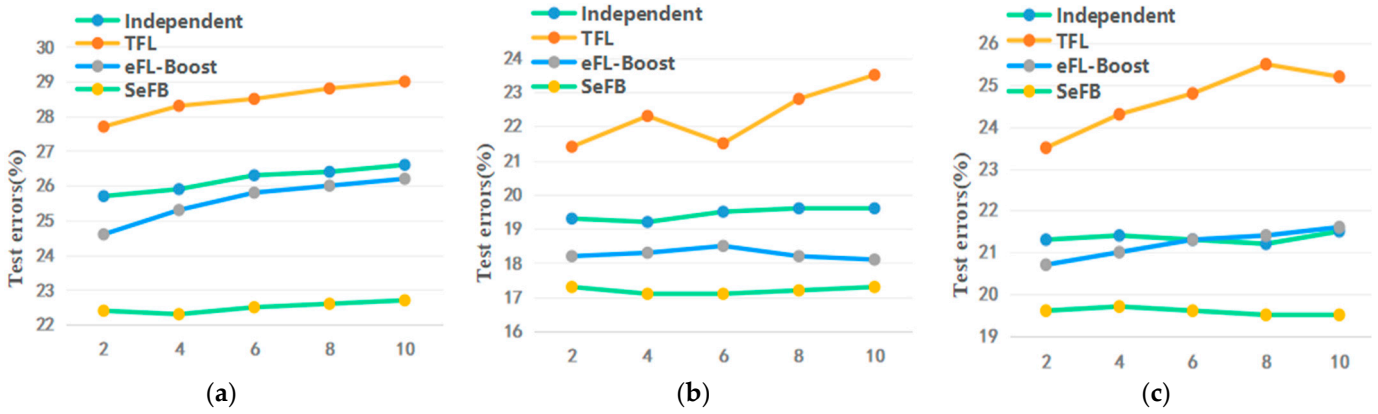


Figure 4. Impact of number of participants on Test Errors. (**a**) Credit 1 dataset, (**b**) Credit 2 dataset, (**c**) Adult dataset.

Figure 4 shows the test errors with different numbers of participants, and the horizontal coordinate indicates the number of participants. The results show that the upper limit of SeFB generalization error may increase with the increase in the number of participants. However, in most cases, the test error of SeFB is lower than the minimum error of Individual and eFL-Boost. The test error of TFL changes sharply with the increase in the number of participants, while SeFB is much more stable.

6.2.2. Performance Impact of Participants' Data Imbalance Ratio

In the practical use of federation learning, the amount of data provided by federated participants is often different. Therefore, federated learning algorithms should be robust to unbalanced distributions of data. This experiment used the partitioning method from a previous study [48] to assign skewed local datasets to the participants. We set the number of participants to 2 and adjust the imbalance ratio θ from 60% to 80%. One of the participants received $\frac{\theta * N_{class0}}{2}$ instances with label 0, and $\frac{\theta * N_{class1}}{2}$ instances with label 1, and the other participant received the opposite.

The experimental results are shown in Table 7 and Figure 5. We can observe that the performance of all models decreases with the increase in imbalance ratio θ. Secondly, the performance of SeFB is able to approach Non. Thirdly, in all cases, the performance of TFL, eFL-Boost and Independent tends to be lower, while the performance of SeFB is always better than them. Thus, SeFB is able to achieve performance enhancement through FL, even when the data of the participants are not balanced.

Table 7. Performance impact of participants' data imbalance ratio.

Dataset	Schemes	F1-Score				ROC-AUC			
		60%	70%	80%	90%	60%	70%	80%	90%
Credit 1	Non	0.823	0.819	0.812	0.807	0.989	0.984	0.978	0.973
	Independent	0.773	0.767	0.763	0.756	0.976	0.971	0.966	0.959
	TFL	0.797	0.795	0.792	0.782	0.979	0.977	0.975	0.967
	eFL-Boost	0.817	0.806	0.801	0.791	0.982	0.979	0.974	0.971
	SeFB	0.819	0.815	0.811	0.805	0.987	0.982	0.977	0.972
Credit 2	Non	0.914	0.908	0.901	0.885	0.937	0.935	0.930	0.926
	Independent	0.892	0.886	0.882	0.875	0.920	0.917	0.914	0.909
	TFL	0.894	0.886	0.889	0.879	0.931	0.926	0.922	0.916
	eFL-Boost	0.909	0.905	0.900	0.891	0.932	0.929	0.927	0.921
	SeFB	0.911	0.905	0.901	0.885	0.936	0.933	0.930	0.924
Adult	Non	0.735	0.729	0.722	0.713	0.925	0.911	0.906	0.902
	Independent	0.687	0.675	0.662	0.650	0.867	0.856	0.845	0.833
	TFL	0.701	0.693	0.689	0.671	0.899	0.896	0.888	0.872
	eFL-Boost	0.729	0.723	0.718	0.702	0.912	0.901	0.896	0.891
	SeFB	0.731	0.728	0.721	0.711	0.918	0.909	0.902	0.901

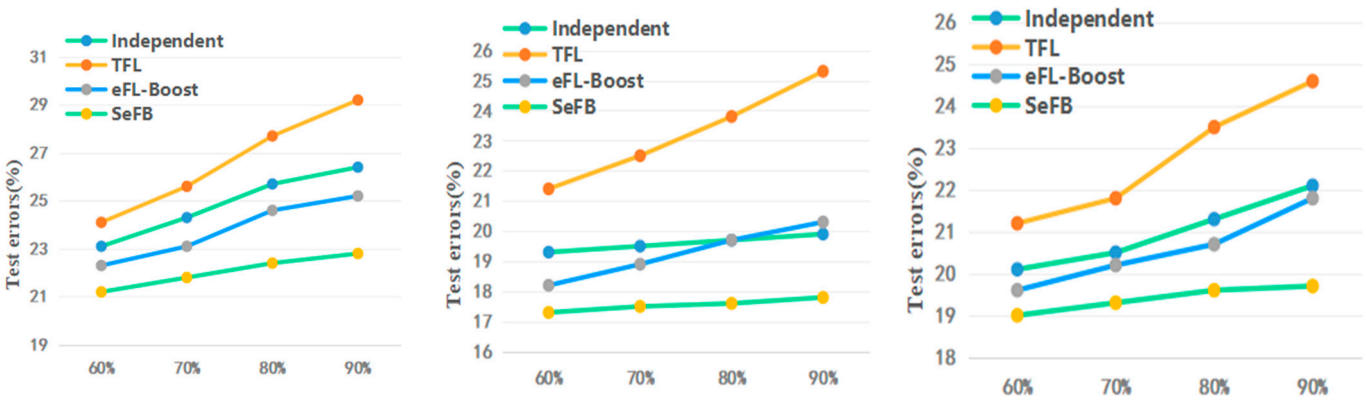


Figure 5. Impact of participants' data imbalance ratio on Test Errors. (**a**) Credit 1 dataset, (**b**) Credit 2 dataset, (**c**) Adult dataset.

Figure 5 shows the test errors with participants' data imbalance ratio, where the horizontal coordinate indicates the imbalance ratio θ. With the increase in θ, SeFB has better prediction performance than Independent, TFL and eFL-Boost.

6.2.3. Performance Impact of Data Volume Size

In most cases, the performance of a machine learning model is influenced by the amount of training data. Therefore, in this experiment, we fixed the number of participants to be three and varied the amount of training data for each participant to evaluate the federated learning model.

The results of the experiment are shown in Table 8 and Figure 6. In Figure 6, the horizontal coordinates in the figure indicate the data ratio of the original dataset, and the amount of data per participant increases as the total number of participants decreases. In this case, the amount of data per participant is 30%, 60% and 90% of the total amount of data. We can observe that the prediction performance of all GBDTs schemes improves as the amount of data increases. Among them, SeFB outperforms Independent, TFL and eFL-Boost.

Table 8. Performance impact of data volume size.

Dataset	Schemes	F1-Score			ROC-AUC		
		30%	60%	90%	30%	60%	90%
Credit 1	Non	0.811	0.835	0.857	0.968	0.974	0.980
	Independent	0.751	0.782	0.801	0.946	0.962	0.966
	TFL	0.791	0.811	0.836	0.952	0.965	0.973
	eFL-Boost	0.801	0.823	0.846	0.964	0.969	0.975
	SeFB	0.805	0.837	0.852	0.965	0.972	0.979
Credit 2	Non	0.898	0.902	0.907	0.922	0.927	0.932
	Independent	0.894	0.898	0.903	0.901	0.909	0.914
	TFL	0.896	0.899	0.901	0.906	0.916	0.924
	eFL-Boost	0.897	0.901	0.903	0.910	0.922	0.927
	SeFB	0.898	0.901	0.905	0.919	0.925	0.931
Adult	Non	0.740	0.751	0.772	0.888	0.896	0.903
	Independent	0.679	0.693	0.714	0.854	0.857	0.862
	TFL	0.712	0.729	0.745	0.867	0.876	0.883
	eFL-Boost	0.730	0.743	0.766	0.875	0.883	0.898
	SeFB	0.736	0.749	0.772	0.885	0.891	0.902

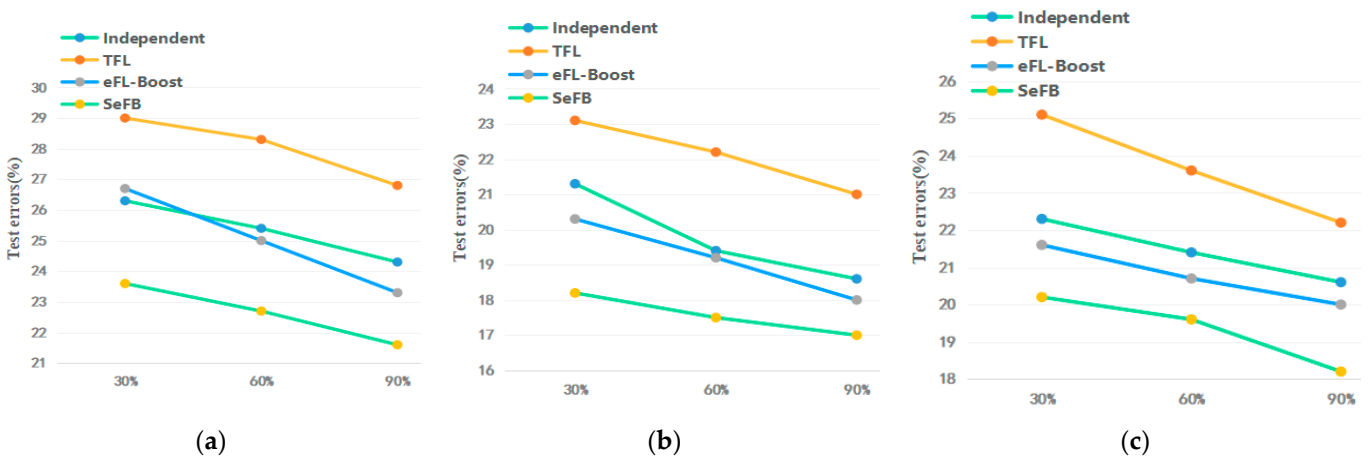


Figure 6. Impact of data volume size on Test Errors. (**a**) Credit 1 dataset, (**b**) Credit 2 dataset, (**c**) Adult dataset.

7. Conclusions

With the emergence of information leakage incidents, federated learning has been paid more and more attention because of its privacy protection characteristics. The SeFB method proposed in this paper is divided into two stages: building a tree and updating the tree. In the first stage, we use LSH to calculate similar instances, and build a tree by weighted gradients. This not only solves the problem of low accuracy of the model caused by different data distribution, but also protects the safety of the original data of each participant. In the second stage, we calculate the global leaf weight that only needs one round of communication to update the tree, and compensate the accuracy loss caused by local calculation by improving the attribute mechanism of the tree. This not only ensures the accuracy of prediction, but also greatly improves the communication efficiency.

In that discussion section of this paper, we first evaluate SeFB from three aspects: security, computational complexity and communication cost. The results show that SeFB not only protects the privacy of the original data, but also ensures the communication efficiency of the model, so it has good practical application value. Secondly, in the aspect of prediction performance, we compare SeFB with Independent, TFL and eFL-Boost through F1-score, ROC-AUC and Test errors. The experimental results show that SeFB is superior to TFL and eFL-Boost in almost all cases, and has the same performance as Non.

During the whole training process, participants need to share the gradient information of local data when updating the boosting tree, so there is still the risk of information leakage. In the future, the research will focus on the privacy security in the process of updating the federated gradient boosting decision tree model to protect the security of local model parameters.

Author Contributions: Conceptualization, X.Z. and S.S.; Methodology, X.J.; Writing—original draft, X.Z.; Supervision, X.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Liaoning Applied Basic Research Program, (2022JH2/1013-00278, 2022JH2/101300279).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Konečný, J.; McMahan, H.B.; Yu, F.X.; Richtárik, P.; Suresh, A.T.; Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv* **2016**, arXiv:1610.05492.
2. Gao, D.; Ju, C.; Wei, X.; Liu, Y.; Chen, T.; Yang, Q. Hhhfl: Hierarchical heterogeneous horizontal federated learning for electroencephalography. *arXiv* **2019**, arXiv:1909.05784.
3. Liu, Y.; Kang, Y.; Zhang, X.; Li, L.; Cheng, Y.; Chen, T.; Hong, M.; Yan, Q. A communication efficient vertical federated learning framework. *arXiv* **2019**, arXiv:1912.11187.
4. Sharma, S.; Xing, C.; Liu, Y.; Kang, Y. Secure and efficient federated transfer learning. In Proceedings of the 2019 IEEE international conference on big data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 2569–2576.
5. Li, X.; Huang, K.; Yang, W.; Wang, S.; Zhang, Z. On the convergence of fedavg on non-iid data. *arXiv* **2019**, arXiv:1907.02189.
6. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and open problems in federated learning. *Found. Trends®Mach. Learn.* **2021**, *14*, 1–210. [CrossRef]
7. Sattler, F.; Wiedemann, S.; Müller, K.R.; Samek, W. Robust and communication-efficient federated learning from non-iid data. *IEEE Trans Neural Netw. Learn. Syst.* **2019**, *31*, 3400–3413. [CrossRef] [PubMed]
8. Wang, H.; Kaplan, Z.; Niu, D.; Li, B. Optimizing federated learning on non-iid data with reinforcement learning. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1698–1707.
9. Li, Q.; Diao, Y.; Chen, Q.; He, B. Federated learning on non-iid data silos: An experimental study. In Proceedings of the 2022 IEEE the 38th International Conference on Data Engineering (ICDE), Kuala Lumpur, Malaysia, 9 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 965–978.
10. Xu, J.; Du, W.; Jin, Y.; He, W.; Cheng, R. Ternary compression for communication-efficient federated learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 1162–1176. [CrossRef]
11. Haddadpour, F.; Kamani, M.M.; Mokhtari, A.; Mahdavi, M. Federated learning with compression: Unified analysis and sharp guarantees. In Proceedings of the International Conference on Artificial Intelligence and Statistics, San Diego, CA, USA, 13–15 April 2021; pp. 2350–2358.
12. Huang, T.; Lin, W.; Wu, W.; He, L.; Li, K.; Zomaya, A.Y. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Trans. Parallel Distrib. Syst.* **2021**, *32*, 1552–1564. [CrossRef]
13. Kairouz, P.; McMahan, B.; Song, S.; Thakkar, O.; Thakurta, A.; Xu, Z. Practical and private (deep) learning without sampling or shuffling. In Proceedings of the International Conference on Machine Learning, Online, 18–24 July 2021; pp. 5213–5225.
14. Zhao, B.; Fan, K.; Yang, K.; Wang, Z.; Li, H.; Yang, Y. Anonymous and privacy-preserving federated learning with industrial big data. *IEEE Trans. Ind. Inform.* **2021**, *17*, 6314–6323. [CrossRef]
15. Li, Y.; Zhou, Y.; Jolfaei, A.; Yu, D.; Xu, G.; Zheng, X. Privacy-preserving federated learning framework based on chained secure multiparty computing. *IEEE Internet Things J.* **2020**, *8*, 6178–6186. [CrossRef]
16. Huang, X.; Ding, Y.; Jiang, Z.L.; Qi, S.; Wang, X.; Liao, Q. DP-FL: A novel differentially private federated learning framework for the unbalanced data. *World Wide Web* **2020**, *23*, 2529–2545. [CrossRef]
17. Kumar, P.; Gupta, G.P.; Tripathi, R. PEFL: Deep privacy-encoding-based federated learning framework for smart agriculture. *IEEE Micro* **2021**, *42*, 33–40. [CrossRef]
18. Elsayed, S.; Thyssens, D.; Rashed, A.; Jomaa, H.S.; Schmidt-Thieme, L. Do we really need deep learning models for time series forecasting? *arXiv* **2021**, arXiv:2101.02118.

19. Gorishniy, Y.; Rubachev, I.; Khrulkov, V.; Babenko, A. Revisiting deep learning models for tabular data. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 18932–18943.
20. Shwartz-Ziv, R.; Armon, A. Tabular data: Deep learning is not all you need. *Inf. Fusion* **2022**, *81*, 84–90. [CrossRef]
21. Grinsztajn, L.; Oyallon, E.; Varoquaux, G. Why do tree-based models still outperform deep learning on tabular data? *arXiv* **2022**, arXiv:2207.08815.
22. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.
23. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Liu, T.Y. Lightgbm: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 52.
24. Prokhorenkova, L.; Gusev, G.; Vorobev, A.; Dorogush, A.V.; Gulin, A. CatBoost: Unbiased boosting with categorical features. *Adv. Neural Inf. Process. Syst.* **2018**, 31.
25. Sun, R.; Wang, G.; Zhang, W.; Hsu, L.T.; Ochieng, W.Y. A gradient boosting decision tree based GPS signal reception classification algorithm. *Appl. Soft Comput.* **2020**, *86*, 105942. [CrossRef]
26. Juan, C.; Gen, L.; Xianhua, C. Research on travel time prediction model of freeway based on gradient boosting decision tree. *IEEE Access* **2018**, *7*, 7466–7480.
27. Jianfeng, H.; Jianliang, M. Automated detection of driver fatigue based on EEG signals using gradient boosting decision tree model. *Cogn. Neurodynamics* **2018**, *12*, 431–440.
28. Cheng, K.; Fan, T.; Jin, Y.; Liu, Y.; Chen, T.; Papadopoulos, D.; Yang, Q. Secureboost: A lossless federated learning framework. *IEEE Intell. Syst.* **2021**, *36*, 87–98. [CrossRef]
29. Deforth, K.; Desgroseilliers, M.; Gama, N.; Georgieva, M.; Jetchev, D.; Vuille, M. XORBoost: Tree boosting in the multiparty computation setting. *Proc. Privacy Enhancing Technol.* **2021**, 66–85. [CrossRef]
30. Feng, Z.; Xiong, H.; Song, C.; Yang, S.; Zhao, B.; Wang, L.; Huan, J. Securegbm: Secure multi-party gradient boosting. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1312–1321.
31. Meng, X.; Feigenbaum, J. Privacy-preserving xgboost inference. *arXiv* **2020**, arXiv:2011.04789.
32. Law, A.; Leung, C.; Poddar, R.; Popa, R.A.; Shi, C.; Sima, O.; Zheng, W. Secure collaborative training and inference for xgboost. In Proceedings of the 2020 workshop on privacy-preserving machine learning in practice, Virtual Event, 9 November 2020; pp. 21–26.
33. Zhao, L.; Ni, L.; Hu, S.; Chen, Y.; Zhou, P.; Xiao, F.; Wu, L. Inprivate digging: Enabling tree-based distributed data mining with differential privacy. In Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, Honolulu, HA, USA, 15–19 April 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 2087–2095.
34. Yamamoto, F.; Ozawa, S.; Wang, L. eFL-Boost: Efficient Federated Learning for Gradient Boosting Decision Trees. *IEEE Access* **2022**, *10*, 43954–43963. [CrossRef]
35. Tian, Z.; Zhang, R.; Hou, X.; Liu, J.; Ren, K. FederBoost: Private federated learning for GBDT. *arXiv* **2020**, arXiv:2011.02796.
36. Qinbin, L.; Zeyi, W.; Bingsheng, H. Practical federated gradient boosting decision trees. *Proc. AAAI Conf. Artif. Intell.* **2020**, *34*, 4642–4649.
37. Gionis, A.; Indyk, P.; Motwani, R. Similarity search in high dimensions via hashing. *VLDB* **1999**, *99*, 518–529.
38. Wang, B.; Yu, S.; Lou, W.; Hou, Y.T. Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud. In Proceedings of the IEEE INFOCOM 2014-IEEE conference on computer communications, Toronto, ON, Canada, 27 April–2 May 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 2112–2120.
39. Qi, L.; Zhang, X.; Dou, W.; Ni, Q. A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data. *IEEE J. Sel. Areas Commun.* **2017**, *35*, 2616–2624. [CrossRef]
40. Datar, M.; Immorlica, N.; Indyk, P.; Mirrokni, V.S. Locality-sensitive hashing scheme based on p-stable distributions. In Proceedings of the twentieth annual symposium on Computational geometry, Brooklyn, NY, USA, 9–11 June 2004; pp. 253–262.
41. Du, W.; Han, Y.S.; Chen, S. Privacy-preserving multivariate statistical analysis: Linear regression and classification. In Proceedings of the 2004 SIAM international conference on data mining, Lake Buena Vista, FL, USA, 22–24 April 2004; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2004; pp. 222–233.
42. Yang, L.; Yan, K.; Chaoping, X.; Tianjian, C.; Qiang, Y. A secure federated transfer learning framework. *IEEE Intell. Syst.* **2020**, *35*, 70–82.
43. Yao, A.C. Protocols for secure computations. In Proceedings of the 23rd annual symposium on foundations of computer science (sfcs 1982), Chicago, IL, USA, 3–5 November 1982; IEEE: Piscataway, NJ, USA, 1982; pp. 160–164.
44. Kaggle. Credit Card Fraud Detection. Available online: https://www.kaggle.com/mlg-ulb/creditcardfraud (accessed on 2 September 2022.).
45. Yeh, I.C.; Lien, C. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Syst. Appl.* **2009**, *36*, 2473–2480. [CrossRef]
46. Kohavi, R.; Becker, B. Adult dataset. In *UCI Machine Learning Repository*; University of California, School of Information and Computer Science: Irvine, CA, USA, 1996.

47. Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated learning with non-iid data. *arXiv* **2018**, arXiv:1806.00582. [CrossRef]
48. Yurochkin, M.; Agarwal, M.; Ghosh, S.; Greenewald, K.; Hoang, N.; Khazaeni, Y. Bayesian nonparametric federated learning of neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; pp. 7252–7261.