Fed-EINI: An Efficient and Interpretable Inference Framework for Decision Tree Ensembles in Federated Learning

Xiaolin Chen [1 2] **Shuai Zhou** [3] **Kai Yang** [4] **Hao Fan** [4] **Zejin Feng** [4] **Zhong Chen** [4] **Yongji Wang** [* 1 2] **Hu Wang** [* 4]

Abstract

The increasing concerns about data privacy and security drive an emerging field of studying privacy-preserving machine learning from isolated data sources, i.e., *federated learning*. A class of federated learning, *vertical federated learning*, where different parties hold different features for common users, has a great potential of driving a more variety of business cooperation among enterprises in many fields. In machine learning, decision tree ensembles such as gradient boosting decision trees (GBDT) and random forest are widely applied powerful models with high interpretability and modeling efficiency. However, the interpretability is compromised in state-of-the-art vertical federated learning frameworks such as SecureBoost with anonymous features to avoid possible data breaches. To address this issue in the inference process, in this paper, we propose Fed-EINI protect data privacy and allow the disclosure of feature meaning by concealing decision paths with a communication-efficient secure computation method for inference outputs. The advantages of Fed-EINI will be demonstrated through both theoretical analysis and extensive numerical results.

1. Introduction

Data in the real world widely exist in isolated islands, which becomes a fundamental limiting factor for AI modeling and data analytics. For example, personal credit assessing in data-driven risk management usually uses three types of data: qualification data, credit data, consumption data, often held by different enterprises and institutions. The security and privacy of data are increasingly concerned in recent years. Typically, operators have enacted laws and regulations to protect the privacy of users' data, such as the General Data Protection Regulation (GDPR) (EU, 2016) by European Union and California Consumer Privacy Act by California, United States. Moreover, the data assets are highly valued, making data owners often reluctant to reveal their data to others. It drives the emergence of a novel field, termed as *federated learning* (Konečný et al., 2016; Yang et al., 2019b), studying privacy-preserving distributed machine learning from multiple data sources without sharing original data.

According to the structure of sample space and feature space across data sources, federated learning can be categorized into three classes (Yang et al., 2019b), i.e., horizontal federated learning, vertical federated learning, and federated transfer learning. Horizontal federated learning (Konečný et al., 2016; Hamer et al., 2020) refers to the scenarios where each participating party holds a subset of all data samples with common feature space. Vertical federated learning (Yang et al., 2019a; Gu et al., 2020) studies collaborative machine learning where different parties share the same sample space but differ in feature space. Federated transfer learning (Liu et al., 2020a) usually explores the area where different parties differ both in sample space and feature space. Among these structures, vertical federated learning is a promising approach to bridge the gap between isolated data providers for business cooperations and beyond the limits of locally available data to AI systems. There are a line of works studying the vertical federated learning of linear regression (Yang et al., 2019a), gradient boosting decision tree (GBDT) (Cheng et al., 2019), random forest (Liu et al., 2020b), kernel methods (Gu et al., 2020), etc.

Decision tree, especially decision tree ensemble models including random forest and GBDT (Zhou, 2009), are an important class of machine learning models due to their powerful generalization capability, high modeling efficiency, and better interpretability. Federated learning of decision tree models (Cheng et al., 2019; Feng et al., 2019; Liu et al., 2020b; Fang & Yang, 2008; Ong et al., 2020) has garnered numerous attention recently. The works of (Li

[*]Equal contribution [1]Cooperative Innovation Center, Institute of Software, Chinese Academy of Sciences, Beijing, China [2]University of Chinese Academy of Sciences, Institute of Software, China [3]University of Technology Sydney [4]JD Technology Group, Beijing, China. Correspondence to: Yongji Wang <ywang@itechs.iscas.ac.cn>, Hu Wang <wanghu5@jd.com>.

This work was presented at the International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML 2021 (FL-ICML'21).

et al., 2020; Ong et al., 2020) focused on improve the efficiency of GBDT model training of horizontal federated learning. SecureBoost (Cheng et al., 2019) first proposed a vertical federated learning framework to build GBDT model, whose security is guaranteed via encrypting the exchanged intermediate values with homomorphic encryption (HE). SecureGBM (Feng et al., 2019) proposed to extend the efficient GBDT framework LightGBM to vertical federated learning, and (Liu et al., 2020b) proposed a federated random forest framework.

Although vertical federated decision tree ensemble models have been widely applied in many fields, unfortunately, we observe that the inference of existing works such as SecureBoost severely compromises the interpretability of the model due to anonymous features. In these works (Cheng et al., 2019; Feng et al., 2019; Liu et al., 2020b), the commonly used multi-interactive (MI) inference procedure makes decision path of Host parties (i.e., data providers) known to the Guest party (i.e., the model user), which brings possible data breaches of Host with disclosed feature meaning. The federated learning models without explicit feature meanings are not interpretable in business applications. Both GDPR in EU and the Equal Credit Opportunity Act (ECOA) (Hsia, 1978) in US grant users the "right to explanation". A new financial industry standard (The People's Bank of China, 2021) recently released by the People's Bank of China specifies the interpretable requirement for features used in the AI algorithms for financial applications.

To address the issue, in this paper, we shall propose Fed-EINI, a secure and interpretable inference framework allowing to disclose feature meanings of Host parties to the Guest party. We observe that possible data breaches of existing works result from unprotected decision paths, which motivates the adoption of HE to compute the inference result with encrypted decision paths securely. To improve the communication efficiency and achieve low-latency inference, we compute, encode and encrypt candidate decision paths locally at each party to make the decision path indistinguishable by other parties. The inference result is then obtained via secure aggregation with only one round of encrypted information exchange. Both theoretical analysis and numerical results will be conducted to demonstrate the advantages of Fed-EINI.

2. Problem Statement

This section introduces the vertical federated decision tree ensemble models and states the federated inference problem. We then summarize the multi-interactive (MI) inference framework adopted in state-of-the-art works and discuss the challenging problem of lacking interpretability.

2.1. Vertical Federated Decision Tree Ensemble Models

Consider a vertical federated learning system with M parties denoted by $P_m, m \in \{1, 2, \cdots, M\}$. Denote the datasets distributed in M parties as $\{\mathbf{X}^m\}_{m=1}^M$. The local dataset of party m, i.e., $\mathbf{X}^m \in \mathbf{R}^{n \times d_m}$, consist of d_m features and n data samples. Therefore, the datasets of M parties can be looked as a vertically split on the large dataset $\mathbf{X} = [\mathbf{X}^1, \cdots, \mathbf{X}^M] \in \mathbf{R}^{n \times d}$ with disjoint feature space, in which $d = \sum_{m=1}^M d_m$. Vertical federated learning is closely concerned with business cooperation where each party is usually a different company, and the data labels are available by only a single party called the Guest party. Other participants without labels are called Host parties. The Guest party performs as the model user requiring more data features from Host parties to improve the performance of the AI model.

Here we take SecureBoost as a representative example to show how to build a federated decision tree ensemble model interactively. SecureBoost is a lossless extension of XGBoost (Chen & Guestrin, 2016), an efficient GBDT modeling framework, to vertical federated learning. We shall greedily build K regression trees $\{f_k\}_{k=1}^K$, where the t-th tree model f_t is built by minimizing the second-order approximation loss function

$$L^{(t)} \simeq \sum_{i=1}^{N} \Big(l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \Big) + \Omega(f_t), \quad (1)$$

where $l(y_i, \hat{y}_i)$ defines the loss between prediction value $\hat{y}_i$ and target label y_i, $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ is the gradient value, $h_i = \partial^2_{\hat{y}^{t-1}} l(y_i, \hat{y}^{t-1})$ is the Hessian value, $\hat{y}_i^{(t-1)} = \sum_{k=1}^{t-1} f_k(x_i)$ is the prediction result of previous $t-1$ trees, and $\Omega(f_t) = \gamma T + \frac{1}{2}\lambda ||\mathbf{w}||^2$ is the regularization term with $\gamma > 0, \lambda > 0$. T is the number of leaves and $\mathbf{w}$ is the vector of weights on leaves.

The key procedure is node generation by interactively computing the information gain for each feature and each possible splitting rule and computing the weights of leaf nodes. Both information gain and weights are functions of the aggregated gradients and Hessians, which can only be computed by the Guest party based on labels. Therefore, SecureBoost proposes that Guest encrypts all gradient g_i and Hessian h_i with additively homomorphic encryption such as Paillier (Paillier, 1999b) and transmits them to each Host party. Host parties compute the corresponding aggregation results of encrypted gradient and Hessian, and transmit them back to the Guest party. The Guest party can then compute the information gain for all possible splitting point, as well as the weights if the node is a leaf node.

With additively homomorphic encryption (Paillier, 1999b) denoted by $[\![\cdot]\!]$, we can efficiently compute the sum and scalar multiplication with ciphertexts, i.e., $[\![u]\!] + [\![v]\!] =$

$[\![u+v]\!]$ and $v \cdot [\![u]\!] = [\![v \cdot u]\!]$.

2.2. Federated Inference Problem

In this subsection, we provide the mathematical description of the inference problem for federated decision tree ensemble models. In the inference process, multiple parties have already collaboratively trained K decision tree models $\{f_k\}_{k=1}^K$. Let the input data sample for inference be $\mathbf{x} = (x_1, x_2, ..., x_d) = (\mathbf{x}^1, \mathbf{x}^2, ...\mathbf{x}^M) \in \mathbf{R}^d$, where $\mathbf{x}^m \in \mathbf{R}^{d_m}$ is the feature uniquely held by party m.

Given the trained decision tree ensemble model $\{f_k\}_{k=1}^K$ and input sample $\mathbf{x}$, the inference result can be expressed as

$$\hat{y} = \mathcal{G}(f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_K(\mathbf{x})), \tag{2}$$

where $\mathcal{G}$ is the ensemble strategy of prediction values of all decision tree models. In vertical federated learning, each participant holds a subset of all nodes in each f_k, and the leaf weights are typically available only by the Guest party. The inference results should be computed securely without sharing feature data among parties.

In the following, we present the expressions of two typical ensemble models, gradient boosting decision tree and random forest.

- **GBDT:** As presented in Section 2.1, the inference result of GBDT model is given by the sum of K trees' prediction values, i.e.,

$$\hat{y} = \sum_{k=1}^{K} f_k(\mathbf{x}) = \sum_{k=1}^{K} \sum_{j \in T_k} w_{(j,k)} \mathbf{I}(\mathbf{x} \in Leaf_j), \tag{3}$$

  where T_k is the index set of all leaf nodes for the k-th tree, $x \in Leaf_j$ represents that the prediction result of data x is the weight of j-th leaf node.

- **Random Forest:** Random forest is a well-known bagging tree method by independently learning many decision trees on a randomly sampled subset of data samples and features. The inference result is given by averaging the prediction values of K trees, i.e.,

$$\hat{y} = \frac{1}{K} \sum_{k=1}^{K} f_k(\mathbf{x}) = \frac{1}{K} \sum_{k=1}^{K} \sum_{j \in T_k} w_{(j,k)} \mathbf{I}(\mathbf{x} \in Leaf_j). \tag{4}$$

2.3. Multi-Interactive (MI) Inference

As shown in Figure 1, existing works (Cheng et al., 2019; Feng et al., 2019) have adopted a MI inference framework to compute the inference result. In this framework, the decision path of a decision tree model is determined by sequentially finding the decision on the current node and moving to the next node. Each decision is made by evoking

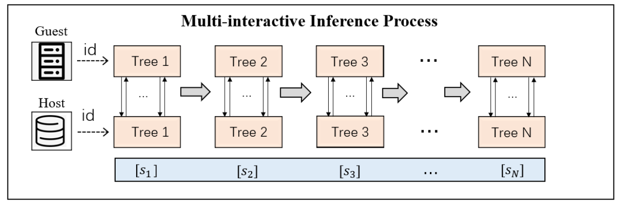


Figure 1. Structure of interactions for the multi-interactive inference framework.

the party which owns the corresponding feature and sending the computation result back to the Guest. The weight on the leaf node reached by the decision path is the decision tree's prediction value. The MI inference framework admits a serial structure of interactions shown in Figure 1, which means the inference is performed tree by tree. Note that the features for Host parties should be anonymous to the Guest, and the Guest party has only the index numbers (e.g., HF_1, HF_2) to avoid possible data breaches.

2.4. Problem Analysis

The pioneering MI inference framework enables a party to make predictions enhanced by external data sources. However, the features for Host parties should be anonymous to the Guest, and the Guest party has only the index numbers for the MI inference framework in existing works (Cheng et al., 2019; Feng et al., 2019). Disclosing the meanings of features of Host parties to Guest leads to possible data breaches. For example, if a feature in Host indicates whether a user "is in blacklist", we know that bad guys usually take only a small portion of all users. Therefore, the splitting direction with fewer samples indicates that the corresponding users are in the blacklist while others are not.

Model interpretability is urgently needed to realize the commercial value of federated learning in many fields such as financial risk management and smart medical diagnosis and treatment. Both GDPR (EU, 2016) enacted by EU and ECOA (Hsia, 1978) in US grants users the "right to explanation". Recently, the People's Bank of China released the financial industry standard titled *evaluation specification of AI algorithm in financial application* (The People's Bank of China, 2021), which specifies the interpretable requirement for features. The meanings of features are critical and mandatory for model interpretability. There are mainly two reasons as listed below.

- Firstly, the meanings of features are essential to prove the compliance of the model to regulators in financial businesses. For example, consumers should not be discriminated by their gender, age, race, and religion as required by ECOA in the US. As shown in Figure 4, a credit risk management model with the gender feature is not allowed, and gender information should be removed from the model. This problem cannot be addressed with the existing MI inference framework in SecureBoost because of the anonymous features.

- Moreover, industry practitioners need the meanings of features to judge the reasonability of AI models, especially for financial risk management. Since financial risks are usually impossible to be quantified explicitly, financial industry practitioners are often careful in making decisions to avoid potential loss. They usually evaluate AI models with their professional knowledge and experience before use, which requires the meanings of used features. For example, according to The expert experience, people with more stable consumption behavior should have higher credit scores. AI models behaving the opposite way are thought to be unreasonable and unusable for financial industry practitioners.

To provide interpretability for inference of vertical federated decision tree ensemble models, we propose to encrypt the decision path to guarantee the security of disclosing the meanings of Host parties' features. However, directly encrypt the decision path following the MI inference framework requires fully homomorphic encryption methods supporting both addition and multiplication, which brings unaffordable computation overhead in applications. To address these challenges, in this paper, we shall propose an efficient and interpretable inference framework named Fed-EINI, which adopts an additively homomorphic encryption method to secure decision paths.

3. Fed-EINI: Proposed Two-Stage Framework

This section presents Fed-EINI, a two-stage inference framework adopting additively homomorphic encryption to secure the decision path, including *parallel calculation* stage and *synchronization* stage.

3.1. Fed-EINI Algorithm

A tree model maps d features $(x_1, \cdots, x_d)$ of a sample $\mathbf{x}$ to the weight of a certain leaf node. A decision path is actually a combination of rules defined by the nodes on the path. Therefore, the output of a decision tree model $f(\mathbf{x})$ is the weight of the leaf node whose corresponding path from the root node to the leaf node holds conditions that can be simultaneously satisfied by the input data sample $\mathbf{x}$.

Denote the sub-model party m held for the k-th tree f_k as f_k^m. The sub-model is defined by a combination of rules according to the locally owned nodes of party m. Any party m can verify if a leaf node is a possible inference output based on its local input features $\mathbf{x}^m$ and locally owned nodes. We term the set of all possible leaf nodes as the *candidate set of party* m denoted by $f_k^m(\mathbf{x}^m)$. In this paper, we obtain the following key observation:

Key Observation: Given an input sample $\mathbf{x}$, the prediction value of a federated decision tree model f_k is the weight of the leaf node simultaneously in all candidate sets given by each party.

The prediction result of a single tree for $\mathbf{x}$ can thus be expressed as the intersection of results of the sub-models of the trees held by all parties, i.e.,

$$f_k(\mathbf{x}) = w_{(j,k)}, \text{ where } j \in \bigcap_{m=1}^{M} f_k^m(\mathbf{x}^m). \tag{5}$$

Proposition 1. *For the $Tree_k$, the weight $w'_{(j,k)}$ obtained by taking the intersection of results of the sub-models is unique and equal to the weight $w_{(j,k)}$ obtained by MI inference. That means,*

$$w_{(j,k)} = f_k(\mathbf{x}) = \bigcap_{m=1}^{M} f_k^m(\mathbf{x}^m) = w'_{(j,k)}. \tag{6}$$

Proof. See Appendix A. □

We can thus reformulate the inference result for federated decision tree ensemble model as

$$\hat{y} = \mathcal{G}\Big(\{f_k(\mathbf{x})\}_{k=1}^K\Big) = \mathcal{G}\Big(\{w_{(j,k)}\}_{k=1}^K\Big), j \in \bigcap_{m=1}^{M} f_k^m(\mathbf{x}^m). \tag{7}$$

Based on this key observation, we propose to compute the candidate sets of each party f_k^m locally and securely compute the inference results with homomorphic encryption (HE) to protect decision paths in each tree model. Since the aggregation functions $\mathcal{G}$ for both GBDT and random forest are linear, we adopt additively HE to improve the computation efficiency instead of fully HE. Therefore, the proposed inference framework is efficient and interpretable, allowing the disclosure of feature meanings of Host parties to Guest. It consists of two stages: parallel calculation stage and synchronization stage. To present the framework concisely, we take a two-party model case as a representative, in which case there are one Guest party and one Host party. It can be easily extended to a multi-party case, which will be demonstrated in Section 3.2.

3.1.1. Stage 1: Parallel Calculation

In this stage, each participant generates candidate sets of leaf nodes based on the split conditions of local nodes and local data features, rather than adopting a sequential communication structure as the MI inference approach. The candidate sets are then encoded as a decision vector for synchronization. Specifically, the encoded decision vector for each party is computed as follows.

- **Guest**: The encoded vector has as many entries as the number of leaf nodes for a tree model, where each entry is the corresponding weight if the leaf node is in the candidate set and 0 otherwise. That is, we compute an encoded decision vector $\mathbf{W}_k^{\text{Guest}} \in \mathbb{R}^{T_k}$ for the k-th decision tree in the Guest party as

$$\mathbf{W_k}^{\text{Guest}} = \left[s_{(j,k)} : s_{(j,k)} = \begin{cases} w_{(j,k)} \; if j \in f_k^{\text{Guest}}(\mathbf{x}^{\text{Guest}}) \\ 0 \quad otherwise \end{cases} \right], \quad (8)$$

  where T_k is the number of leaf nodes in the k-th decision tree model.

- **Host**: Similarly, the Host party encodes a candidate set as 1 if the leaf node is in the candidate set and 0 otherwise. The encoded decision vector $W_k^{\text{Host}} \in \mathbb{R}^{T_k}$ of the k-th decision tree for Host as

$$\mathbf{W_k}^{\text{Host}} = \left[s_{(j,k)} : s_{(j,k)} = \begin{cases} 1 \;\; if j \in f_k^{\text{Host}}(\mathbf{x}^{\text{Host}}) \\ 0 \;\; otherwise \end{cases} \right]. \quad (9)$$

3.1.2. Stage 2: Synchronization

In the synchronization stage, the Guest encrypted weight items as the encoding vector and sent it to the Host to compute the inference result, where the 0's in the encoding vector behave as confusion items to make the decision path indistinguishable for any party. Note that the public key is known to each party while the private key is held only by the Guest.

Specifically, the Guest party first encrypts and synchronizes all decision vectors $\{[\![\mathbf{W}_k^{\text{Guest}}]\!]\}_{k=1}^K$ to the Host party. Host merges each decision vector of the Guest party with each of its local decision vector and takes the sum of them to obtain the encrypted inference result, i.e.,

$$[\![\hat{y}]\!] = \sum_{k=1}^{K} [\![S_k]\!] = \sum_{k=1}^{K} \langle \mathbf{1}, [\![\mathbf{W}_k^{\text{Guest}}]\!] \circ \mathbf{W}_k^{\text{Host}} \rangle. \quad (10)$$

Here $\langle \cdot, \cdot \rangle$ denotes the inner product, and $\circ$ denotes the element-wise product of two vectors. Finally, the aggregation value $[\![\hat{y}]\!]$ is sent back to Guest to obtain the inference result through decryption.

The overall procedure of Fed-EINI algorithm is presented in Algorithm 1.

Here we present a toy example as illustrated in Figure 2. In the k-th tree model, circles or polygons represent nodes held by guest or host. In parallel calculation stage, for the sample $\mathbf{x}$, it does not satisfy the Guest's split condition of GF_1 and GF_2. The possible decision path for $\mathbf{x}$ at the Guest is the paths through weight $w_{(1,k)}$ and weight $w_{(2,k)}$.

$$\mathbf{W}_k^{\text{Guest}} = \left[w_{(1,k)}, w_{(2,k)}, 0, 0, 0, 0 \right]. \quad (11)$$

Algorithm 1 Fed-EINI: an efficient and interpretable inference framework.

Input : $x^{\text{Guest}}, x^{\text{Host}}, \{f_k^{\text{Guest}}\}_{k=1}^K, \{f_k^{\text{Host}}\}_{k=1}^K$
Output : Y
Set $[q] = 0, [S_k] = 0$
for $k = 1, \cdots, K$ **do**
  Stage 1: Parallel Calculation
    Guest&Host:Load parameters of f_k^{Guest} or f_k^{Host};
    Guest&Host:generate W_k^{Guest} or W_k^{Host} for x^{Guest} or x^{Host} during tree search according to equation(8)(9);

  Stage 2: Synchronization
    Guest: Encrypt and push $[\![\mathbf{W}_k^{\text{Guest}}]\!]$ to Host;
    Host: Pull $[\![\mathbf{W}_{k-1}]\!]$ from Guest;
    Host: $[\![S_k]\!] = \langle \mathbf{1}, [\![\mathbf{W}_k^{\text{Guest}}]\!] \circ \mathbf{W}_k^{\text{Host}} \rangle$;
end
Host: Push value $[\![\hat{y}]\!] = \sum_{k=1}^{K} [\![S_k]\!]$ to Guest;
Guest: Decrypt and get the prediction $\hat{y}$;

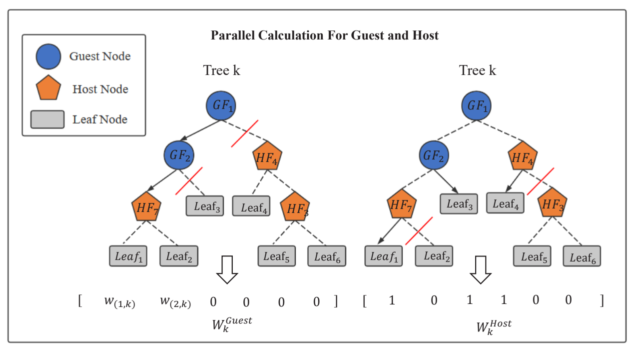


Figure 2. Parallel calculation at each party.

The instance x can't satisfy the Host's split condition of HF_4 and HF_7. The possible decision path for $\mathbf{x}$ at the Guest is the paths corresponding to Leaf_1 , Leaf_3 and Leaf_4 . These paths are encoded as 1, else 0.

$$\mathbf{W}_k^{\text{Host}} = \left[1, 0, 1, 1, 0, 0 \right]. \quad (12)$$

In synchronization stage, the Guest party encrypts and synchronizes all decision vectors $\{[\![\mathbf{W}_k^{\text{Guest}}]\!]\}_{k=1}^K$ to the Host party. Host merges each decision vectors as (10), and send $[\![\hat{y}]\!] = \sum_{k=1}^{K} [\![S_k]\!]$ to the Guest party for prediction.

3.2. Extension To Multi-party Scenarios

The proposed Fed-EINI framework can be easily extended to multi-party scenarios. As shown in Figure 3, the parallel calculation stage is the same, while in the synchronization stage, each Host party updates the encrypted decision vector generated by the Guest and transmits it to the next Host party. Specifically, the i-th Host party updates the encrypted

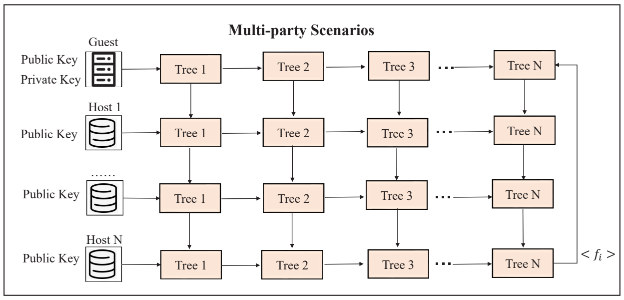


Figure 3. Multi-party security inference.

decision vector as

$$\mathbf{W}_k(i) = \mathbf{W}_k(i-1) \circ \mathbf{W}_k^{\text{Host}_i},\ \mathbf{W}_k(0) = \mathbf{W}_k^{\text{Guest}} \quad (13)$$

for $i = 0, \cdots, m-1$ with a Guest party (party 0) and $m-1$ Host parties. The encrypted inference result can be computed as

$$[\![\hat{y}]\!] = \sum_{k=1}^{K} [\![S_k]\!] = \sum_{k=1}^{K} \langle \mathbf{1}, [\![\mathbf{W}_k(m-1)]\!] \rangle. \quad (14)$$

The $(m-1)$-th Host party transmits $[\![\hat{y}]\!]$ to the Guest party to decrypt the inference result.

4. Analysis of Fed-EINI

In this section, we provide theoretical analysis on security and interpretability, as well as the efficiency of Fed-EINI.

4.1. Security and Interpretability

This subsection will demonstrate that Fed-EINI is secure and interpretable while disclosing the meaning of features to the Guest party under the semi-honest and non-collusive assumption. We will first show that the shared information in plaintext across parties will not reveal data privacy, followed by demonstrating that cracking the exchanged ciphertexts across parties is computationally infeasible.

Our method discloses the meaning of features and achieves the same security as the existing framework with the semi-honest assumption. We make a comprehensive comparison on the information available to each party between the proposed Fed-EINI and the multiple-interactive (MI) inference framework adopted in existing frameworks such as SecureBoost (Cheng et al., 2019), as illustrated in Table 1. Note that each party is honest-but-curious and the Guest does not collude with any Host given the semi-honest and non-collusive assumption. As shown in Table 1, we categorize the related information used in inference into two classes, i.e., model information and data information. Compared with the MI inference in SecureBoost, the proposed Fed-EINI allows the disclosure of the meanings of features held by passive parties to the Guest while hiding the splitting path information. In the inference process, the disclosure of the splitting path adopted in SecureBoost is privacy-sensitive if the Guest also has the availability to the feature meaning. To address this issue, SecureBoost has proposed to hide the meaning of features held by passive parties, and Guest only knows index numbers of features. Therefore, the Guest knows the split path of each sample but does not know the meaning of the features corresponding to the split path. In the proposed Fed-EINI approach, Guest knows the plain-text feature meaning of each splitting node without any information about the split directions based on other parties' data. We should notice that the splitting path in Fed-EINI is secured by Paillier cryptosystem (Paillier, 1999a), which is IND-CCA2 secure in the random oracle model.

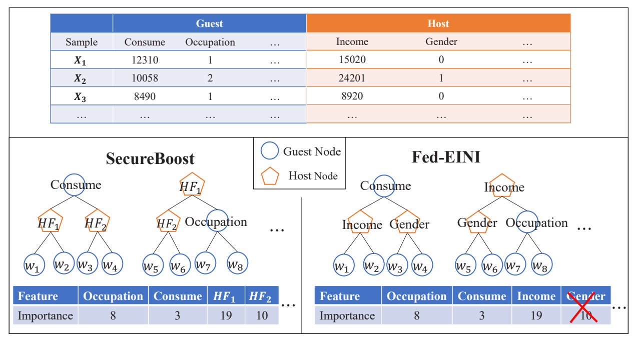


Figure 4. Comparison of model structure information and feature importance.

We provide an illustrative example in Figure 4 to show the differences between Fed-EINI and SecureBoost for GBDT model inference on available model structure information. It can be seen that Fed-EINI provides both the meanings of features and their importance, which is more interpretable and applicable. In real applications, features held by a party could be already known by other parties due to business dealings in the past. In this case, the approach of anonymizing features in MI inference can still lead to data breaches, while Fed-EINI provides better protection for data privacy. Therefore, Fed-EINI is a more secure and interpretable framework for business use.

4.2. Efficiency

Fed-EINI adopts efficient additively homomorphic encryption methods to secure decision paths. Nevertheless, the encryption operations and ciphertext exchanges among parties are still more costly in computation and communication than plaintext adopted in MI inference. We further adopt parallel calculation to improve the computation efficiency and reduce the number of interactions to one round to improve the communication efficiency in Fed-EINI.

Parallel Inference: For the MI inference framework, the

Table 1. Comparison of SecureBoost and Fed-EINI in information sharing

		SecureBoost		Fed-EINI	
		Guest	Host	Guest	Host
Model Information	Model structure	√	√	√	√
	Weights of leaf nodes	√	×	√	×
	Model parameters	√	√	√	√
	Splitting rules	Local nodes	Local nodes	Local nodes	Local nodes
Data Information	Local data	√	√	√	√
	Number of features	√	√	√	√
	Meaning of features	Local features	Local features	All features	Local features
	Decision path	Complete decision path	Decision path based on local nodes	Decision path based on local nodes	Decision path based on local nodes

split direction of a node is decided only after its parent node has been visited. With this sequential structure, each party has to wait for the completion of another party to perform local inference. Our proposed inference algorithm decouples the interaction process between multiple parties, and each party generates all the candidates in parallel based on its local splitting condition and local data.

One-Round Communication: In the MI inference, to predict x, it is necessary to interact at each node level to query whether the split condition is satisfied. When inferring the sample x in MI inference, communication must be performed at each level. The communication complexity of MI inference is $O(K * Depth)$. Since $f(x)$ can be expressed as the intersection of all participants' candidates $\{f_k^m(x^m)\}_{k=1}^K$, in our framework, the inference of each tree only needs to communicate once at last layer. The total time of communication is $O(K)$.

5. Experimental Results

In this section, we conduct numerical experiments with the proposed Fed-EINI and the MI inference framework (SecureBoost as representative) to show the accuracy and efficiency of Fed-EINI.

5.1. Experiments Setup and Metrics

Our experiments are conducted on three datasets, Credit1 (Give me some credit, 2011), Credit2 (UCI Machine Learning Repository, 2017), and JDT, to verify the performance of Fed-EINI on classification task. Details of dataset statistics are shown in Table 2. The summary of the three datasets is as follows:

(1) Credit 1: It's a credit scoring dataset used to predict whether a user would repay on time. It consists of 30000 data instances, and each instance has 25 attributes;

(2) Credit 2: It's also a credit scoring dataset to classify whether a user suffers from a financial problem. It consists of 150000 data instances, and each instance has 10 attributes;

(3) JDT: It's a non-public dataset of JD Technology used for credit scoring. It consists of 512082 data instances, and each instance has 113 attributes.

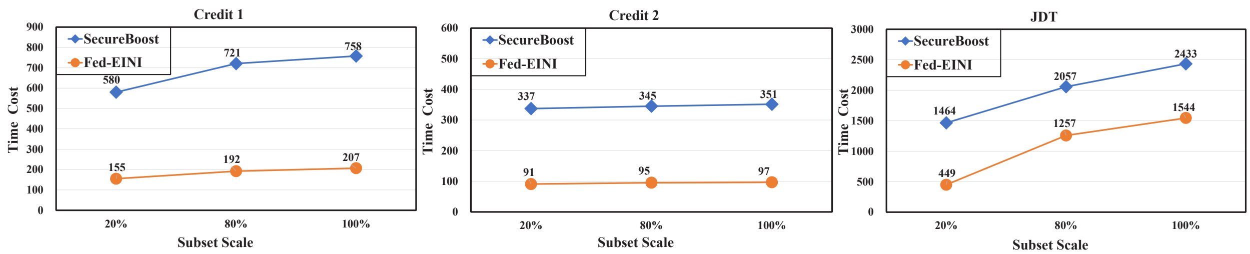
Table 2. Description of datasets

Datasets	Instances	Attributes	Data Size
Credit 1	30000	25	5.3M
Credit 2	150000	10	7.21M
JDT	512082	113	147M

To formulate the datasets for each party, we split each dataset into two parts vertically. Guest holds five features, and Host holds the remaining features. We randomly select 60% of the data as the training set, 20% as the validation set, and the remaining as the test set. To investigate the performance of our proposed method on different dataset scales, we also take 20%, 80%, and 100% of every dataset as subsets.

The Paillier encryption scheme is taken as our additively homomorphic scheme with a key size of 512 bits. All experiments are conducted on two machines with 128GB RAM and 32 CPU cores. We set the maximum number of modeling decision trees as 100 and the maximum tree depth as 4. The experimental party is divided into two sub-sections: accuracy and efficiency.

In this paper, we select AUC (area under the curve) and KS (maximum difference of TPR and FPR) to evaluate the accuracy of our framework. Besides, we use the time cost of the entire inference process to evaluate the efficiency of inference.



(a) (b) (c)

Figure 5. Efficiency comparisons between Fed-EINI and SecureBoost. a) Credit 1 dataset. b) Credit 2 dataset. c) JDT dataset.

Table 3. Accuracy comparisons between Fed-EINI and SecureBoost

		SecureBoost		Fed-EINI	
	Sampling Rate	AUC	KS	AUC	KS
Credit1	20%	0.880	60.3	0.880	60.3
	80%	0.853	55.0	0.853	55.0
	100%	0.855	54.9	0.855	54.9
Credit2	20%	0.773	43.0	0.773	43.0
	80%	0.771	41.0	0.771	41.0
	100%	0.771	40.9	0.771	40.9
JDT	20%	0.636	19.9	0.636	19.9
	80%	0.639	19.8	0.639	19.8
	100%	0.638	19.7	0.638	19.7

5.2. Accuracy Results

To prove the accuracy of Fed-EINI, we use the SecureBoost inference method and our improved inference method on the above datasets to conduct experiments and calculate the predictions' evaluation metrics based on the same training model. Table 3 provides the performance comparisons between Fed-EINI and SecureBoost measured by AUC and KS on the randomly sampled subset of each dataset. The metrics of our framework and the MI SecureBoost inference are completely identical to the 3rd decimal digit. Although we use different calculations, the goals of our inference and the original method are the same, so the difference in the results is very small. We can infer from that our inference framework is lossless compared with MI inference.

5.3. Efficiency Results

To show the training efficiency of Fed-EINI, we conduct inference experiments on the federated tree algorithm (GBDT). First, we use the same dataset for model training and then use the SecureBoost inference framework and Fed-EINI framework for inference. We count the time cost of the whole process of inference.

It can be seen from Figure 5, the efficiency of our proposed inference framework method exceeds the SecureBoost inference method. On the credit datasets (Credit1, Credit2), the inference time with our computing method only accounted for about 27% of the GBDT method. On the JDT data set, the average time consumed by our encoding method only accounts for 49% of the time consumed by GBDT, and in the worst case, this ratio is only 63.4%. We guess that this is due to the imbalance in the division of the JDT data set. Guest holds five features, while Host has 108 features. In the parallel calculation stage, the encoding time of the Guest is much shorter than that of the Host. The Host is always in a computing state, so the efficiency improvement is less than Credit datasets.

6. Conclusion

In this paper, we studied the inference problem for vertical federated decision tree ensemble models. To address the lack of interpretability, we proposed Fed-EINI to encrypt decision paths to make it secure to disclose feature meanings to the Guest party, namely the model user party. To reduce the high costs resulted from encryption operations and ciphertext exchanges, we proposed an efficient two-stage framework including parallel calculation stage and synchronization stage. It adopted efficient additively homomorphic encryption and parallel calculation to improve computation efficiency and required only one round of information exchange to improve communication efficiency. Since the decision path is invisible to any party, Fed-EINI provides a more secure and interpretable inference framework than existing works for vertical federated decision tree ensemble models. Experimental results demonstrated the accuracy and efficiency of Fed-EINI compared with state-of-the-art methods.

References

Regulation (EU) 2016/679 of the European Parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (General Data Protection Regulation). *OJ L 119*, pp. 1–88, 2016. https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=OJ:L:2016:119:FULL&from=EN.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 785–794, 2016.

Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., and Yang, Q. SecureBoost: A lossless federated learning framework. *arXiv preprint arXiv:1901.08755*, 2019.

Fang, W. and Yang, B. Privacy preserving decision tree learning over vertically partitioned data. In *2008 International Conference on Computer Science and Software Engineering*, volume 3, pp. 1049–1052. IEEE, 2008.

Feng, Z., Xiong, H., Song, C., Yang, S., Zhao, B., Wang, L., Chen, Z., Yang, S., Liu, L., and Huan, J. SecureGBM: Secure multi-party gradient boosting. In *2019 IEEE International Conference on Big Data*, pp. 1312–1321. IEEE, 2019.

Give me some credit. Give me some credit. https://www.kaggle.com/c/GiveMeSomeCredit/data, 2011.

Gu, B., Dang, Z., Li, X., and Huang, H. Federated doubly stochastic kernel learning for vertically partitioned data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2483–2493, 2020.

Hamer, J., Mohri, M., and Suresh, A. T. FedBoost: A communication-efficient algorithm for federated learning. In *International Conference on Machine Learning*, pp. 3973–3983. PMLR, 2020.

Hsia, D. C. Credit scoring and the equal credit opportunity act. *Hastings LJ*, 30:371, 1978.

Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., and Bacon, D. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Li, Q., Wen, Z., and He, B. Practical federated gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4642–4649, 2020.

Liu, Y., Kang, Y., Xing, C., Chen, T., and Yang, Q. A secure federated transfer learning framework. *IEEE Intelligent Systems*, 35(4):70–82, 2020a.

Liu, Y., Liu, Y., Liu, Z., Liang, Y., Meng, C., Zhang, J., and Zheng, Y. Federated forest. *IEEE Transactions on Big Data*, 2020b.

Ong, Y. J., Zhou, Y., Baracaldo, N., and Ludwig, H. Adaptive histogram-based gradient boosted trees for federated learning. *arXiv preprint arXiv:2012.06670*, 2020.

Paillier, P. Cryptosystems based on composite residuosity. *École Nationale Supérieure des Télécommunications*, 1999a.

Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pp. 223–238. Springer, 1999b.

The People's Bank of China. *Evaluation specification of artificial intelligence algorithm in financial application*. Financial Industry Standard, Beijing, China, 2021. https://www.cfstc.org/bzgk/gk/view/yulan.jsp?i_id=1912.

UCI Machine Learning Repository. default of credit card clients data set. https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients, 2017.

Yang, K., Fan, T., Chen, T., Shi, Y., and Yang, Q. A quasi-newton method based vertical federated learning framework for logistic regression. In *NeurIPS Workshop on Federated Learning for User Privacy and Data Confidentiality*, 2019a.

Yang, Q., Liu, Y., Chen, T., and Tong, Y. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):1–19, 2019b.

Zhou, Z.-H. Ensemble learning. In *Encyclopedia of Biometrics*, pp. 270–273. Springer, 2009.

A. Proof of the Proposition

Proof. Firstly, every leaf weight $w_{(j,k)}$ corresponds to a path on the k-th tree f_k. If a sample $\mathbf{x}$ satisfies the splitting condition of the path $w_{(j,k)}$, since $f_k = \bigcap_{m=1}^{M} f_k^m(\mathbf{x}^m)$, then the x satisfies the condition of the path $w_{(j,k)}$ on tree f_k^m. And $w_{(j,k)}$ will exist in all the sets $\{f_k^m(\mathbf{x}^m)\}_{m=1}^{M}$, then $f_k(\mathbf{x})$ belongs to the intersection of items $\bigcap_{m=1}^{M} f_k^m(\mathbf{x}^m)$. Secondly, if the intersection is not unique, there will be another weight $w_{(j',k)}$ ($j' \neq j$), and x satisfy all the splitting conditions of the search tree $\{f_k^m\}_{m=1}^{M}$ on path $w_{(j',k)}$. Then, x satisfy the splitting condition on path $w_{(j',k)}$. However, the left and right path of each node of the decision tree are mutually exclusive conditions, so there will not be a sample $\mathbf{x}$ that satisfies two different search paths on a tree at the same time. $\square$