JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 38, 57-82 (2022)
DOI: 10.6688/JISE.202201_38(1).0004

Game-based Theory Rational Delegation Learning Scheme

KANG XIANG[1,2], YOU-LIANG TIAN[1,2,+], SHENG GAO[3],
CHANG-GEN PENG[1,2] AND WEI-JIE TAN[1]
[1]*State Key Laboratory of Public Big Data*
College of Computer Science and Technology
[2]*Institute of Cryptography and Data Security*
Guizhou University
Guiyang, 550025 P.R. China
[3]*School of Information*
Central University of Finance and Economics
Beijing, 100081 P.R. China
E-mail: xiangkang5258@gmail.com; youliangtian@163.com[+]

Many enterprises or smart mobile devices collecting user data (*e.g.* smart-watches and wearable healthcare devices, *etc.*) are limited by their own computing power, so can't mine useful information from the data. To address this problem, this paper proposes a rational delegation machine learning pattern. In the proposed pattern, we firstly construct a reliable game model, and then build a formal model of delegation machine learning by delegation computation thought. Finally, we design a rational delegation learning scheme (RDLS) for decision tree model. The feasibility and reliability of the scheme are guaranteed by the incentive and constraint mechanism of game model. Moreover, we analyze the security and performance of the proposed scheme, the results show that the scheme reduces the client's computing costs and can not disclose any useful information. Last but not least, the experimental result demonstrates that the scheme can obtain a decision tree model with high accuracy in the case of ensuring the security of data.

Keywords: rational delegation learning, game theory, decision tree, machine learning, privacy protection

1. INTRODUCTION

With the rapid development of wireless mobile networks [1], machine learning [2] and other related technologies, a large number of favorable information in the smart mobile devices can be mined and utilized. Nevertheless, due to the complexity of machine learning technology and the high demand for data-processing, many enterprises, individuals or smart mobile devices are limited by their own computing power and can't mine useful information from the data, so they can only rely on the service provider which has the computing power to mine and learn by machine learning technology. Such as wearable healthcare devices collects user information and uploads it to the cloud through wireless mobile network for machine learning model training [3], so as to classify or forecast data. However, in the absence of feasible schemes and technical support, it is unsafe

Received September 30, 2020; revised February 18, 2021; accepted May 7, 2021.
Communicated by Changqiao Xu.
[+] Corresponding author.

for an enterprise to directly entrust the data set to the untrusted cloud service provider for processing.

All the aforementioned issues motivate the need to explore effective approaches for privacy-preserving machine learning training. Lindel and Pinkas initiated the study of privacy-preserving decision tree training [4]. Since then, many approaches have been proposed [5–9]. Hoogh *et al*. [10] proposes a privacy protection decision tree training scheme under the semi-honest model. Using Paillier cryptosystem [11] and Fairplay [12], Kikuchi [13] proposed a privacy-preserving decision tree training (PPDT) scheme for vertically partitioned datasets. However, this scheme can only be used for the boolean target class. In particular, Li *et al*. [14] proposed an outsourced privacy-preserving ID3 decision tree (OPPID3) algorithm over encrypted datasets for two-party. The solution suitable for data owners looking to outsource their data storage, *i.e*., data owners can outsource their encrypted data and mining tasks to a semi-trusted (*i.e*., curious-but-honest) cloud in a privacy-preserving manner. Two years later, they also proposed an outsourced privacy-preserving C4.5 decision tree (OPPC4.5) algorithm over horizontally and vertically partitioned dataset among multiple parties [15]. Nonetheless, in their schemes, the users should stay on-line during the training process. Roughly, these approaches mainly fall into three types, *i.e*., randomization-based obfuscation, differential privacy, and cryptography-based approaches. The randomization-based obfuscation and differential privacy are usually efficient but of low classification accuracy. Comparatively, the cryptography-based approaches could provide desirable privacy with high accuracy. However, the computing costs is so high that it cannot be applied to large data sets. In addition, in above researches, the server is usually assumed to be honest or semi-honest. But in reality, the server is usually rational, that is, the behavior of the server is driven by interests. If the server is assumed to be honest or semi-honest, the practical feasibility of the outsourced privacy-preserving decision tree training protocol will definitely be reduced.

In this paper, we are mainly interested in privacy protection in the process of data outsourcing and feasible outsourcing training schemes. It is of great theoretical significance and practical application value to design a new outsourcing service pattern that rational participants are introduced based on the traditional delegation computation [16] and machine learning, when the classification accuracy and user calculation cost need to be considered at the same time. We name this machine learning model outsourcing training as delegation learning pattern, as shown in the Fig. 1.

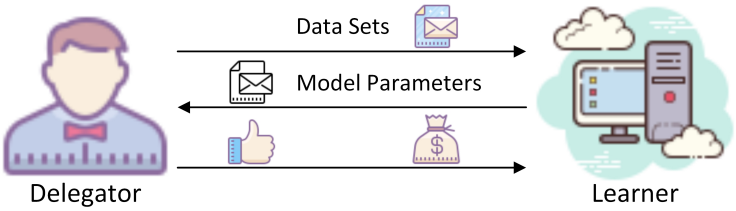


Fig. 1. The delegation learning pattern.

The delegation learning pattern raises an important security requirement issue between the delegator and the server. Because there is a lot of sensitive information in the data sent by the delegator to the service party, some malicious attackers may gather pri-

vacy information of delegator. Consequently, it is necessary to hide or delete sensitive information in the data set.

The design of the delegation learning scheme needs to consider the following factors:

I. The service party cannot access the real data. In other words, the service party should use the encrypted data to train the machine learning model, and no sensitive information can be obtained from the mined results. Due to the service party may gain the user's privacy from the real data, or even secretly retain and use the trained model, the more serious thing that the service party may resell the client's data set, so it will be insecure.

II. The method for removing sensitive information must be as simple as possible. And the encrypted data must meet the requirements of model training. Because the amount of data used for model training are usually large.

III. During the model training process, the service party should be prevented laziness or forge model parameters to deceive the delegator. And the final model obtained by the delegator must be correct and available. This is the basic purpose and meaning of delegation learning.

In brief, even if the data set sent by the delegator is encrypted, the service party must ensure that the model trained on the encrypted data set is correct and feasible. In other words, if the design of the delegation learning scheme meets the requirements of I, II and III, the client will be able to outsource the task of machine learning model training to the service providers.

1.1 Related Works

The related research mainly includes two aspects: delegation computation and machine learning. The delegation computation can be roughly divided into two categories: a complexity theory-based construction scheme and a cryptography-based construction scheme. The interactive proof method [17, 18] and homomorphic encryption technology [19–21] are commonly respectively used in the above two schemes. Regardless of the category, however, traditional delegation computation protocols usually assume that the participants are honest or malicious. Nevertheless, in practical applications, the participants are mostly rational, and each participant may be driven by interests to make dishonest behavior. Therefore, it has become current hotspot that use game theory [22] to study the rational delegation computation and the relationship between participants. Originally, Azar *et al*. [23] proposed a rational proof system according to the appropriate scoring rules, in which the participants are neither honest nor malicious, but rational. Subsequently, Azar *et al*. [24] construct anultra-efficient rational proof system by using the idea of utility gaps. In [25–27], the researchers also studied from a rational perspective. In addition, other works [28,29] studied the rational secret sharing and rational delegation computing technology. Inspired by [30,31], we also establish incentive mechanism in the proposed scheme.

The related research of privacy protection technology in machine learning mainly includes decision tree [32], K-means clustering [33, 34], support vector machine classification [35, 36], linear regression [37–39] and logistic regression [40]. Early privacy protection technologies can be divided into data-based perturbation methods and secure

multi-party computing-based methods. In terms of data perturbation, Agrawal *et al*. [41] proposed a privacy protection method which adds random noise to implement decision tree mining. Let the real data be X, randomly generate noise r with known distribution, and the disturbed data $Y = X + r$. The real data X is not public, and the users of the data get the distribution of the disturbed data Y and noise r. Since r is a random number, the user only knows the distribution of r without knowing its specific value, so the real data X cannot be obtained. However, this method of randomly adding noise is too simple. Kargupta *et al*. [42, 43] based on the random matrix theory, proposed a method for estimating real data from perturbed data. Subsequently, Weiping *et al*. [44] designed a privacy-preserving decision tree mining method for discrete attributes. They use a reversible transition probability matrix to perturb attributes. However, there is still a risk that private data will be leaked because there is still a considerable portion of real data that has not been disturbed. Bu *et al*. [45] presented a function-based perturbation method, which uses inverse function transformation to restore the decision tree T' on the perturbed data Y to the decision tree T on the real data X. In terms of secure multi-party computing, Mohassel *et al*. [46] proposed a new and efficient confidential machine learning protocol for linear regression, logistic regression and neural network training. This protocol belongs to two server model, in which the data owner allocates its private data to two non collusive servers, which use secure two-party computing to train various models of joint data. In [46], the researchers assume that the two servers are not collusive, however, the service provider is rational in reality. A year later, Mohassel *et al*. [47] designed and implemented a general machine learning framework for privacy protection, and used it to obtain a new solution of training linear regression, logical regression and neural network model. The framework includes three servers, in which data owners secretly share their data among the three servers. Ma *et al*. [48] proposed a new secure multi-party deep learning framework for cloud computing, which distributed a large number of training data to multiple participants, and allowed multi-party learning to generate the same neural network model based on the aggregate data set on the cloud server. Compared with [47, 48], our proposed scheme contains only one server, which not only avoids collusion between servers, but also reduces risk of privacy disclosure and extra cost caused by communication between servers.

To the best of our knowledge, in the existing schemes, users should communicate with the cloud server several times to get the final results. Therefore, these works cannot support off-line users.

1.2 Our Contributions

In this paper, we introduce game theory and rational participants into delegation learning for outsourcing model training to cloud service provider. Our main contributions are as follows:

1. We introduce rational participants into the delegation learning pattern. In other words, all participants are rational. In addition, based on the utility thought and the game theory, we construct the game model and incentive function of rational delegation learning scheme. The scheme achieves the utility balance of both parties by inspiring and restricting the participants, and can obtain a high accuracy model finally.
2. We derive the formal model of rational delegation learning based on the traditional

delegation computation, and design a rational delegation learning scheme (RDLS) for the decision tree model. This scheme not only ensures the security of user data but also has the fairness and result concealment characteristic.

3. According to the thought of One-hot codes, we design a text data and discrete data preprocessing method of outputting integer function-based perturbation (OIFP). In this method, we replace the real data and adopt the piecewise function-based perturbation (FP) method to encrypt the data set.

This article is organized as follows: Section 2 introduces the preliminary knowledge required in this article; Section 3 analyzes the delegation learning game model; Section 4 proposes the rational delegation learning model; Section 5 designs rational delegation learning scheme; Section 6 performs security analysis; Section 7 presents performance evaluation; Section 8 concludes this paper.

2. PRELIMINARIES

2.1 Game

The basic game representation consists of three elements: participant set PS, strategy space S and utility function u, namely $G = \{PS, S, u\}, S = \{S_1, S_2, ..., S_n\}, u = \{u_1, u_2, ..., u_n\}$. Utility function $u_i : S \rightarrow R(R\ represents\ real\ space)$ represents the ultimate benefit of the i-th participant in selecting different combinations of strategies. For detailed definitions, please refer to literature [22].

2.2 Statistical Machine Learning

Statistical machine learning (SML) is to build a probabilistic statistical model based on data to predict and analyze the data. The basic representation of statistical machine learning can be composed of three elements: model hypothesis space M , criterion set C, and algorithm set A, namely $SML = \{M, C, A\}$.

• $M = \{M_1, M_2, ..., M_n\}$. Before model training, its possible parameters are multiple or even infinite, so the possible models are multiple or even infinite. The set of these models forms the model hypothesis space.

• $C = \{C_1, C_2, ..., C_n\}$. The criterion set represents the method for selecting the model M_i with the best parameters from the hypothesis space.

• $A = \{A_1, A_2, ..., A_n\}$. The algorithm set represents the method of selecting the model M_i from the hypothesis space.

2.3 Decision Tree Model

In machine learning, the decision tree model is usually applied in classification and prediction. It represents a mapping relationship between object characteristics and label value. Each node in the tree represents a feature, and each bifurcation path represents a possible eigenvalue, and each leaf node represents the resulting value of the object represented by the path from the root node to the leaf node. The essential to construct decision tree is how to choose the optimal partition attribute. In general, as the partition process progressing, ideally, the samples contained in the branch nodes of the decision

tree belong to the same category as much as possible, in other words, the "purity" of the nodes become more and more higher. As a rule, "information entropy", "information gain" and "gini index" are the most frequently-used indicators for measuring the "purity" of a sample set. For a detailed description of the decision tree model, please refer to literature [49].

2.4 Monochromatic Value and Monochromatic Piece

Suppose there are n instances in data set D, and each instance has d attributes ($B = \{B_1, B_2, ..., B_d\}$) and classification label Y. We express the range of attribute B_i value as set $b_i (i = 1, 2, ..., d)$, and put all instances in ascending or descending order with the value of b_i. If the label of all instances with value v on attribute B_i are equal, then v is a monochromatic value [45]. Namely, $\forall c_1, c_2 \in D$, if $c_1.B_i = c_2.B_i = v$ and $c_1.Y = c_2.Y$, then v is a monochromatic value, where c is an instance. For example, in the following Table 1, there are 8 instances in total. We put all instances in ascending order with the value of age attribute. All values except 25 in the table are monochromatic values. If the age attribute values of all instances in the same data piece are monochromatic and the label values are equal, then this data block is called a monochromatic piece relative to age attribute. In Table 1, p_1 and p_3 are monochromatic pieces.

Table 1. The salary sample data.

Case :	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
$Age(B_1)$:	19	22	24	25	25	32	32	35
$Sex(B_2)$:	*w*	*m*	*m*	*w*	*w*	*m*	*w*	*m*
$Education(B_3)$:	*junior*	*bachelor*	*bachelor*	*master*	*bachelor*	*doctor*	*bachelor*	*master*
$Salary(Y)$:	*Low*	*Low*	*Low*	*High*	*Low*	*High*	*High*	*High*
Pieces :		p_1			p_2		p_3	

2.5 Piecewise Function Perturbation

According to Section 2.4, we randomly insert (w–1) breakpoints to divide the data that has been sorted according to the value of the attribute B_i into w pieces, namely $nb_i = \delta_1(nb_i) \cup \delta_2(nb_i) \cup \cdots \cup \delta_w(nb_i)$ and $\delta_r(nb_i) \cap \delta_k(nb_i) = \emptyset, r \neq k$, where nb_i represents the value sequence of n instances on the attribute B_i. Then w data pieces are respectively disturbed by w different functions, namely $nb_i' = \{f_1(\delta_1(nb_i)), f_2(\delta_2(nb_i)), ..., f_w(\delta_w(nb_i))\}$. Suppose that the data is arranged in ascending order. In order to satisfy the global monotonicity, if and only if, $1 < r < k < w, \forall v \in \delta_r(nb_i), \forall u \in \delta_k(nb_i)$, then $f_r(v) < f_k(u)$. According to [45], monochromatic pieces are suitable for any permutation function, but non-monochromatic pieces are only limited to monotonic functions. Here, we can define the function family suitable for monochromatic piece as F_{mon}, and the function family suitable for non-monochromatic piece as F_{nonmon}. Note that the F_{nonmon} is closed in combination, in other words, if $f, g \in F_{nonmon}$, the combination of function f and g is monotonic.

3. THE GAME-ANALYSIS OF RATIONAL DELEGATION LEARNING PATTERN

During the verification process of the machine learning model, the same model is tested by different test set of the same distribution, the final results are biased. Therefore, we introduce rational participants into the delegation learning pattern to ensure the reliability of the delegation learning process by the utility function. Assume that the participants include the delegator U and the learner L, and all of them are rational. The delegator outsources the task of model M training to the learner. The game model of rational delegation learning is defined as: $G = \{PS, S, u\}$, $PS = \{U, L\}$. The relevant definitions are as follows:

- $V(M)$ represents the worth of the model;
- $L(M)$ expresses the cost of the learner training model;
- $U(M)$ means the cost that the delegator needs to pay to the learner for outsourcing the training model;
- $T(D)$ represents the transmission cost that the delegator sends the data set and the learning task to the learner;
- $T(W)$ expresses the transmission cost of sending the parameter set W of the model to the delegator after the learner trained the model.

The above definition must meet the following conditions,

$$\begin{aligned} &V(M) > U(M) > L(M), \\ &U(M) > L(M) + T(W). \end{aligned} \tag{1}$$

Otherwise, the model is not necessary to delegate learning.

Since the participants are rational, they may behave maliciously for their own benefits. On the one hand, in general, the delegator doesn't spend transmission cost to maliciously deceive the learner. However, it is not excluded that the delegator deliberately transmits meaningless data to maliciously occupy the computing resources of the learner. On the other hand, in order to reduce calculation cost, the learner may choose to be lazy and finally transmit a wrong result to the delegator. Therefore, both the delegator and the learner have the same set of behavioral strategies $S : \{honest, malicious\}$.

During the delegation learning process, each participant will definitely act to maximize their own interests from the perspective of self-interest. In order to obtain a better model, the delegator can motivate the learner by setting the incentive function Q and the minimum required accuracy acc_u of the actual application of the model. For simplicity, suppose the delegator's incentive budget is E and the incentive function is a quadratic function:

$$Q(acc_T) = \theta(acc_T - acc_u)^2 \times E, \tag{2}$$

where θ and acc_T respectively represents the incentive coefficient and the actual accuracy of the model, and the incentive coefficient satisfies $\theta(1 - acc_u)^2 = 1$. For example, the minimum accuracy requirement set by the delegator is 80%, then $\theta = 25$. If and only if $acc_T - acc_u > 0$, the delegator will pay the incentive amount of money $Q(acc_T)$. In short, only when the higher the accuracy of the model trained by the learner, the more the

learner will gain income. Of course, the reward, transmission cost and remuneration must meet the following conditions,

$$V(M) > Q(acc_T) + T(D) + U(M). \tag{3}$$

Otherwise, the model will lose the value of outsourcing training. The final utility functions of the delegator and the learner are denoted by u_1 and u_2, as shown in Table 2.

Table 2. The behavior utility matrix of participants.

		Delegator U	
		Honest	Malicious
Learner L	Honest	$u_1 = V(M) - U(M) - T(D) - Q(acc_T)$ $u_2 = U(M) + Q(acc_T) - L(M) - T(W)$	$u_1 = -T(D)$ $u_2 = -L(M) - T(W)$
	Malicious	$u_1 = -U(M) - T(D)$ $u_2 = U(M) - T(W)$	$u_1 = -T(D)$ $u_2 = -T(W)$

In Table 2 above, because the delegator can not accurately verify the results using the test set, the learner may not seriously complete the task of the delegator. Therefore, when the learner makes malicious behavior (such as, learner lies that the accuracy of the model has reached acc_u), we do not consider the cost $L(M)$ of model trained by the learner. According to the above behavior utility matrix, it can be seen that the learner will prefer to choose a malicious strategy in order to maximize his own interests. This shows that the feasibility of delegation learning cannot be guaranteed. Therefore, we add a trusted third-party platform P to the delegation learning pattern. It is to ensure that the interests of the honest party are not harmed, so as to play a role in constraining both sides. The third-party is similar to the government notarization agency, which has no model training ability but only verification ability. Suppose that the cost of introducing a third-party and the fee of third-party verification is P_R and P_V respectively. The P_R should be paid jointly by the delegator and the learner. If a party engages in malicious deception, the verification fee P_V will be paid by it. We provide that before the start of the delegation learning process, the delegator and the learner respectively submits the deposit c and l on platform P. The deposits must meet the following conditions,

$$c \geq U(M) + P_V, \, l > U(M) + T(D) + P_V + P_R/2. \tag{4}$$

We will explain the deposit setting in detail in Section 6. Of course, Eqs. (1) and (3) should be changed to

$$\begin{aligned} &U(M) > L(M) + T(W) + P_R/2, \\ &V(M) > Q(acc_T) + T(D) + U(M) + P_R/2. \end{aligned} \tag{5}$$

The final utility game tree of participants is shown in the Fig. 2.

According to the above analysis, only when both the delegator and the learner choose the honest strategy, the interests of both parties can reach the optimal, and this strategy combination is Nash equilibrium of this game model.

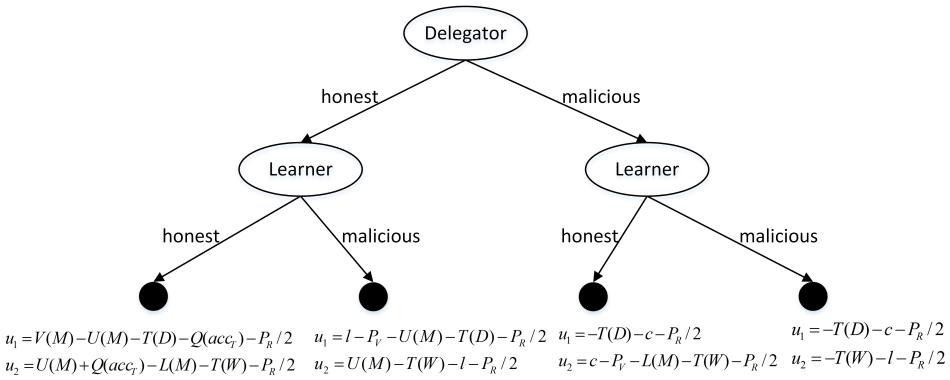


Fig. 2. The final utility game tree of participants.

4. THE PROPOSED RATIONAL DELEGATION LEARNING MODEL

4.1 System Model

Delegation learning is different from the traditional delegation computation. The traditional delegation computation means that the delegator delegates the computing party to compute the value of a specific function $f(x)$ that it has no ability to compute. And finally the computing party will return a verifiable result to the delegator. However, delegation learning is a new pattern based on the traditional delegation computation and statistical machine learning technology to fit the relationship model between features and targets. The learner uses the data set given by the delegator to train a model that can deal with classification or prediction problems. The basic form of the delegation learning pattern can be expressed as:

$$TrS + VS + acc_u \xrightarrow{C+A} M(W), \tag{6}$$

where TrS represents the training set, VS represents the validation set, and W represents the parameter set of model M. According to Section 2.2, the C and the A respectively represents the criteria and algorithms of the parameter optimal model selected from the hypothesis space. The reason why we introduce rational participants and construct a game model in the delegation learning pattern is precisely to avoid the lazy and lying of the learner. At the same time, it also motivates the learner to choose the most suitable criterion and algorithm from the criterion set and the algorithm set for training.

According to the analysis in Section 3, we can see that all participants in the delegation learning pattern are untrustworthy. Therefore, the trusted third-party P is introduced into the delegation learning pattern. The rational delegation learning structure diagram is shown in the Fig. 3.

In this structure, participants include the delegator U and the learner L, and all of them are rational. After the two parties reach the entrustment agreement, they will respectively deliver the corresponding deposits c and l to the third-party. Firstly, the delegator respectively publishes acc_u and D' to the learner and the third-party P, where D' is the

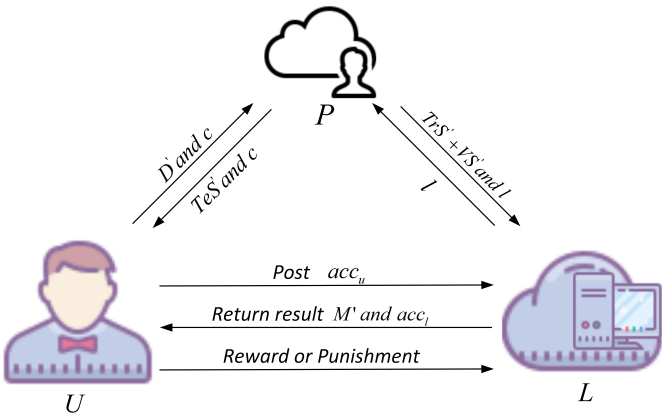


Fig. 3. The rational delegation learning structure diagram.

encrypted data set. The third-party generates the training set, validation set and test set from the data set D' and respectively distributes it to the delegator and learner. Secondly, the learner uses the training set to train and obtain the model M', and returns the model and acc_l to the delegator, where acc_l represents the accuracy that the learner promises the model can achieve. Finally, the delegator pays the remuneration or punishes the learner according to the results of the model verification. If both parties adopt a honest strategy, they can withdraw their deposits from the third-party after the entrustment process ends, otherwise the third-party will help honest party to confiscate the deposit of the dishonest party for maintaining the rights of the honest party. The specific steps of the rational delegation learning scheme are described in detail in the rest of this paper.

4.2 Formal Model

Based on the delegation computation thought, we derive a formal model of rational delegation learning for decision tree model. Due to the each node selection in the construction of the decision tree model is not affected by the true value of each feature attribute, but depends on the information gain or gain rate of each feature attribute. Therefore, the encrypted data set in the construction of the decision tree model has no effect on the model training.

For the convenience of description, we assume that the delegator has a discrete or text data set D, which contains n instances, as shown in the Table 3. Each instance has attribute set $B = \{B_1, B_2, ..., B_d\}$ and a label Y. The label value set and attribute B_i value set is represented by y and b_i respectively, *i.e.* $y = \{y_1, y_2, ..., y_s\}$, $b_i = \{b_{i,1}, b_{i,2}, ..., b_{i,s}\}$. Of course, the number of each attribute value is not necessarily equal. It is temporarily set as s for the convenience of description. In addition $d, n \in N^*$, $i = \{1, 2, ..., d\}$, $j = \{1, 2, ..., s\}$.

The formal model of the rational delegation learning scheme (RDLS) for decision tree consists of the following three parts, namely $RDLS = \{OIFP, Learning, Veri\}$.

I. The function-based perturbation encryption method $OIFP$ includes the following three parts:

Table 3. The delegator's data set.

$B:$	B_1	B_2	B_3	$\cdots$	B_d	Y
Case 1 :	$b_{1,j}$	$b_{2,j}$	$b_{3,j}$	$\cdots$	$b_{d,j}$	y_j
Case 2 :	$b_{1,j}$	$b_{2,j}$	$b_{3,j}$	$\cdots$	$b_{d,j}$	y_j
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Case n :	$b_{1,j}$	$b_{2,j}$	$b_{3,j}$	$\cdots$	$b_{d,j}$	y_j

1. $\textit{One-hotGen}(tab \xrightarrow{F_{(0,1)}} tab_b)$: First of all, the delegator needs to extract the attribute set, the attribute value set, label and label value set from data set D, and represents them with B, b_i, Y and y respectively. Then, the values in b_i and y are arranged in disorder to generate the basic information table tab, in which the data are replaced by $b_{i,j}$ and y_j. Finally, tab is transformed by mapping function $F_{(0,1)}$ to obtain the basic information table tab_b after boolean transformation.

2. $\textit{IntegerGen}(tab_b \xrightarrow{O_N} D_N)$: The data in tab_b is converted to integers and sorted in descending or ascending order. Then the actual value is replaced by the virtual value to obtain the complete virtual data set D_N, namely $D_N \leftarrow tab_N \leftarrow O_N(tab_b)$. Where O_N is the virtual conversion function.

3. $FP(D_N \xrightarrow{\vec{f}_{B_i}} D')$: Firstly, the delegator randomly inserts (w-1) breakpoints to divide the sequence nb_i of attribute B_i value into w pieces, and secretly sets the function families F_{mon} and F_{nonmon}. Next, the delegator randomly select the conversion functions from the function families F_{mon} and F_{nonmon}, and combine them into conversion function vector $\vec{f}_{B_i} = [f_1, f_2, ..., f_w]$, $\vec{f}_{B_i} \in \{F_{mon}, F_{nonmon}\}$. Finally, the delegator can obtain the converted sequence nb_i' of attribute B_i value, namely $nb_i' = \vec{f}_{B_i}(nb_i)$. After the d-round conversion, the disturbed data set D' can be obtained. Note that in the transformation, the delegator needs to secret preserve the breakpoint position and inverse function vector $\vec{f}_{B_i}^{-1} = [f_1^{-1}, f_2^{-1}, ..., f_w^{-1}]$ of each attribute sequence, so as to restore the real decision tree T from the decision tree T' based on D'.

II. $\textit{Learning}((D', acc_u) \xrightarrow{C,A} (T', acc_l))$: According to the data set D' given by the delegator and the minimum accuracy requirement, the learner selects the appropriate criterion and algorithm for mining, and returns the final decision tree T' and accuracy acc_l.
III. $\textit{Veri}(acc_T \xrightarrow{?} acc_l)$: The delegator verifies whether the accuracy of the decision tree T approaches to acc_l. Because there is a deviation when the test set verifies the accuracy of the model, the result $acc_T = acc_l$ is usually not obtained. Of course, the degree of "approaching" can be determined through negotiation between both parties. For example, when the condition $acc_u < acc_l - \lambda < acc_T < acc_l + \lambda$ is established, the delegator considers that the result returned by the learner is acceptable, otherwise the delegator can initiate a verification request to a third-party, where λ represents the average deviation of the model accuracy test.

5. THE PROPOSED RATIONAL DELEGATION LEARNING SCHEME

According to the above game-analysis and formal description of delegation learning, we design the following rational delegation learning scheme for decision tree model. Assuming that the learner can accept the request of the delegator and reach a delegation learning agreement. The proposed scheme is as follows:

5.1 Initialization Phase

I. Data processing

The delegator cleans the data set, only retains the necessary information to construct the decision tree, and then disturbs the data set according to the $OIFP$ method. Regarding the setting of functions $F_{(0,1)}$ and O_N, as long as the function satisfies the data conversion relationship, the delegator can set it arbitrarily. For the convenience of explanation, we take the data in Table 1 as an example and design the following simple conversion function:

$$\begin{aligned} &\text{Boolean Conversion: } F_{(0,1)}(b_{i,j}) = 0^{(j-1)}10^{(|b_i|-j)},\ F_{(0,1)}(y_i) = 0^{(i-1)}10^{(|y|-i)},\\ &\text{Integer Conversion: } O_N(b_{i,j}) = (|b_i|-j)^k,\ O_N(y_i) = |y|-i. \end{aligned} \tag{7}$$

The symbol $(*)$ in the upper right corner of the above function expresses that the quantity of 0 is $*$, the $|*|$ expresses the quantity of elements in the set $*$. The k expresses the index, $k \in N^*$, $k > 1$. The value of k is determined by the delegator, and even the delegator can take different values of k for each attribute. In the rest of this article, we will further explain.

After the function conversion, the following basic information bijection table tab_N can be obtained, as shown in the Table 4.

Table 4. The basic information bijection table tab_N.

B:	B_1						B_2		B_3				Y	
A-v:	24	35	25	22	19	32	w	m	ju	ma	ba	do	L	H
$b_{i,j}$:	$b_{1,1}$	$b_{1,2}$	$b_{1,3}$	$b_{1,4}$	$b_{1,5}$	$b_{1,6}$	$b_{2,1}$	$b_{2,2}$	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$	$b_{3,4}$	y_1	y_2
O-G:	10^5	0^110^4	0^210^3	0^310^2	0^410^1	0^51	10^1	0^11	10^3	0^110^2	0^210^1	0^31	10^1	0^11
I-G:	5^k	4^k	3^k	2^k	1	0	1	0	3^k	2^k	1	0	1	0

In Table 4, the first row represents the substitute value of attribute name, the second row is the actual value of each attribute, the third row is the substitute value of each attribute value, the fourth row and the last row are the converted virtual values. It can be seen that all the data in the original data, including text data, are converted into non-negative integers, which not only avoids the leakage of real data, but also makes it easier to perform the next step function perturbation operation. In addition, the data in the original data set D can be replaced by the data in tab_N to obtain a virtual data set D_N, as shown in the Table 5.

Table 5. The virtual data set D_N.

Case :	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
B_1 :	1	2^k	5^k	3^k	3^k	0	0	4^k
B_2 :	1	0	0	1	1	0	1	0
B_3 :	3^k	1	1	2^k	1	0	1	2^k
Y :	1	1	1	0	1	0	0	0

According to the descriptions in Section 2.4 and Section 2.5, we might as well arrange the attribute values of B_1 in descending order, and randomly select two breakpoints to divide the data into three pieces. As shown in the Table 6.

Table 6. The monochrome pieces of attribute B_1.

Case :	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8
B_1 :	5^k	4^k	3^k	3^k	2^k	1	0	0
Y :	1	0	0	1	1	1	0	0
Pieces :	p_1				p_2		p_3	

From Table 6, it is can be observed that the p_2 and p_3 data pieces are monochromatic piece, and p_1 is a non-monochromatic piece. Firstly, we need to randomly select the conversion functions from the function families F_{mon} and F_{nonmon}, and combine them into conversion function vector $\vec{f}_{B_1} = [f_1, f_2, f_3]$, $f_2, f_3 \in F_{mon}$, $f_1 \in F_{nonmon}$. Then, three data pieces are respectively disturbed by three different functions, namely $nb'_1 = \{f_1(\delta_1(nb_1)), f_2(\delta_2(nb_1)), f_3(\delta_3(nb_1))\}$. It is worth noting that the global monotonic invariant characteristic must be satisfied when performing piecewise function perturbation, namely $\forall u \in \{f_1(5^k), f_1(4^k), f_1(3^k)\}$, $\forall v \in \{f_2(2^k), f_2(1)\}$, must be $u > v > f_3(0)$. The rest may be deduced by analogy, the final perturbed data set D' can be obtained after the function vectors $\vec{f}_{B_2}$ and $\vec{f}_{B_3}$ respectively convert the values of B_2 and B_3.

II. Creating test set, validation set and training set

Firstly, the delegator sends the data set D' to the third-party P. Then the third-party randomly selects 20% and 10% of the data from the D' as the test set TeS' and the validation set VS', and all the remaining data as the training set TrS', namely $D' = TeS' + VS' + TrS'$. We assume that there are m instances in the training set and validation set, namely $TrS' + VS' = \{(X_1, Y_1), (X_2, Y_2), ..., (X_m, Y_m)\}$, where X and Y represent all attribute values and tag value of each instance respectively. It is worth noting that the reason why the training set and the test set are extracted by the third-party from the data set D' is to ensure that the data when building the model and testing the model have the same distribution. At the same time, it is avoided to use different distribution data to test model by delegator in testing phase.

III. Setting the time node and acc_u

The delegator publishes the minimum requirement for the accuracy acc_u of the model, and negotiates with the learner to decide that the learner must return to the decision tree T' and its promised accuracy acc_l within a limited time t. If the delegator verifies that the result is correct, the $U(M)$ and $Q(acc_T)$ fee must be paid to the learner

within a limited time t'.

IV. Submitting deposit and sending data set

After the delegator and the learner reach the delegation agreement, they respectively submit the deposit c and l to the trusted third-party. Subsequent, the third-party P respectively sends $TrS' + VS'$ and TeS' to the learner and the delegator.

5.2 Learning Phase

Firstly, the learner selects the algorithm A_i from the algorithm space A and learns the data features in the TrS' to obtain the model, namely $TrS' \xrightarrow{A_i} T''$. Secondly, iteratively optimize the model using different criterion C_i to make the model optimal, namely $T'' \xrightarrow{C_i, VS'} (T', acc_l)$. Finally, the learner returns the optimal model T' and acc_l to the delegator.

5.3 Verification and Payment Phase

This phase is discussed into two situations:

- The first case: the learner does not return the result within the agreed time t;
- The second case: the learner returns the result within the agreed time t.

In the first case, it shows that the learner does not honesty implement according to the delegation learning agreement. At this time, the delegator can contact P to recover his deposit c and confiscate the deposit l of the learner.

In the second case, the delegator decrypts T' after receiving T' and acc_l by using its own secretly saved inverse function vector $\vec{f}_{B_i}^{-1}$, bijection table tab_N and breakpoint location of each attribute to obtain the real decision tree model T. Then the delegator verifies accuracy of T. The specific steps are as follows:

I. The delegator uses $\vec{f}_{B_i}^{-1}$, the breakpoint locations of each attribute and the bijection table tab_N to restore the test set TeS;

II. The delegator tests T by the TeS to get the accuracy acc of the model.

$$acc(T, TeS) = \frac{1}{n-m} \sum_{i=1}^{n-m} \mathrm{I}(T(X_i) = Y_i), \tag{8}$$

where $\mathrm{I}(*)$ represents the indication function and takes a value of 1 or 0 when $(*)$ is true or false.

III. Of course, the delegator can evenly divide the TeS into h parts ($TeS = \{TeS_1, TeS_2, ..., TeS_h\}$) to test the model and use the final average value $\overline{acc}$ as the accuracy acc_T of the model. The value of h can generally define a minimum value. For example, $h \geq 5$.

$$\begin{aligned} acc_T &= \overline{acc} = \frac{1}{h} \sum_{j=1}^{h} acc_j(T, TeS_j), \\ \lambda &= \frac{1}{h} \sum_{j=1}^{h} |acc_j - \overline{acc}|. \end{aligned} \tag{9}$$

IV. The delegator announce the result of comparing acc_T with acc_l and discusses the following situations:

i. When $acc_u \leq acc_l - \lambda \leq acc_T$, the delegator must pay remuneration and reward within the time of t', otherwise the learner can make a request to the third-party P to confiscate the delegator's deposit.
ii. When $acc_u \leq acc_T < acc_l - \lambda$ or $acc_T < acc_u$, if the learner questions the test result announced by the delegator, it can initiate a verification request to a third-party P.
The third-party verification process is as following:
1. The third-party obtains T' and acc_l from the learner.
2. The third-party uses the same method as the delegator to test the model with the encrypted TeS'.
The verification results include the following three situations:
(1) When the result shows $acc_u \leq acc_T < acc_l - \lambda$, it means that the delegator does not deceive the learner during the test process, so the verification fee of the third-party should be paid by the learner. But the delegator must pay the remuneration and reward to the learner within a limited time according to the actual accuracy acc_T of the model, otherwise the third-party helps the learner to confiscate the delegator's deposit.
(2) However, when the result shows $acc_u \leq acc_l - \lambda \leq acc_T$, it means that the delegator deceives the learner during the test process, so the verification fee of the third-party should be paid by the delegator. Not only that, the learner can request the third-party to confiscate the delegator's deposit.
(3) When the result shows $acc_T < acc_u$, it means that the learner deceives the delegator during the learning process, so the verification fee of the third-party should be paid by the learner. At the same time, the delegator can request the third-party to help him confiscate the deposit of the learner.

6. SECURITY ANALYSIS

In this section, we analyze the security of the rational delegation learning scheme from the characteristics of fairness and data concealment.

Theorem 1: If the deposits of both sides respectively meet the conditions $c \geq U(M) + P_V$ and $l > U(M) + T(D) + P_V + P_R/2$, the rational delegation learning scheme is fair.
Proof: In the proposed scheme, each participant may choose to make dishonest behavior in order to maximize their own interests. In order to ensure the fairness of the scheme, at the beginning of the scheme, the delegator U and the learner L respectively submit deposit c and l to the trusted third-party platform P. The P_V in the deposit of all parties is used to pay verification fee to third-party.

Suppose 1: The strategies chosen by the delegator and the learner are {*malicious*, *honesty*}. According to the game-analysis in Section 3, the final utility of the delegator and the learner is

$$u_1 = -c - T(D) - P_R/2, \quad u_2 = c - P_V - L(M) - T(W) - P_R/2. \tag{10}$$

In order to protect the interests of the learner from damage, there must be $u_2 > 0$. Let's recall Eq. (5) in Section 3, so there must be $c - P_V \geq U(M)$.

Suppose 2: The strategies chosen by the delegator and the learner are {*honesty*,

malicious}, the final utility of the delegator and the learner is

$$u_1 = l - P_V - U(M) - T(D) - P_R/2, \quad u_2 = U(M) - T(W) - l - P_R/2. \tag{11}$$

Similarly, in order to protect the interests of the delegator from damage, there must be $u_1 > 0$. Therefore, there must be $l - P_V > U(M) + T(D) + P_R/2$.

In summary, when the deposits of the delegator and the learner meet the theorem 1 requirements, the interests of the honest party can not be harmed. In other words, any party in the scheme will be severely punished as long as he chooses a malicious strategy, so the proposed rational delegation learning scheme is fair.

Theorem 2: If the value width of the data piece $\delta_r(nb_i), r \in [1,w]$ of each attribute is large enough, the data set D_N of the delegator is hidden. That is to say, the rational delegation learning scheme is safe.

Proof. In [45], the author has demonstrated data safety of the monochromatic pieces and decision tree T'. However, concerning the defense sorting attack of non-monochrome pieces, we do other improvements on the basis of it. To be more precise, we made a simple virtualization replacement of the original data before the perturbation of the piece-wise function, namely $One\text{-}hotGen(tab) \xrightarrow{F_{(0,1)}} (tab_b)$ and $IntegerGen(tab_b) \xrightarrow{O_N} (D_N)$. We make the value width of the data piece controllable, and also reduce the correlation between attributes, which means that the leakage risk of related information between attributes is further reduced. For a better explanation, we first give some simple definitions:

• $g_r : \delta'_r(nb_i) \to \delta_r(nb_i)$ represents the attacker's cracking function.

• $[\min^k, \max^k] \in \delta_r(nb_i)$, the $\max^k - \min^k$ represents the value width of the r-th piece of the attribute B_i.

• If $|g_r(v') - f_r^{-1}(v')| \le \rho$, then we think the attacker cracked a numerical point $R_\rho = [v-\rho, v+\rho]$, where $v = f_r^{-1}(v')$, ρ represents the distance between the attacker's guess value and the real value.

In terms of defense against sorting attacks, as a matter of fact, when $\delta'_r(nb_i)$ contains some discontinuous values, the attacker can only crack the value $v' \in \delta'_r(nb_i)$ to a limited width $R_g = [v_1, v_2] \in \delta_r(nb_i)$. Assume that there are respectively m values and n values in front of and behind the value v' in piece p'_r, the width that the attacker can attack is $R_g = [\min^k + m, \max^k - n]$. The probability that the value v' is cracked can be defined as

$$P_{v'}(|g_r(v') - f_r^{-1}(v')| \le \rho) = \frac{|R_g \cap R_\rho|}{|R_g|}. \tag{12}$$

According to the above analysis, it is clear that when k is larger, the range of R_g is wider and the probability $P_{v'}$ is smaller. If the value of k is large enough, the probability $P_{v'}$ is negligible, so the data set D_N of the delegator is safe. Of course, the delegator can determine the value of k according to the characteristics of each data piece p_r. When the number of discrete values in the piece is large, k can take a larger value to make the width of the R_g sufficiently large. However, assume that although an expert-level attacker cracks and obtains a complete data set D_N, it will not help the attacker to obtain the original data set D. For example, in Table 6 , knowing rows 2 and 3 can not help the hacker to crack actual value of row 2 in Table 1.

Regarding the leakage of association information between attributes, we take age attribute and label as example, and compare the data in Table 1 and Table 6. The Pearson correlation coefficient values of the age attribute and the label respectively in these two tables are 0.8114 and 0.2917. This shows that the age attribute in the original data Table 1 has a strong correlation with the label. Of course, we can also see from the original data that older people generally have higher salaries. However, after we converted the original correlation in the data was destroyed. Therefore, the leakage risk of relevant information between data is also reduced.

To sum up, the data is safe in the proposed rational delegation learning scheme.

Theorem 3: The proposed rational delegation learning scheme does not disclose any information about the final verification results.
Proof: On the one hand, according to the third-party verification process in Section 5.3, it can be seen that malicious participants not only need to pay verification fee but also its deposit is confiscated. Consequently, from a rational point of view, rational participants will not risk losing their deposits to choose malicious strategies. In other words, the third-party verification function will not be called in the end.

On the other hand, in the model verification process of this scheme, there is no need for any interaction between the delegator and the learner, and the model verification is completely completed by the delegator independently. Therefore, the risk of being attacked is reduced. Although the attacker intercepted the model T' from the results returned by the learning party, according to theorem 2 above, the attacker cannot obtain any useful information from the model.

In summary, the proposed scheme do not disclose real and useful information.

7. PERFORMANCE EVALUATION

In order to evaluate the performance of the proposed RDLS, we used a 110MB car insurance prediction claim data set[1] as the training set to conduct experimental tests on the encryption efficiency of the proposed scheme, the time cost of model outsourcing training and the final accuracy of the model. A total of three Linux devices are used, one of which plays the role of user U, and the other two play the roles of server L and trusted third party P respectively. The test environment is as follows:
I. Hardware Environment
(1) CPU: Intel(R) Core(TM) i5-7500 3.4GH; (2) RAM: 8GB; (3) SSD: 256GB; (4) Bandwidth: 20Mbps
II. Software Environment
(1) OS: Linux(Ubuntu19.10); (2) Database software: MySQL8.0.18; (3) IDE: PyCharm; (4) Programming language: python3

First of all, in terms of data encryption efficiency, we conducted a comparative experimental test for the encryption method OIFP designed by us and the existing work [5, 9, 14, 15, 45]. The outsourcing protocols proposed in work [5, 9, 14, 15] use Distributed Two Trapdoors, Circuit Private, Li's symmetric and BCP Homomorphic Encryption (HE)

[1]Data Set: https://www.kaggle.com/priyasd/portoseguro?select=train.csv

to encrypt the private data respectively. The OIFP designed in this paper and the encryption method FP proposed in work [45] are based on piecewise function perturbation. In this experiment, we used python to establish a homomorphic cryptographic systems with a security parameter ε of 1024 for each of the above homomorphic encryption algorithms, and use the car insurance prediction claim data set to test the encryption efficiency, as shown in Fig. 4.

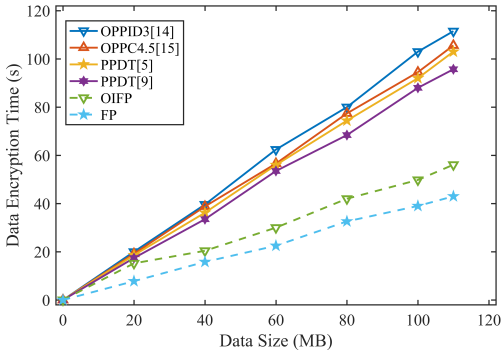


Fig. 4. Encryption time costs.

Although the homomorphic encryption algorithm can provide high-intensity data privacy, it can be seen from the Fig. 4 that the method based on piecewise function perturbation in terms of encryption time costs is much lower than that of the homomorphic encryption algorithm. However, because the method FP based on piecewise function perturbation directly inserts breakpoints on the private data set and uses the inverse function perturbation, it is not enough to provide the required data privacy and will affect the accuracy of the final decision tree model. For this reason, the OIFP method we designed has been improved and enhanced on the basis of the FP method. That is, before the private data is inserted into the breakpoint and the inverse function is used to perturb, OIFP pre-encrypts the private data set and generates a virtual data set, and uses the virtual data set instead of the private data set to perform the perturbation operation. Although OIFP's preprocessing of private data sets increases the time costs on data encryption, it also reduces the risk of leakage of correlations between data attributes and does not affect the accuracy of the final decision tree model. Therefore, we believe that such improvements and enhancements are worthwhile. Although the time costs of data encryption is increased compared with FP, the encryption efficiency is still much higher than that of homomorphic encryption algorithms, and it can provide the required data privacy.

Secondly, in terms of model outsourcing training time and communication costs, we also compared the proposed RDLS with the privacy protection outsourcing protocols and schemes proposed by work [5, 9, 14, 15]. Among them, the work [15] proposed the privacy protection outsourcing decision tree C4.5 protocol, and other work proposed the privacy protection outsourcing decision tree ID3 protocol. In addition, the outsourcing protocol proposed by work [14, 15] is oriented to multi-user and dual-server scenarios. To this end, we simulated two users on the device playing the user role and evenly divided the training set into two parts, simulated dual servers on the device playing the server

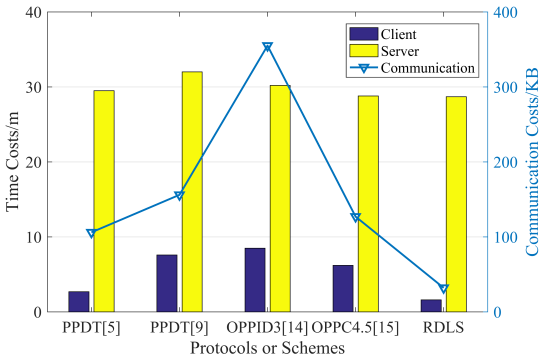


Fig. 5. The model training time and communication costs.

role, and performed outsourcing ID3 and C4.5 experiment for different protocols. As shown in Fig. 5. We use the time spend on the client and server to represent the computing costs of the client and server during the entire process of model outsourcing training (where the computing costs of the client includes the costs of assisting server calculation, encrypting the private data set and decrypting the final result), and use the communication volume (in order to more directly reflect the interaction degree of the collaborative computing between the client and the server in the model outsourcing training process, so the communication volume of encrypted data set is not included.) between the client and the server to represent the intermediate computing communication costs of the entire outsourcing protocol. Because in the privacy protection decision tree outsourcing training protocol proposed by Akavia *et al*. [9] and Li *et al*. [14,15], the client needs to coordinate with the server to calculate, that is, in the protocol, the server only undertakes part of the calculation tasks, some critical and sensitive computing tasks are still borne by the client, so the computing costs of the client is relatively high. The same reason causes frequent interactions between the client and the server or between the server and the server, so the communication costs is relatively high. But it is worth noted that the privacy protection decision tree outsourcing training protocol proposed by Liu *et al*. [5] is similar in the form of privacy data outsourcing to our proposed RDLS, that is, private data sets are encrypted and outsourced to the server at one-time. In the model outsourcing training process, the client and the server do not need to calculate collaboratively, so the computing costs of the client is lower. However, compared with our proposed scheme, Liu *et al*. [5] use the Distributed Two Trapdoors Homomorphic Encryption system to encrypt private data sets, so the computing costs of the client is higher. At the same time, they use dual-server coordination to perform model training on the encrypted data set, which leads to frequent interactions between the servers. In contrast, in our proposed solution, the client and the server only need to deposit and withdraw their respective deposits and transport the final result. Therefore, the communication costs is lower. It can be seen from Fig. 5 that our proposed RDLS significantly reduces the client's computing costs and the communication costs during the protocol execution.

In addition, according to the incentive function Q in Section 3, we analyze the relationship between the reward obtained by the learner and the accuracy of the model.

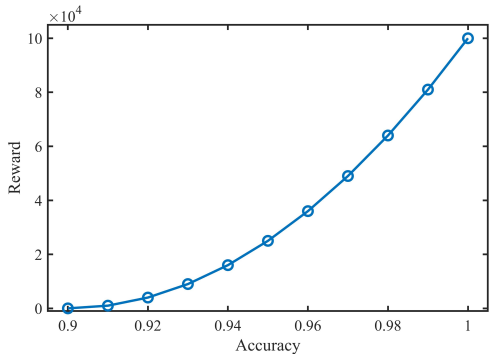


Fig. 6. The relationship between reward and accuracy.

Generally speaking, during model training, the higher the accuracy of the model is, the more difficult it is to further improve the accuracy of the model. Here, it is assumed that the minimum accuracy requirement set by the delegator is 90%, and the reward amount of E is 100,000 yuan, *i.e.*

$$\begin{aligned} &\theta(1-0.9)^2 = 1 \;\Rightarrow\; \theta = 100, \\ &Q(acc_T) = 100(acc_T - 0.9)^2 \times 100000, \; (acc_T > 0.9), \end{aligned} \tag{13}$$

then, we can get the relationship between reward and model accuracy, as shown in the Fig. 6. It can be seen that when the learner further improves the accuracy of the model, the reward given by the delegator is more abundant. This shows that the incentive function we set is reasonable, that is to say, the incentive function has an incentive effect on the learner. Subsequently, in terms of model accuracy, we also conducted a comparative test,

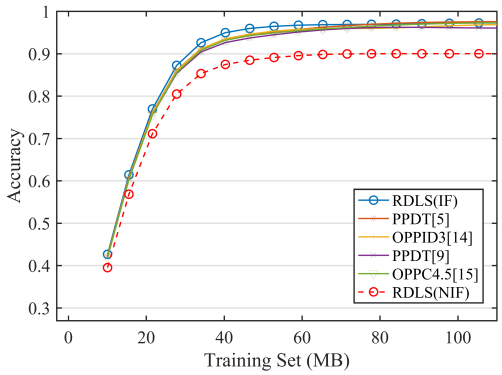


Fig. 7. The decision tree model accuracy.

as shown in Fig. 7. In the protocol proposed in the work [5,9,14,15], the server is usually assumed to be honest or semi-honest, that is, the server is assumed to try its best to train the model. But in reality, the server is usually rational. In order to make the proposed

scheme more realistic, we introduce rational participants into the proposed scheme, that is, the behavior of the server for model training is driven by interests, so we stress the case of setting the incentive function (IF) and without the incentive function (NIF). It can be seen from the Fig. 7 that when the incentive function is not set, the learner improves the accuracy of the model to 90% and no longer trains. However, when the incentive function is set, the learner tries his best to improve the accuracy of the model in order to obtain more benefits. Compared with the above work, when setting the incentive function, the scheme we proposed is not only more practical and feasible, but also a high-accuracy decision tree model can be obtained in the end.

Finally, we also conducted a more detailed comparative analysis from other aspects, as shown in Table 7. Because in the outsourcing protocol proposed by Akavia *et al*. [9] and Li *et al*. [14, 15], the client needs to assist the server to calculate, so off-line users are not supported, in addition, the communication complexity and client complexity are generally higher. That is to say, the computing costs on the client side is still relatively large. The outsourcing protocol proposed by Liu *et al*. [5] supports off-line users, but it adopts a dual-server pattern, and the server needs to have prior knowledge such as data set attribute names and attribute value information, which may lead to the decision tree model be inferred by the server. In addition, the trusted third party P plays an important role in the our proposed scheme. In order to restrain rational servers, the trusted third party adopts the method of confiscating the deposit of malicious servers to protect the interests of honest users. Therefore, we stress that the trusted third party in the RDLS scheme is essential.

Table 7. Protocols comparison summary.

Protocols	Client Complexity	Server Complexity	Communication Rounds	Server Has Prior Knowledge	Support off-line	Need Trusted Third Party
PPDT[5]	$O(n\|B\|^{1})$	$O(\overline{\|b\|}^{2}\|B\|log_2\|B\|)$	$\|y\|^{3}\overline{\|b\|}+\|B\|$	✓	✓	✓
PPDT[9]	$O(\|B\|\|y\|s^{4}t^{5})$	$O(\|B\|\|y\|stn)$	d^{6}	×	×	×
OPPID3[14]	$O((n)ln(n))$	$O((\|B\|n)ln(\|B\|n))$	$\|B\|n$	×	×	×
OPPC4.5[15]	$O(\|B\|\varepsilon)$	$O(\|B\|log_2\|B\|)$	$2\|B\|$	✓	×	×
RDLS	$O(\|B\|\overline{\|b\|})$	$O(\|B\|log_2\|B\|)$	7	×	✓	✓

[1] $|B|$ represents the number of attributes.
[2] $\overline{|b|}$ represents the average number of attribute values quantity.
[3] $|y|$ represents the number of label values.
[4] s represents the threshold of the decision tree node.
[5] t represents the number of decision tree nodes.
[6] d represents the depth of the decision tree.

In summary, the rational delegation learning scheme we proposed can reduce the client's computing costs and the communication costs during the protocol execution process while providing sufficient security, and the client can finally obtain a high-accuracy decision tree model. It is particularly noteworthy that we have introduced rational participants to make the proposed scheme more practical.

8. CONCLUSIONS

This paper proposes a rational delegation learning pattern and designs an outsourcing scheme for decision tree model, which reduces the leakage risk of delegator's data, the client's computing costs and the communication costs during the protocol execution. We establish the incentive function in learning process, which makes the rational learner to try his best to improve the model accuracy. Therefore, the feasibility of the rational delegation learning scheme is improved. Finally, we evaluate the proposed scheme, the results show that the delegator can get a high-accuracy model.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China under Grant Nos. 61662009 and 61772008; Science and Technology Major Support Program of Guizhou Province under Grant No.20183001; Key Program of the National Natural Science Union Foundation of China under Grant No.U1836205; Science and Technology Program of Guizhou Province under Grant No.[2019]1098; Project of High-level Innovative Talents of Guizhou Province under Grant No. [2020]6008.

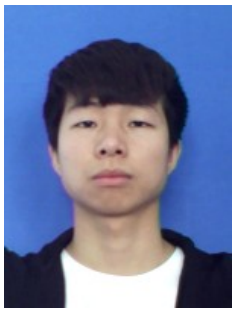
REFERENCES

1. X. Wang, Y. Zhang, V. C. Leung, N. Guizani, and T. Jiang, "D2d big data: Content deliveries over wireless device-to-device sharing in large-scale mobile networks," *IEEE Wireless Communications*, Vol. 25, 2018, pp. 32-38.
2. M. Somvanshi, P. Chavan, S. Tambade, and S. Shinde, "A review of machine learning techniques using decision tree and support vector machine," in *Proceedings of International Conference on Computing Communication Control and Automation*, 2016, pp. 1-7.
3. M. Chen, Y. Ma, Y. Li, D. Wu, Y. Zhang, and C.-H. Youn, "Wearable 2.0: Enabling human-cloud integration in next generation healthcare systems," *IEEE Communications Magazine*, Vol. 55, 2017, pp. 54-61.
4. Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Proceedings of Annual International Cryptology Conference*, 2000, pp. 36-54.
5. L. Liu, R. Chen, X. Liu, J. Su, and L. Qiao, "Towards practical privacy-preserving decision tree training and evaluation in the cloud," *IEEE Transactions on Information Forensics and Security*, Vol. 15, 2020, pp. 2914-2929.
6. S. Truex, L. Liu, M. E. Gursoy, and L. Yu, "Privacy-preserving inductive learning with decision trees," in *Proceedings of IEEE International Congress on Big Data*, 2017, pp. 57-64.
7. D. J. Wu, T. Feng, M. Naehrig, and K. Lauter, "Privately evaluating decision trees and random forests," *Proceedings on Privacy Enhancing Technologies*, Vol. 2016, 2016, pp. 335-355.

8. C. Wang, A. Wang, J. Xu, Q. Wang, and F. Zhou, "Outsourced privacy-preserving decision tree classification service over encrypted data," *Journal of Information Security and Applications*, Vol. 53, 2020, pp. 102517-102529.
9. A. Akavia, M. Leibovich, Y. S. Resheff, R. Ron, M. Shahar, and M. Vald, "Privacy-preserving decision tree training and prediction against malicious server," *IACR Cryptol. ePrint Arch.*, Vol. 2019, 2019, pp. 1282-1315.
10. S. de Hoogh, B. Schoenmakers, P. Chen, and H. op den Akker, "Practical secure decision tree learning in a teletreatment application," in *Proceedings of International Conference on Financial Cryptography and Data Security*, 2014, pp. 179-194.
11. P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of International Conference on Theory and Applications of Cryptographic Techniques*, 1999, pp. 223-238.
12. D. Malkhi, N. Nisan, B. Pinkas, Y. Sella *et al.*, "Fairplay-secure two-party computation system." in *Proceedings of USENIX Security Symposium*, 2004, pp. 287-302.
13. H. Kikuchi, K. Itoh, M. Ushida, H. Tsuda, and Y. Yamaoka, "Privacy-preserving decision tree learning with boolean target class," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. 98, 2015, pp. 2291-2300.
14. Y. Li, Z. L. Jiang, X. Wang, S.-M. Yiu, and P. Zhang, "Outsourcing privacy preserving id3 decision tree algorithm over encrypted data-sets for two-parties," in *Proceedings of IEEE Trustcom/BigDataSE/ICESS*, 2017, pp. 1070-1075.
15. Y. Li, Z. L. Jiang, L. Yao, X. Wang, S.-M. Yiu, and Z. Huang, "Outsourced privacy-preserving c4. 5 decision tree algorithm over horizontally and vertically partitioned dataset among multiple parties," *Cluster Computing*, Vol. 22, 2019, pp. 1581-1593.
16. S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, "Delegating computation: interactive proofs for muggles," *Journal of the ACM*, Vol. 62, 2015, pp. 1-64.
17. S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, Vol. 18, 1989, pp. 186-208.
18. G. N. Rothblum, S. Vadhan, and A. Wigderson, "Interactive proofs of proximity: delegating computation in sublinear time," in *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, 2013, pp. 793-802.
19. T. S. Fun and A. Samsudin, "A survey of homomorphic encryption for outsourced big data computation," *KSII Transactions on Internet and Information Systems*, Vol. 10, 2016, pp. 3826-3851.
20. R. Gennaro and D. Wichs, "Fully homomorphic message authenticators," in *Proceedings of International Conference on Theory and Application of Cryptology and Information Security*, 2013, pp. 301-320.
21. R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Proceedings of Annual Cryptology Conference*, 2010, pp. 465-482.
22. J. R. Marden and J. S. Shamma, "Game theory and control," *Annual Review of Control, Robotics, and Autonomous Systems*, Vol. 1, 2018, pp. 105-134.
23. P. D. Azar and S. Micali, "Rational proofs," in *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, 2012, pp. 1017-1028.
24. P. D. Azar and S. Micali, "Super-efficient rational proofs," in *Proceedings of the 14th ACM Conference on Electronic Commerce*, 2013, pp. 29-30.

25. Y. Tian, J. Guo, Y. Wu, and H. Lin, "Towards attack and defense views of rational delegation of computation," *IEEE Access*, Vol. 7, 2019, pp. 44037-44049.
26. S. Guo, P. Hubáček, A. Rosen, and M. Vald, "Rational arguments: single round delegation with sublinear verification," in *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science*, 2014, pp. 523-540.
27. Y. Tian, C. Peng, D. Lin, J. Ma, Q. Jiang, and W. Ji, "Bayesian mechanism for rational secret sharing scheme," *Science China Information Sciences*, Vol. 58, 2015, pp. 1-13.
28. D. Zhang, Y. Tian, C. Yue, and M. Fan, "Cooperate delegation of computation for rational party using zero-determinant strategy approach," *IEEE Access*, Vol. 8, 2020, pp. 27734-27743.
29. Y.-L. Tian, J.-F. Ma, C.-G. Peng, and W.-J. Ji, "Game-theoretic analysis for the secret sharing scheme," *Dianzi Xuebao(Acta Electronica Sinica)*, Vol. 39, 2011, pp. 2790-2795.
30. Q. Li and Y. Tian, "Rational delegation computing using information theory and game theory approach," in *Proceedings of International Conference on Multimedia Modeling*, 2020, pp. 669-680.
31. C. Dong, Y. Wang, A. Aldweesh, P. McCorry, and A. van Moorsel, "Betrayal, distrust, and rationality: Smart counter-collusion contracts for verifiable cloud computing," in *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 211-227.
32. F. Khodaparast, M. Sheikhalishahi, H. Haghighi, and F. Martinelli, "Privacy preserving random decision tree classification over horizontally and vertically partitioned data," in *Proceedings of IEEE 16th International Conference on Dependable, Autonomic and Secure Computing*, 2018, pp. 600-607.
33. Z. Gheid and Y. Challal, "Efficient and privacy-preserving k-means clustering for big data mining," in *Proceedings of IEEE Trustcom/BigDataSE/ISPA*, 2016, pp. 791-798.
34. V. Baby and N. S. Chandra, "Distributed threshold k-means clustering for privacy preserving data mining," in *Proceedings of International Conference on Advances in Computing, Communications and Informatics*, 2016, pp. 2286-2289.
35. L. Sun, W.-S. Mu, B. Qi, and Z.-J. Zhou, "A new privacy-preserving proximal support vector machine for classification of vertically partitioned data," *International Journal of Machine Learning and Cybernetics*, Vol. 6, 2015, pp. 109-118.
36. M. Z. Omer, H. Gao, and F. Sayed, "Privacy preserving in distributed svm data mining on vertical partitioned data," in *Proceedings of the 3rd IEEE International Conference on Soft Computing and Machine Intelligence*, 2016, pp. 84-89.
37. M. D. Cock, R. Dowsley, A. C. Nascimento, and S. C. Newman, "Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data," in *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security*, 2015, pp. 3-14.
38. A. Gascón, P. Schoppmann, B. Balle, M. Raykova, J. Doerner, S. Zahur, and D. Evans, "Privacy-preserving distributed linear regression on high-dimensional data," *Proceedings on Privacy Enhancing Technologies*, Vol. 2017, 2017, pp. 345-364.
39. G. Qiu, X. Gui, and Y. Zhao, "Privacy-preserving linear regression on distributed data by homomorphic encryption and data masking," *IEEE Access*, Vol. 8, 2020, pp. 107601-107613.

40. C. Bonte and F. Vercauteren, "Privacy-preserving logistic regression training," *BMC Medical Genomics*, Vol. 11, 2018, pp. 13-21.
41. R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, 2000, pp. 439-450.
42. H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *Proceedings of the 3rd IEEE International Conference on Data Mining*, 2003, pp. 99-106.
43. H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "Random-data perturbation techniques and privacy-preserving data mining," *Knowledge and Information Systems*, Vol. 7, 2005, pp. 387-414.
44. G. Weiping, W. Wei, Z. Haofeng, and S. Baile, "Privacy preserving classification mining," *Journal of Computer Research and Development*, Vol. 43, 2006, pp. 39-45.
45. S. Bu, L. V. Lakshmanan, R. T. Ng, and G. Ramesh, "Preservation of patterns and input-output privacy," in *Proceedings of IEEE 23rd International Conference on Data Engineering*, 2007, pp. 696-705.
46. P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *Proceedings of IEEE Symposium on Security and Privacy*, 2017, pp. 19-38.
47. P. Mohassel and P. Rindal, "Aby3: A mixed protocol framework for machine learning," in *Proceedings of ACM SIGSAC Conference on Computer and Communications Security*, 2018, pp. 35-52.
48. X. Ma, F. Zhang, X. Chen, and J. Shen, "Privacy preserving multi-party computation delegation for deep learning in cloud computing," *Information Sciences*, Vol. 459, 2018, pp. 103-116.
49. B. Hssina, A. Merbouha, H. Ezzikouri, and M. Erritali, "A comparative study of decision tree id3 and c4. 5," *International Journal of Advanced Computer Science and Applications*, Vol. 4, 2014, pp. 13-19.



Kang Xiang received his B.S. degree in Computer Science and Technology from Hubei Normal University in 2018. He is currently pursuing the M.S. degree at the College of Computer Science and Technology, Guizhou University, China. He research interests include delegation computing, big data security and privacy protection.

You-Liang Tian received the B.S. and M.S. degrees in Mathematics from Guizhou Universuty, Guiyang, China, in 2004 and 2009 respectively, and the Ph.D. degree in Cryptography from Xidian University, Xian, China in 2012. Now, he is a Professor and Ph.D. supervisor in the College of Computer Science and Technology at Guizhou University. His current research interests include algorithmic game theory, cryptography and security protocols, big data security and privacy protection, blockchain and electronic currency *etc*.

Sheng Gao is an Assistant Professor in the School of Information at Central University of Finance and Economics. He received the B.S. degree in Information and Computation Science from Xi'an University of Posts and Telecommunications, in 2009, and the Ph.D. degree in Computer Science and Technology from Xidian University, in 2014. His current research interests include finance information security and privacy computing.

Chang-Gen Peng received his Ph.D. degree from the Department of Computer Science and Technology, Guizhou University, China, in 2007. He is presently a Professor, Ph.D. supervisor at the College of Computer Science and Technology, Guizhou University. He is an academic leader of data security and cryptography at State Key Laboratory of Public Big Data. His research interests include data privacy, cryptography and big data technology and security.

Wei-Jie Tan received the M.S. degree in Communication and Information System from the Communication University of China, Beijing, China, in 2011, and the Ph.D. degree in Information and Communications Engineering from Northwestern Polytechnical University, Xi'an, China, in 2019. From 2016 to 2017, he was a visiting researcher with the Audio Analysis Laboratory, AD:MT, Aalborg University, Denmark. He is currently with the faculty of the State Key Laboratory of Public Big Data, Guizhou University. His research interests include communication network security, communication signal processing, array signal processing, and sparse signal processing.