IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 15, 2021, accepted December 31, 2021, date of publication January 11, 2022, date of current version January 18, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3141709

Extremely Randomized Trees With Privacy Preservation for Distributed Structured Health Data

AMIN AMINIFAR[1], **MATIN SHOKRI**[2], **FAZLE RABBI**[1,3], **VIOLET KA I. PUN**[1,4], **AND YNGVE LAMO**[1]

[1]Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, 5063 Bergen, Norway
[2]Faculty of Computer Engineering, K. N. Toosi University of Technology, Tehran 19697-64499, Iran
[3]Department of Information Science and Media Studies, University of Bergen, 5007 Bergen, Norway
[4]Department of Informatics, University of Oslo, 0315 Oslo, Norway

Corresponding author: Amin Aminifar (amin.aminifar@hvl.no)

This work was supported by the INTROducing Mental health through the Adaptive Technology (INTROMAT) Project by the Norwegian Research Council (NFR) under Grant 259293.

ABSTRACT Artificial intelligence and machine learning have recently attracted considerable attention in the healthcare domain. The data used by machine learning algorithms in healthcare applications is often distributed over multiple sources, for instance, hospitals or patients' personal devices. One main difficulty lies in analyzing such data without compromising patients' privacy and personal data, which is a primary concern in healthcare applications. Therefore, in these applications, we are interested in running machine learning algorithms over distributed data without disclosing sensitive information about the data subjects. In this paper, we propose a distributed extremely randomized trees algorithm for learning from distributed data with privacy preservation. We present the implementation of our technique (which we refer to as *k*-PPD-ERT) on a cloud platform and demonstrate its performance based on medical data, including Heart Disease, Breast Cancer, and mental health datasets (Depresjon and Psykose datasets) associated with the Norwegian INTROducing Mental health through Adaptive Technology (INTROMAT) project.

INDEX TERMS Distributed learning, extremely randomized trees, privacy-preserving machine learning, structured health data, federated machine learning.

I. INTRODUCTION

Artificial intelligence (AI) and automated decision-making have the potential to improve accuracy and efficiency in healthcare applications. In particular, AI is proven to outperform medical experts in certain domains. Two examples are the classification of rhythms in electrocardiography signals with deep neural networks in [1] and prediction of breast cancer using the AI system presented in [2]; more related studies can be found in [3], [4]. However, the application of AI and machine learning for automated decision-making in healthcare comes with challenges, such as security and privacy. For instance, a patient's privacy is violated if sharing his/her medical data with a third-party data recipient reveals that he/she has a medical condition. This becomes more challenging considering that, in healthcare systems, the data could be distributed over a number of sources rather than being stored in a central database.

In distributed settings, hospitals need to apply data mining methods to extract useful patterns from patients' data. Although hospitals may individually be able to use their limited resources and locally stored health information to perform data mining, the use of available health information across several hospitals leads to obtaining more valuable and accurate information. However, this is a challenging task due to privacy and legal concerns. Hospitals often need to comply with privacy regulations that restrict sharing health information about patients with other parties, e.g., other hospitals, family doctors, and specialists [5], [6]. A similar problem exists when the data is distributed over patients' personal devices, such as mobile phones or wearable devices [7]–[11].

Traditionally, it was assumed that all sources holding part of the data might share their information with a trusted party. However, such an assumption, i.e., putting this level of trust

The associate editor coordinating the review of this manuscript and approving it for publication was Liangxiu Han.

in a third party, is not feasible in every scenario because the privacy of data sources cannot be protected from the third party [12]. In order to address the privacy concern, one solution would be to perturb the data before sharing it. However, perturbation-based solutions have limitations in satisfying data privacy and data utility requirements [13], [14]. This is because the utility of the data will decrease if the perturbation is not precisely controlled, and the privacy will not be preserved if the perturbation is not sufficient [14]. Similarly, anonymization techniques, e.g., [15]–[20], share an altered version of data to prevent the re-identification of data subjects [21]. Moreover, methods providing differential privacy [22] share data while preserving the privacy of individuals by adding noise. Nevertheless, in these techniques, there is always a trade-off between data privacy and data utility [13].

Previous studies also consider cryptographic techniques and secure multi-party computation methods for conducting privacy-preserving data mining [23]–[25]. However, such approaches are inefficient, mainly when dealing with large-scale data, due to considerable communication and computation costs [14]. Several techniques, e.g., [12], [26], [27], have been proposed to address these types of overheads in the privacy-preserving machine learning algorithms and to improve their efficiency.

In this paper, we target the problem of learning from data held on multiple sources without explicit sharing of raw information. We assume that the learning data is horizontally partitioned, meaning that different records of data are stored on different sources. We focus on the classification problem and structured health data, which can be stored in spreadsheets. We build upon our previous work [28] and propose a scalable privacy-preserving framework for distributed machine learning based on the extremely randomized trees algorithm, which has a linear overhead in the number of parties and can handle missing values. We refer to our approach as k-PPD-ERT (Privacy-Preserving Distributed Extremely Randomized Trees), in which k is the number of colluding parties in our approach. We use two popular publicly available healthcare datasets for performance evaluation, i.e., the Heart Disease [29] and the Breast Cancer Wisconsin (Diagnostic) [30] datasets. This data represents medical applications where missing values are present, and our algorithm is designed to handle such scenarios. Finally, we present the implementation of our technique on Amazon's AWS cloud and evaluate it in a real-world setting based on the mental health datasets associated with the Norwegian INTROducing Mental health through Adaptive Technology (INTROMAT) project [31].

The remainder of this paper is organized as follows. Section II reviews the state of the art of distributed privacy-preserving machine learning techniques to address the discussed problem. Section III covers the background related to the extremely randomized trees algorithm and secure multi-party computation. In Section IV, we illustrate our proposed k-PPD-ERT method, which is an adaptation and extension of the ERT algorithm for distributed settings. Section V illustrates the distributed extremely randomized trees algorithm through a small example. In Section VI, we evaluate the performance, overhead and privacy of the proposed technique. Section VII serves as the conclusion of this article.

II. STATE OF THE ART

The topic of collaborative learning from distributed data has been discussed in the literature for many years. A wide range of distributed learning techniques has been proposed in the literature that do not explicitly consider privacy aspects [26], [32]–[34]. Nevertheless, such techniques indirectly contribute to privacy preservation by limiting the amount of data that has to be shared with other parties or transferred to central servers or the cloud.

Randomization has been adopted in several studies [35]–[38] to preserve the privacy of individuals in data mining techniques. For instance, a technique that incorporates noise into raw data before sharing and performing data mining processes is proposed in [35]. However, the original values can be estimated using noise removal techniques. Hence, such techniques do not provide strong privacy guarantees [14], [39]–[41].

Secure multi-party computation (SMC) has been employed in several studies [12], [23]–[25], [42], [43] to perform data mining over data distributed in multiple parties, where no private information except the mining results should be disclosed. In SMC, we are interested in the result of a computation without knowing the secret values required for this computation. Therefore, techniques utilizing SMC usually compute intermediate results in the learning process without revealing the secret to other parties. Although such methods can satisfy the privacy requirements, the incorporation of inefficient secure computation techniques and homomorphic encryption in the method can substantially increase the communication and computation overheads. This leads to issues related to efficiency, particularly when we have a large number of parties or when we are dealing with a high volume of data [12].

Cryptographic methods have been adopted by several studies [23], [24], [44] for achieving privacy [14]. These methods address classification, clustering, anomaly detection, etc., by employing different data mining algorithms [45]–[48]. Nevertheless, such techniques usually suffer from communication and computation overheads and are impractical when dealing with large-scale data [49].

Federated learning has been proposed to collaboratively train a model, with the orchestration of one party, while keeping the training data decentralized [26], [32], [50]. Several systematic literature reviews of the state-of-the-art federated machine learning techniques are performed in [51]–[53]. The majority of previous studies in this domain have focused on deep neural network algorithms. In such neural network algorithms, in addition to data-holder parties' contribution, i.e., gradients, sharing model parameters is also a privacy concern.

This is due to recent attacks on the neural networks, i.e., membership inference attack [54], [55]. For addressing privacy concerns, previous studies adopt differential privacy [22] in their methods [56]–[58]. However, differential privacy can degrade the performance of the machine learning model due to the trade-off between privacy and data utility [13].

In many applications, tree-based methods can be more accurate than neural networks. Deep neural network algorithms are appropriate solutions when dealing with unstructured data, e.g., for video, audio, and text in [59]–[61]. However, the tree-based methods can outperform such algorithms when dealing with structure data, where the data attributes are individually meaningful, and we do not have strong multi-scale structures related to time or space [62]. Therefore, tree-based algorithms are currently being adopted in many applications in which the training data is structured.

Tree-based machine learning techniques have been investigated in conjunction with privacy concerns and distributed learning in several studies [12], [14], [42], [63]. In [14], the authors consider the problem of learning decision trees, with Random Decision Trees (RDT) algorithm [63]. They present a technique based on homomorphic encryption and apply it for horizontally and vertically partitioned datasets. However, this approach suffers from high computational complexity. In [42], the authors propose the utilization of SMC techniques for learning decision trees based on the ID3 algorithm [64]. In this approach, the data is horizontally partitioned and distributed among two parties. The number of parties in this method can be increased to more than two, but the efficiency and scalability of the technique decrease [49]. Moreover, perturbation techniques may also be used to build approximate decision trees. In [65], the authors propose the application of Randomized Response techniques to disguise the data before transferring it to a center for learning decision trees based on their modified ID3 algorithm. Nevertheless, transferring the entire data from all sources to one center, even after applying randomization techniques, undermines our confidence in the technique's privacy.

Gradient and tree-based algorithms have been employed by several studies in conjunction with strategies related to federated learning [66]–[69]. In [68], the authors propose a privacy-preserving distributed data mining method for regression and classification based on the Gradient Boosting Decision Tree (GBDT) algorithm [70]. The trees are trained locally on data-holder parties and passed to the following parties after being modified according to differential privacy requirements [68]. Nevertheless, injecting noise into participants' contribution, model parameters, etc., can increase the learning time and degrade the results of learning due to the trade-off between privacy and data utility [13]. Similarly, in [69], the authors propose a method based on GBDT for distributed scenarios called SimFL. In this framework, each party boosts a number of trees utilizing similarity information using locality-sensitive hashing. However, their privacy model is weaker than secure multi-party computation for improving efficiency, and their model performance is not the same as GBDT but comparable to it [69].

There are other studies that propose *tree-based methods that are not gradient-based* but are under the name of federated learning, e.g., [71], [72]. In [72], the authors propose a method employing the decision tree algorithm, ID3, that uses the combination of differential privacy and secure multi-party computation for addressing privacy concerns. The model's performance is degraded compared to the performance of the machine learning model in a centralized scenario. In [71], the authors propose a solution based on the random forest algorithm [73], [74]. This method requires a third-party trusted server and employs encryption, which increases the communication and computation overheads [12].

Closely connected to this work, the authors in [12] propose a tree-based method that utilizes a secure multi-party computation technique as an additional layer in their approach to have more confidence about its privacy. Particularly, Shamir's secret sharing [75] is used to aggregate the results received from each party at every step of learning with the ID3 algorithm. The limitation in the incorporation of methods with high communication and computation overheads leads to higher efficiency. However, Shamir's secret sharing technique still introduces major overheads in communication and computation and suffers from the scalability problem.

In our preliminary study [76], we have considered the problem of privacy-preserving machine learning using the extremely randomized trees algorithm, which is only robust to two colluding parties (in the worst-case scenario). We extend this idea to k colluding parties in [28]. However, this approach suffers from quadratic complexity in the worst-case scenario, i.e., $O(n^2)$, and is limited to datasets without missing values, which is rarely a case in real-world healthcare applications. In this work, we addressed these problems and proposed a scalable privacy-preserving distributed extremely randomized trees framework, with $O(kn)$ complexity, where k can be adjusted based on the sensitivity of the data. We implement our technique on Amazon's AWS cloud and evaluate it in a real-world setting based on the mental health datasets associated with the Norwegian INTROducing Mental health through Adaptive Technology (INTROMAT) project.

III. BACKGROUND

In this section, we present a brief overview of the extremely randomized trees (ERT) algorithm and secure multi-party computation (SMC), which provide the basis for our privacy-preserving distributed machine learning framework.

A. THE ERT ALGORITHM

ERT [77] is a tree-based ensemble learning algorithm that has been widely used for solving classification problems due to its learning performance and robustness to overfitting, which are among the characteristics of tree-based ensemble learning algorithms [62], [78], [79]. However, the traditional ERT algorithm is used when the data is stored in a central location.

We adapt the ERT algorithm for distributed settings where data is stored and essentially distributed among several parties. In the following, we discuss some of the advantages of the ERT algorithm compared to other available solutions for its utilization in distributed settings.

First, since the ERT algorithm is an ensemble learning method, it is robust in tackling overfitting. Ensemble learning methods incorporate weak learners to generate weak classifiers that are independent of other generated classifiers. Therefore, based on Condorcet's jury theorem (1785) [80], the majority vote of this ensemble of learned classifiers predicts better than the vote of an individual classifier, and if we increase the number of classifiers, the accuracy improves [81]. Therefore, in the ensemble learning method, we generate a collection of classifiers instead of only one, e.g., in [12], and finally predict based on the voting result of the learned classifiers. In such ensemble learning methods, randomness parameters in the learning algorithm cause generating classifiers different from each other. In the ERT algorithm, the randomness of candidate attributes and the splitting point for every decision node in the tree are the randomness parameters [77], which result in learning different classifiers. The ERT approach follows the logic of bagging in ensemble learning. Bagging combines the learned classifiers by voting, i.e., it predicts based on the majority vote among the learned classifiers. While not increasing the bias, bagging leads to lower variance in our learned model since we are averaging, and the lower variance in the learned model reduces the risk of overfitting [78].

Second, ERT is tree-based, and tree-based algorithms have been shown to outperform other techniques for structured data that we are addressing. In [62], the authors report that for tabular-style data where the data attributes are individually meaningful and where we do not have strong multi-scale structures related to time or space, learned models from tree-based algorithms usually outperform models learned by standard deep neural networks, e.g., [26], [32]. Moreover, in the health domain's applications, the interpretability of the learned models is advantageous. The patterns that tree-based learned models unveil, particularly in the healthcare domain, may be more useful than the prediction capability of the learned model [62]. Tree-based algorithms are more interpretable compared to deep neural networks [79]. This is an advantage for ERT. However, since ERT is an ensemble learning method, and in ensemble methods, instead of learning a model with a single tree, e.g., in the ID3 algorithm [64], the algorithm constructs several trees as a model. Hence, this decreases the explainability of such approaches compared to the ID3 algorithm.

B. SECURE MULTI-PARTY COMPUTATION

The secure multi-party computation framework, initiated by Yao's Millionaires' problem [82], considers the problem of collaborative computation among several parties, each of which holds a secret value. The parties are interested in the result of a computation performed based on their secret values, while they refrain from sharing their secret values with other parties.

A simple solution for computing the desired value without sharing secret values with other parties is to share them with a party that is trusted by everyone. The trusted party can then perform the computation and return the result to all parties. However, the assumption of trusted parties is not feasible in many scenarios because the privacy of parties with secret values cannot be protected from the third party, so such solutions are not practical. Therefore, based on the type of the computation and the scenarios, we need to devise other solutions to perform the desired collaborative computation in a secure way and without violating privacy.

To illustrate SMC, we describe a simple method for secure aggregation of secret values. Figure 1 represents the method for secure aggregation. In this example, we have four parties, each holding a secret value ($S.V.$), and the parties are interested in the summation of all secret values, i.e., $\sum_{i=1}^{4} S.V._i$. For securely aggregating the secret values:

(i) The first party generates a random mask, aggregates it with its secret value ($S.V._1$), and sends the result to the next party.

(ii) The following parties receive the input, aggregate it with their secret values, and send the result to the next party. The last party sends the result to the first party.

(iii) The first party receives the result from the last party, removes its random mask from the result, and informs all parties about the final result.

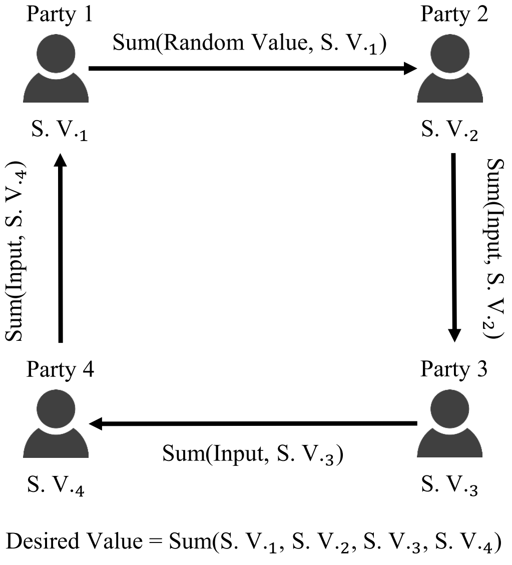


FIGURE 1. Secure aggregation.

In this way, each party cannot identify the secret value of the previous parties based on the received information. However, in this method, if two neighboring parties, i.e., the parties before and after a certain party in the ring, collude, they will be able to identify the secret value of the victim party. For instance, if Party 2 reveals the input of Party 3, and at the same time, Party 4 reveals the output of Party 3, then they can reveal the secret value of Party 3. Therefore, the minimum number of colluding parties required for identifying a secret value is two in this method. Moreover, in terms of overhead, for one secure computation operation in this method, each party sends one message and receives one message. Thus, the communication overhead for this method is $2n$, in which n is the number of parties.

IV. PRIVACY-PRESERVING DISTRIBUTED EXTREMELY RANDOMIZED TREES

This section presents the proposed solution, which is based on the extremely randomized trees (ERT) algorithm and the secure multi-party computation (SMC) scheme. As mentioned in Section I, we refer to our approach as k-PPD-ERT, where k is the number of colluding parties in our approach in the secure aggregation process. Note that k is a parameter that can be tuned based on the privacy requirements. The algorithm preserves privacy since, on the one hand, the algorithm is distributed and the raw data is not directly shared, and, on the other hand, the partial information is aggregated using a secure multi-party computation technique. Finally, our proposed framework is based on the ERT or Extremely Randomized Trees algorithm in [77].

A. ADAPTATION OF ERT FOR DISTRIBUTED SETTINGS

This section presents the detailed procedure of learning an ensemble of decision trees based on the ERT algorithm in the discussed setting. The pseudocode of the algorithm is also provided for clarity.

1) INITIALIZATION AND START OF THE LEARNING PROCESS

We have two types of parties in our distributed learning framework. We have a *mediator* that mediates and orchestrates the overall learning process and several *data-holder parties* that collaborate with each other and the mediator to learn a classification model. Algorithm 1 and Algorithm 2 show the pseudocodes of the procedures and functions for the mediator and data-holder parties, respectively.

(a) **Sharing the Random Seeds**
To start this process, a global seed for the random function is agreed upon among all parties (Algorithm 1, Line 1 and Algorithm 2, Line 1). The global seed is common among the mediator and all data holders. In the ERT algorithm, we have two parameters of randomness for learning a weak classifier. First, we need to randomly select several attributes for the candidate decision nodes, at every step of building our decision tree (Algorithm 1, Line 24 Algorithm 2, Line 25). Second, a random splitting point for every attribute in the candidate decision node is required (Algorithm 1, Line 25, and Lines 28–35, and Algorithm 2, Line 26, and Lines 29–36). The data-holder parties and the mediator are required to use the same candidate decision nodes at every step when learning a decision tree. For this purpose, we use the global random seed that all parties, including the mediator, utilize to locally generate these candidate decision nodes (Algorithm 1, Line 11, and Algorithm 2, Line 17). This is instead of making these randomly-made candidate decision nodes in the

Algorithm 1 Mediator

```
1  • The global random seed (known to all parties) is set in
     the mediator
2  • Wait for data-holder parties' connection
3  for $i = 1$ to $M$ do
4      • Generate tree: $t_i$ = Build_k-PPD-ERT(0, 'None')
5  end
6  $E = \{t_1, t_2, \ldots, t_M\}$
7  Function Build_k-PPD-ERT(Split_ID, Branch)
8      • Send Secret_aggregation(Split_ID, Branch)
         request to data-holder parties
9      • Wait until receiving the results from data-holder
         parties
10     • Sum = aggregated the received results form
         data-holder parties
11     • Generate_splits() (based on the global seed)
12     if number of classified records is less than $n_{min}$ or
         labels of the classified records are the same then
13         return a leaf label
14     else
15         • Calculate each split's score (Information
             Gain) based on Sum
16         • Select the split with the highest score.
17         • Inform all parties about the selected split (for
             Split_ID)
18         • Build tree_T = Build_k-PPD-ERT(next
             Split_ID, 'T')
19         • Build tree_F = Build_k-PPD-ERT(next
             Split_ID, 'F')
20         • Create a node with the selected split, attach
             tree_T and tree_F as T and F subtrees, and
             return the resulting tree.
21     end
22 end
23 Function Generate_splits()
24     • Select D attributes randomly: $\{a_1, \ldots, a_D\}$
25     • Generate D splits: $\{s_1, \ldots, s_D\}$, where $s_i$ =
         Pick_rand_split($a_i$)
26     return splits $\{s_1, \ldots, s_D\}$
27 end
28 Function Pick_rand_split(a)
29     if a is categorical then
30         return a possible category
31     end
32     if a is numerical then
33         return a possible value in the min and max range
34     end
35 end
```

Algorithm 2 Data-Holder Party

```
1  • The global random seed (known to all parties) is set in
   the data-holder party
2  • Wait for completion of data-holder parties
   initialization. In initialization, $k$ selected data-holder
   parties send their unique seeds to other data holders.
   In initialization, $SSA_{pj}^{pi}$ is sent by party $i$ ($i$ is among the $k$
   selected parties) and received by party $j$
3  • Connect to the Mediator
4  Function Secret_aggregation(Split_ID, Branch)
5      • $secret\_val^{P_j}$ = Split_data(Split_ID, Branch)
6      • $rand\_sum_{others}^{P_j}$ = Generate and aggregate random
       masks based the received seeds
7      if the party, $P_j$, is among $k$ selected data-holder
       parties for secure aggregation then
8          • $rand\_sum_{self}^{P_j}$ = Generate and aggregate
           random masks based the sent seeds
9      else
10         • $rand\_sum_{self}^{P_j} = 0$
11     end
12     • Result =
       $secret\_val^{P_j} - rand\_sum_{self}^{P_j} + rand\_sum_{others}^{P_j}$
13     • Send Result to the mediator
14 end
15 Function Split_data(Split_ID, Branch)
16     • $S_{sub}$ = records in the computational node that
       should be split based on Split_ID and Branch
17     • $\{s_1, \ldots, s_D\}$ = Generate_splits() (based on the
       global seed)
18     for $i = 1$ to $D$ do
19         • Split $S_{sub}$ to two sets (T, F) by $s_i$
20         • Append vectors $\{Vec_T, Vec_F\}$ representing the
           records' labels for each of the above sets to
           Result
21     end
22     return Result
23 end
24 Function Generate_splits()
25     • Select $D$ attributes randomly: $\{a_1, \ldots, a_D\}$
26     • Generate $D$ splits: $\{s_1, \ldots, s_D\}$, where $s_i$ =
       Pick_rand_split($a_i$)
27     return splits $\{s_1, \ldots, s_D\}$
28 end
29 Function Pick_rand_split(a)
30     if a is categorical then
31         return a possible category
32     end
33     if a is numerical then
34         return a possible value in the min and max range
35     end
36 end
```

mediator and sharing them with all parties for further tasks. Since all parties use a common random seed, i.e., the global random seed, they generate the same candidate decision nodes at every step, without major communication overhead.

In addition, for the secure aggregation of partial results, described further in Section IV-B, k selected data-holder parties send unique seeds for the random function to other data holders through secure communication (Algorithm 2, Line 2). These random seeds are exclusive and private for each pair of data-holder parties.

(b) **Initiate the Process of Learning One Decision Tree**

The privacy-preserving distributed ERT algorithm is an ensemble learning method, therefore, we repeat the process of learning a decision tree for M times, until we have M decision trees (Algorithm 1, Lines 3–5). The number of trees, M, is a parameter tuned by the user to make a trade-off between robustness and overhead. We learn different decision trees every time due to the randomness in ERT. Finally, after repeating the process of learning a decision tree M times, we store the trees in E (Algorithm 1, Line 6). For future prediction, the ensemble of the learned trees, E, will be used.

2) THE PROCESS OF LEARNING ONE DECISION TREE

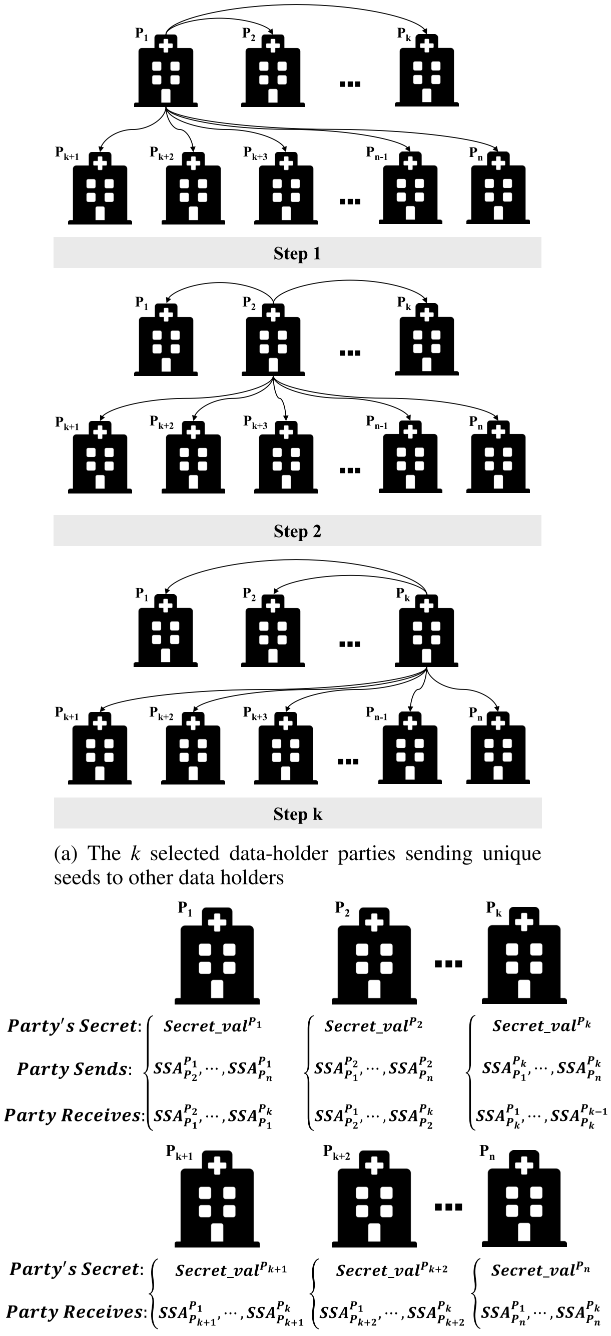
The learning of a decision tree based on the privacy-preserving distributed ERT algorithm is a recursive procedure. The procedure is executed top-down and starts from the root and ends in the leaves. For the root decision node, the *Split_ID* or the ID for the decision node is zero, and there is no previous branch, so the *Branch* input is set to '*None*'(Algorithm 1, Line 4).

(a) **Generation of Candidate Decision Nodes**

For building each decision tree, extremely randomized tree, the mediator generates the candidate decision nodes (Algorithm 1, Line 11). The mediator will further select the best decision node among the candidates based on the results received from data-holder parties. The candidate decision nodes are generated randomly, based on the global random seed, according to Algorithm 1, Lines 23–35, and Algorithm 2, Lines 24–36. The number of candidate decision nodes, D, is a parameter in the ERT algorithm tuned by the user. D attributes from all possible attributes are selected for candidate decision nodes (Algorithm 1, Line 24, and Algorithm 2, Line 25). Then, each candidate decision node's splitting point is selected (Algorithm 1, Line 25, and Algorithm 2, Line 26). If the attribute is categorical, one random possible category is selected to be checked (Algorithm 1, Lines 29–31, and Algorithm 2, Lines 30–32); otherwise, when the attribute is numerical, a point in the possible range is selected for comparison in the decision node (Algorithm 1, Lines 32–34, and Algorithm 2, Lines 33–35). We assume that all parties already have the possible categories and ranges for each attribute.

(b) **Parties Classify Their Records**

To decide about the candidate decision nodes for each branch, the mediator requires the collective outcome of


P1
P2
Pk
Pk+1
Pk+2
Pk+3
Pn-1
Pn
Step 1
Step 2
Step k

(a) The k selected data-holder parties sending unique seeds to other data holders

P1
P2
Pk
Party's Secret: $Secret\_val^{P_1}$
Party Sends: $SSA_{P_2}^{P_1}, \cdots, SSA_{P_n}^{P_1}$
Party Receives: $SSA_{P_1}^{P_2}, \cdots, SSA_{P_1}^{P_k}$
$Secret\_val^{P_2}$
$SSA_{P_1}^{P_2}, \cdots, SSA_{P_n}^{P_2}$
$SSA_{P_2}^{P_1}, \cdots, SSA_{P_2}^{P_k}$
$Secret\_val^{P_k}$
$SSA_{P_1}^{P_k}, \cdots, SSA_{P_n}^{P_k}$
$SSA_{P_k}^{P_1}, \cdots, SSA_{P_k}^{P_{k-1}}$
Pk+1
Pk+2
Pn
Party's Secret: $Secret\_val^{P_{k+1}}$
Party Receives: $SSA_{P_{k+1}}^{P_1}, \cdots, SSA_{P_{k+1}}^{P_k}$
$Secret\_val^{P_{k+2}}$
$SSA_{P_{k+2}}^{P_1}, \cdots, SSA_{P_{k+2}}^{P_k}$
$Secret\_val^{P_n}$
$SSA_{P_n}^{P_1}, \cdots, SSA_{P_n}^{P_k}$

(b) The sent and received seeds after the initialization

FIGURE 2. Initialization.

the classification with candidate decision nodes from all data holders on all their data. By having the combination of data record labels for each branch (*True* and *False*), the mediator can decide if we require a leaf or we need to calculate the score, i.e., information gain (Algorithm 1, Line 12). Information gain captures the extent of samples' purity (concerning their class/category) after splitting and is used as a basis for comparing decision nodes. The mediator sends a request to data-holder parties and waits for receiving the result from all parties, which is masked according to the secure aggregation technique described in Section IV-B (Algorithm 1, Lines 8–9). The masked results are two vectors, one for each of the *True* and *False* branches, representing the combination of data record labels after classification with each candidate decision node.

Each party receives *Split_ID* and *Branch* to determine the local records for classification (Algorithm 2, Line 16). Then, the party randomly generates candidate decision nodes based on Lines 24–36 in Algorithm 2 and the global random seed (Algorithm 2, Line 17). Next, it classifies the selected local data based on each candidate decision node and returns the result (Algorithm 2, Lines 18–22).

We describe how each party returns the result to the mediator in the following, using an example. Vec_T represents the combination of labels for the records that fall in the *True* branch, and Vec_F represents the combination of labels for the records that fall in the *False* branch. For instance, if three records with labels A, A, and B fall in the *True* branch of the candidate decision node, and we have three labels, A, B, and C in the dataset, then $Vec_T = [2, 1, 0]$.

(c) **Each Party Sends the Result to the Mediator**

After adopting the secure aggregation protocol described in Section IV-B, each data-holder party returns the masked result to the mediator to select the best decision node (or generate a leaf instead of a decision node). For every candidate decision node, the mediator receives and aggregates the results from all parties and obtains two vectors, for *True* and *False* branches, representing the combination of data labels (Algorithm 1, Lines 9–10).

(d) **Mediator Determines the Best Candidate for the Decision Node**

Now that the mediator has the value of *Sum* (Algorithm 1, Line 10), it determines if a decision node or a leaf node is required here in the tree (Algorithm 1, Lines 12). If all labels are the same or if the number of received labels is less than our threshold parameter, the mediator introduces a leaf node (Algorithm 1, Line 13). Otherwise, the mediator calculates the score, i.e., information gain, of each candidate decision node based on the results from data-holder parties (Algorithm 1, Line 15). It then selects the candidate decision node with the highest information gain and informs all parties about it (Algorithm 1, Lines 16–17). The selected node will be used to build the tree at the mediator (Algorithm 1, Line 20). This decision is communicated to all data-holder parties and is required to select records for classification at every step (Algorithm 2, Line 16).

(e) **The Mediator Initiates Another Round From the First Step**

After selecting the best candidate decision node, the mediator continues the process for each branch of this decision node. Therefore, the same process is performed from the first step, for each of the *True* and *False* branches (Algorithm 1, Lines 18–19). After returning from these recursive calls, the selected subtrees for each branch are returned (Algorithm 1, Lines 13 and 20).

B. SECURE AGGREGATION OF RESULTS FROM DATA-HOLDER PARTIES

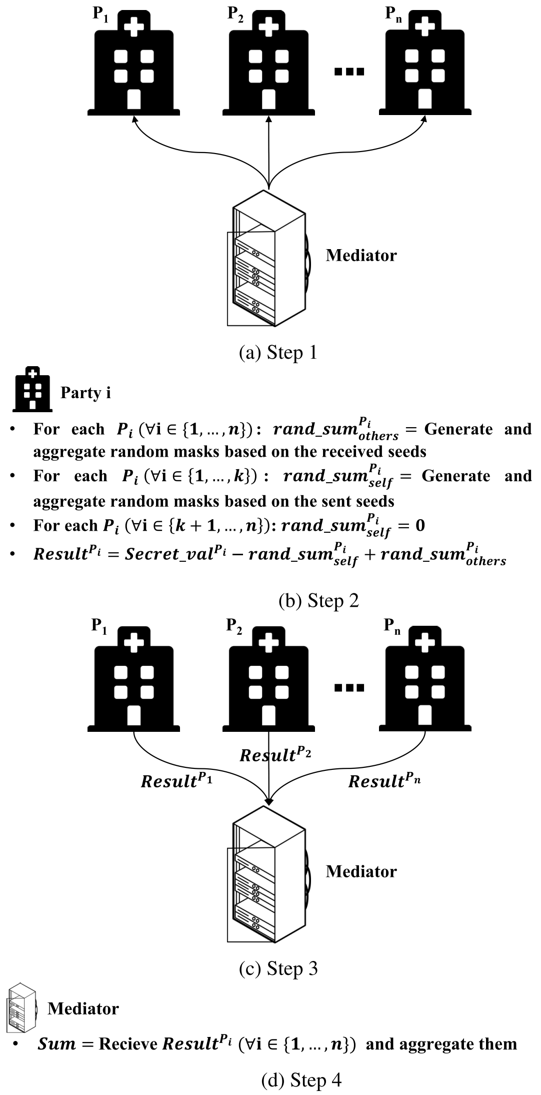
We adopt an SMC technique in our proposed distributed ERT algorithm to avoid sharing the vectors representing the combination of the data record labels for each candidate decision node and each branch in each data-holder party. In addition to the provided privacy by not sharing the raw values of data attributes, which is by construction, the adoption of an SMC technique for aggregating the partial results from data-holder parties contributes to privacy preservation. In an extreme example, suppose our data has one sensitive attribute in it, e.g., having conducted transgender surgery before, and each data-holder party has only one record on it. Then, sharing the partial results from one party, the vectors for the combination of data record labels for each candidate decision node, can reveal sensitive information. If the candidate decision node is "whether the record falls into the transgender branch or not," the mediator can infer if that individual with the specified record has undergone transgender surgery. Therefore, to avoid such vulnerabilities, we adopt an SMC technique to aggregate the partial results from the data-holder parties.

The secure aggregation procedure begins with an initialization process. Subsequently, the parties can securely aggregate their secret values through this approach.

1) INITIALIZATION

In the initialization phase, k selected data-holder parties share their unique seeds for the random function with all parties. These seeds are unique and private between each pair of parties. Without loss of generality and for the simplicity of the presentation, we assume that the k selected data-holder parties are P_i ($\forall i \in \{1, \ldots, k\}$). Party P_i ($\forall i \in \{1, \ldots, k\}$) sends unique seeds to party P_j ($\forall j \in \{1, \ldots, n \mid i \neq j\}$). Figure 2a shows this process.

The seed party P_i shares with party P_j is represented with $SSA_{P_j}^{P_i}$, and it is a unique seed; *SSA* is the short form of Seed for Secure Aggregation. Parties *1* to k, send $n-1$ and receive $k-1$ seeds. Parties $k+1$ to n, receive k seeds. This is shown in Figure 2b. Therefore, k parties send $n-1$ and receive $k-1$ messages, and $n-k$ parties send zero and receive k messages. The total communication overhead for initialization is $2k(n-1)$. The communication overhead by adopting this approach is equal to $O(kn)$, which can be adjusted by adapting k based on the sensitivity of the data. If all parties were required to send and receive seed, then, the communication overhead would be equal to $2n(n-1)$. The communication overhead by adopting this approach is equal to $O(n^2)$ [28].



(a) Step 1

Party i

- For each P_i ($\forall$i $\in \{1, \ldots, n\}$): $rand\_sum_{others}^{P_i}$ = Generate and aggregate random masks based on the received seeds
- For each P_i ($\forall$i $\in \{1, \ldots, k\}$) : $rand\_sum_{self}^{P_i}$ = Generate and aggregate random masks based on the sent seeds
- For each P_i ($\forall$i $\in \{k+1, \ldots, n\}$): $rand\_sum_{self}^{P_i} = 0$
- $Result^{P_i} = Secret\_val^{P_i} - rand\_sum_{self}^{P_i} + rand\_sum_{others}^{P_i}$

(b) Step 2

(c) Step 3

Mediator

- Sum = Recieve $Result^{P_i}$ ($\forall$i $\in \{1, \ldots, n\}$) and aggregate them

(d) Step 4

FIGURE 3. **Secure aggregation.**

2) SECURE AGGREGATION

In the adopted SMC technique, shown in Figure 3, parties add random masks to their partial result vectors and pass them to the mediator. The mediator aggregates the partial results received from all parties. After aggregation, the random masks from all parties cancel each other. We now describe the proposed technique in detail:

- **Step 1:** The mediator initiates the secure aggregation process round (Algorithm 1, Line 8). This is shown in Figure 3a.
- **Step 2:** Data-holder parties generate random masks and aggregate them with their secret values (Algorithm 2, Line 12). This is shown in Figure 3b.
  - Parties P_i ($\forall i \in \{1, \ldots, k\}$) generate random masks based on the sent and received seeds (Algorithm 2, Lines 6–11).
  - Parties P_i ($\forall i \in \{k+1, \ldots, n\}$) generate random masks based on received seeds (Algorithm 2, Lines 6–11).

- **Step 3:** In the next step, the parties send the masked results to the mediator (Algorithm 2, Line 13). Then, the mediator receives the results from all parties (Algorithm 1, Line 9). Figure 3c shows this.
- **Step 4:** In the last step, the mediator aggregates all the received results to obtain the desired value, i.e., the aggregated secret values from all parties (Algorithm 1, Line 10). This is shown in Figure 3d.

3) PRIVACY

We now show that the secret values of the parties are kept private in our proposed protocol. The partial result $Result^{P_i}$, which is shared with the mediator, consists of three components: $secret\_val^{P_i}$, $rnd\_sum_{self}^{P_i}$, and $rnd\_sum_{others}^{P_i}$. The two components, $rnd\_sum_{self}^{P_i}$ and $rnd\_sum_{others}^{P_i}$, mask the secret value.

- For P_i ($\forall i \in \{1, \dots, k\}$), the value of $rnd\_sum_{self}^{P_i}$ can only be identified by the collusion of $n - 1$ parties holding the random seeds for generating the random masks, which are the components of $rnd\_sum_{self}^{P_i}$. At the same time, $rnd\_sum_{others}^{P_i}$ can only be identified by the collusion of $k - 1$ parties that generate the components of $rnd\_sum_{others}^{P_i}$. Therefore, the minimum number of colluding parties required to reveal the secrete value of P_i is $n - 1$.
- For P_i ($\forall i \in \{k + 1, \dots, n\}$), the value of $rnd\_sum_{self}^{P_i}$ is zero and known to all, and $secret\_val^{P_i}$ is masked by $rnd\_sum_{others}^{P_i}$. However, $rnd\_sum_{others}^{P_i}$ can only be identified by the collusion of k parties that generate the components of $rnd\_sum_{others}^{P_i}$, i.e., the k selected parties for secure aggregation.

In the worst case, i.e., for P_i ($\forall i \in \{k + 1, \dots, n\}$), the k selected parties for secure aggregation are required to collude to identify a secret value; hence, the minimum number of colluding data-holder parties is equal to k. Moreover, since only the mediator receives the victim's partial result, the collusion of other parties without the mediator's participation is not possible. Therefore, for identifying a secret value, the collusion of k data-holder parties and the mediator is necessary.

4) CORRECTNESS

We also show that the final value of the aggregation of partial results is equal to the aggregation of secret values. The aggregation of all the partial results sent to the mediator is as follows:

$$\begin{aligned}
&\sum_{j=1}^{n} Result^{P_j} \\
&= secret\_val^{P_1} - rnd\_sum_{self}^{P_1} + rnd\_sum_{others}^{P_1} \\
&\quad \vdots \\
&\quad + secret\_val^{P_n} - rnd\_sum_{self}^{P_n} + rnd\_sum_{others}^{P_n} \\
&= \sum_{j=1}^{n} secret\_val^{P_j} - \sum_{j=1}^{n} rnd\_sum_{self}^{P_j} + \sum_{j=1}^{n} rnd\_sum_{others}^{P_j}.
\end{aligned} \tag{1}$$

In addition, we also have the following equations for the data-holder parties:

- For P_i ($\forall i \in \{1, \dots, k\}$), $rnd\_sum_{self}^{P_i} = \sum_{j=1}^{n} rnd_{P_j}^{P_i} - rnd_{P_i}^{P_i}$, where $rnd_{P_j}^{P_i}$ is the shared random mask between P_i and P_j. On the other hand, $rnd\_sum_{others}^{P_i} = \sum_{j=1}^{k} rnd_{P_i}^{P_j} - rnd_{P_i}^{P_i}$.
- For P_i ($\forall i \in \{k + 1, \dots, n\}$), $rnd\_sum_{self}^{P_i} = 0$. On the other hand, $rnd\_sum_{others}^{P_i} = \sum_{j=1}^{k} rnd_{P_i}^{P_j}$.

Substituting these in Equation 1, we obtain:

$$\begin{aligned}
&\sum_{j=1}^{n} Result^{P_j} \\
&= \sum_{j=1}^{n} secret\_val^{P_j} - \sum_{j=1}^{n} rnd\_sum_{self}^{P_j} + \sum_{j=1}^{n} rnd\_sum_{others}^{P_j} \\
&= \sum_{j=1}^{n} secret\_val^{P_j} - \sum_{j=1}^{k} (\sum_{i=1}^{n} rnd_{P_i}^{P_j} - rnd_{P_j}^{P_j}) - \sum_{j=k+1}^{n} (0) \\
&\quad + \sum_{j=1}^{k} (\sum_{i=1}^{k} rnd_{P_j}^{P_i} - rnd_{P_j}^{P_j}) + \sum_{j=k+1}^{n} (\sum_{i=1}^{k} rnd_{P_j}^{P_i}) \\
&= \sum_{j=1}^{n} secret\_val^{P_j} - \sum_{j=1}^{k} (\sum_{i=1}^{n} rnd_{P_i}^{P_j}) + \sum_{j=1}^{k} (rnd_{P_j}^{P_j}) \\
&\quad + \sum_{j=1}^{k} (\sum_{i=1}^{k} rnd_{P_j}^{P_i}) - \sum_{j=1}^{k} (rnd_{P_j}^{P_j}) + \sum_{j=k+1}^{n} (\sum_{i=1}^{k} rnd_{P_j}^{P_i}) \\
&= \sum_{j=1}^{n} secret\_val^{P_j} - \sum_{i=1}^{n} (\sum_{j=1}^{k} rnd_{P_i}^{P_j}) + \sum_{j=1}^{n} (\sum_{i=1}^{k} rnd_{P_j}^{P_i}) \\
&= \sum_{j=1}^{n} secret\_val^{P_j}.
\end{aligned} \tag{2}$$

The above equation shows that the aggregation of partial results from data-holder parties is equal to the aggregation of data-holder parties' secret values.

As shown above, the correctness and accuracy of our SMC technique do not depend on k or the minimum number of colluding parties. By increasing k, the minimum number of colluding parties required for revealing a secret value increases, which in turn improves the privacy of the method. Increasing k increases the communication overhead in the initialization phase. Therefore, the trade-off is between privacy and communication overhead of the initialization phase.

C. HANDLING MISSING VALUES

In this section, handling missing values when the data is distributed is explained in the context of our proposed privacy-preserving distributed learning framework, i.e., k-PPD-ERT. In the application of distributed learning approaches, particularly in the healthcare domain, we deal with data with missing values. Missing values in a dataset may occur as a result of improper collection of data, refusal of

TABLE 1. Example of structured data distributed among two parties with missing values.

Party	Record	Sex	Height
	1	M	170
1	2	F	155
	3	M	?
	1	F	?
2	2	F	165
	3	M	178

patients to share information, etc. In scenarios where the data is distributed, handling missing values can require a different procedure in comparison to scenarios in which the data is held in one center.

Several approaches can still be used in such scenarios, e.g., deleting records with missing values. However, they might not be helpful in all cases, e.g., where we have a low number of data records or when the percentage of records with missing values is high. Another solution is to replace the missing values in an attribute with the mean/average of the available values in that attribute. This approach avoids deleting data records and is particularly relevant when dealing with smaller datasets with missing values.

For calculating the mean of the available values for an attribute, we require the summation of these values. Due to privacy concerns, data-holder parties refrain from sharing the summation of their available values with others. In particular, this is a major privacy concern when each data-holder party holds only one record. Therefore, we adopt the approach presented in Section IV-B to address this issue, as we merely require the final summation of the available values.

We explain the approach using an example. Suppose we have two parties, and each party holds three records. Table 1 represents the data for each party. Each record contains the sex and height of record owners or patients. Two records miss the value for height. Assume that by preserving privacy, we can calculate the summation of available values for the height, i.e., 668 in our example, as well as the summation of the number of records not missing the height value, i.e., 4 in our example. In that case, we can calculate the mean for the height, i.e., 167 in our example.

The summation of the available values and the number of available values are calculated using our secure aggregation method. Finally, the mediator divides the summation of the available values by the number of available values and calculates the mean. Then, the mean is shared with all parties to replace the missing values.

Our technique may also be modified based on the problem settings. For instance, in the above example, suppose the user requires the mean of values for male and female patients separately, i.e., 174 and 160, respectively. Then, our technique can be adjusted by only securely aggregating the available values belonging to male or female patients.

We use the same technique for categorical attributes, i.e., to calculate the frequencies of categories in one attribute. Then, we may decide how to fill the missing values based on these frequencies. We may decide to replace all values with the most frequent category, i.e., the mode. The missing category can also be drawn randomly based on the distribution of frequencies. Moreover, we may also decide on filling the missing values by jointly considering the frequencies and information from other attributes.

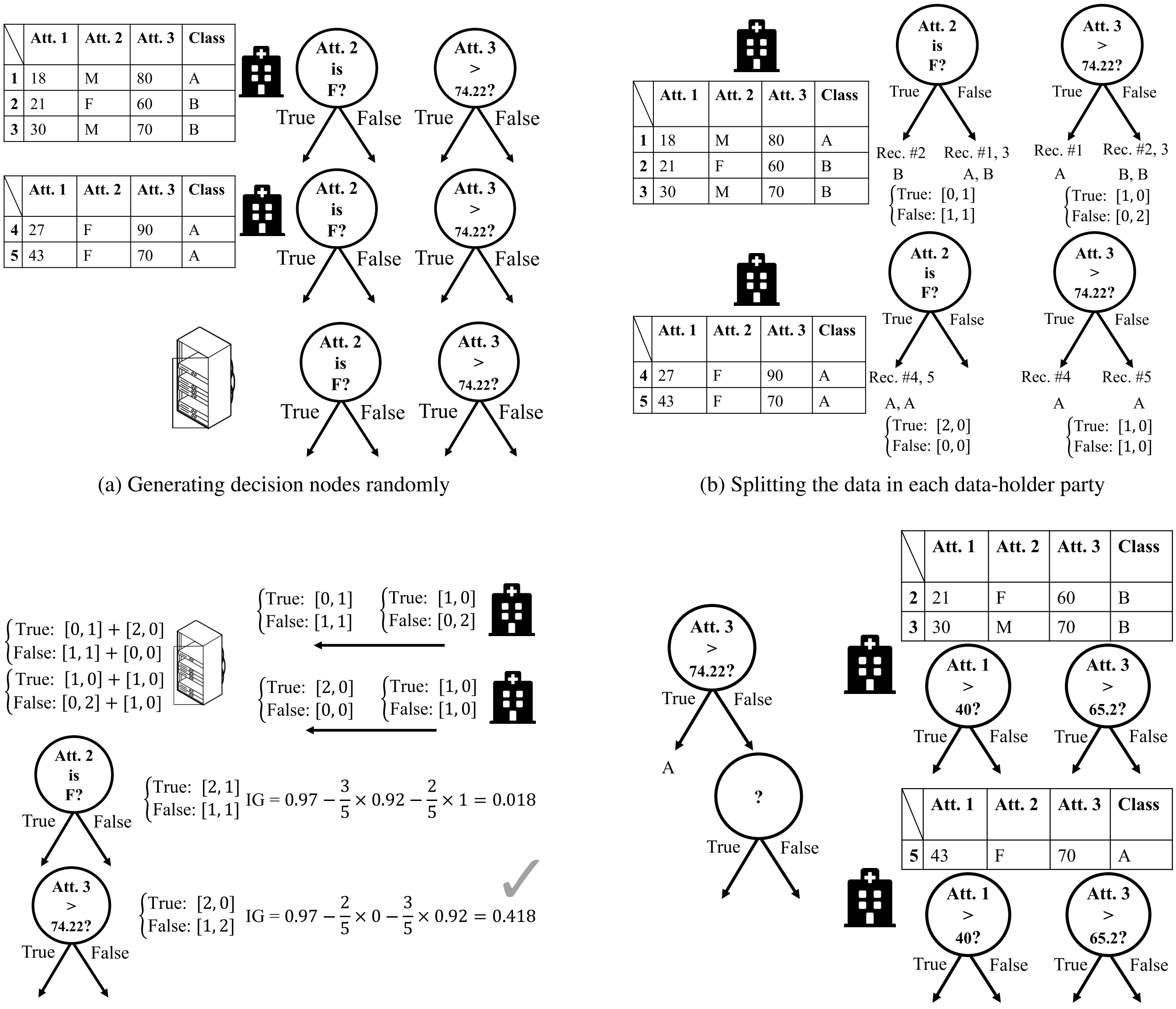
V. ILLUSTRATIVE EXAMPLE

In this section, we provide an illustrative example to clarify the procedure of learning for our algorithm. This procedure is shown in Figure 4. For the sake of simplicity of the presentation, we do not consider the secure aggregation in this section. In the learning process initiation, the global random seed, secure aggregation's random seeds, number and type of data attributes, possible categories or range of data attributes, and learning parameters for the algorithm are shared among all parties. In our example, we have two data-holder parties and a mediator. The first and second parties hold three and two training data records, respectively, as shown in Figure 4a. Each record has three attributes (two numerical and one categorical) and one label.

The goal is to learn an ensemble of decision trees from all the records available on the data-holder parties based on our algorithm. The mediator initiates a round of learning a decision tree and, after finishing the procedure for learning one tree, repeats it to have an ensemble of decision trees. At every step of choosing a decision node for the decision tree, each party, including the mediator, generates two random decision nodes based on the global seed. Since all parties use the same seed, they locally generate candidate decision nodes that are similar to the generated decision nodes in other parties. Figure 4a shows the local generation of the candidate decision nodes for the first decision tree's root.

In the next step, the parties classify their records using each randomly generated candidate decision node, as shown in Figure 4b. Several data records fall under the *True* branch (for each candidate decision node) and several fall under the *False* branch. Therefore, based on the records' labels (classes), we make two vectors for each branch that represent the combination of the labels. For instance, for the first candidate decision node in the first party: the *True* vector is [0, 1], and it means that zero records of this party belonging to class (label) A, and one record of this party belonging to class (label) B fall under the *True* branch of this candidate decision node. Thus, each data-holder party, for each candidate decision node, generates two vectors representing the combination of records labels (that fall under *True* and *False* branches).

The resulting vectors for each candidate decision node and in all data-holder parties should be returned to the mediator and, then, be aggregated there. Figure 4c shows this procedure, in which all vectors for the *True* and *False* branches of each candidate decision node are returned to the mediator. At this point, for each candidate decision node, the mediator has the combination of labels for the *True* and *False* branches. In addition to deciding on making a leaf or decision node in the decision tree's current position, such vectors determine



	Att. 1	Att. 2	Att. 3	Class
1	18	M	80	A
2	21	F	60	B
3	30	M	70	B

	Att. 1	Att. 2	Att. 3	Class
4	27	F	90	A
5	43	F	70	A

(a) Generating decision nodes randomly

(b) Splitting the data in each data-holder party

(c) Aggregating the results, calculating the scores

	Att. 1	Att. 2	Att. 3	Class
2	21	F	60	B
3	30	M	70	B

	Att. 1	Att. 2	Att. 3	Class
5	43	F	70	A

(d) Continuing the same process for the rest of the tree

FIGURE 4. **Illustrative example.**

which candidate decision node has a higher score/information gain and should be selected. For calculating the score (information gain) for a decision node, the combination of labels at each branch is required. In our example, the second decision node has a higher information gain and is selected.

As shown in Figure 4d, the second candidate decision node is selected for the root of the decision tree. After checking the labels in its *True* branch, [2, 0], we observe that all the records falling in the *True* branch belong to the same class (have the same label: *A*). Therefore, instead of making a decision node, we make a leaf in the *True* branch. We follow the same procedure of making a decision node for the *False* branch. However, this time, the data-holder parties only consider the records that fall in the root's *False* branch, i.e., 2, 3, and 5. We continue the same procedure for the rest of the tree.

VI. EVALUATION AND DISCUSSION

In this section, we evaluate our proposed approach with respect to classification performance, scalability and overhead, and privacy criteria [83].

A. DATA

We consider two sets of data for the evaluation in this paper. First, we consider two popular publicly available healthcare datasets, i.e., Heart Disease [29] and Breast Cancer Wisconsin (Diagnostic) [30]. For the Heart Disease case, we utilize the processed Cleveland's data [84] to predict the presence or absence of heart disease. In the other case, Wisconsin Diagnostic Breast Cancer (WDBC) data [84] is used to predict breast cancer's diagnosis as benign or malignant.

In addition to the above publicly available datasets, we also consider two mental health detests associated with the Norwegian INTROMAT (INTROducing Mental health through Adaptive Technology) project:

- The Depresjon dataset [85] contains motor activity data from 55 individuals (30 females and 25 males) recorded using an ActiGraph wristband worn on the right wrist. 23 individuals in this dataset have been diagnosed with depression, including both unipolar and bipolar individuals, while the remaining 32 are in the control group. Each individual wore an ActiGraph wristband for an arbitrary number of days, ranging from 5 to 20 days. The condition and control groups were monitored for 291 and 402 days in total, respectively.
- The Psykose dataset [86] contains motor activity data from 54 individuals (23 females and 31 males) recorded using an ActiGraph wristband worn on the right wrist. 22 individuals in this dataset have been diagnosed with schizophrenia, and all used antipsychotic medications, while the remaining 32 are in the control group. Each individual wore an ActiGraph wristband for an arbitrary number of days, ranging from 8 to 20 days. The condition and control groups were monitored for 285 and 402 days in total, respectively.

B. PERFORMANCE EVALUATION METRICS

The performance of the proposed algorithm is evaluated by measuring the F1-score ($F1$), Accuracy (ACC), and Matthews Correlation Coefficient (MCC), which are defined as follows:

$$
\begin{aligned}
F1 &= \frac{TP}{TP+0.5\cdot(FP+FN)}\\
ACC &= \frac{TP+TN}{TP+FP+TN+FN}\\
MCC &= \frac{TP\cdot TN - FP\cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}
\end{aligned}
$$

where FP, TN, TP and FN definitions are the false positive, true negative, true positive, and false negative, respectively.

C. EVALUATION AND RESULTS

1) CLASSIFICATION PERFORMANCE FOR WIDELY USED HEALTHCARE DATASETS

To evaluate the classification performance for Heart Disease [29] and Breast Cancer Wisconsin (Diagnostic) [30] datasets, we perform a three-fold cross-validation. We divide the dataset into three parts, and in each round, we use one of the parts as the test set and the rest as the training set and finally report the averaged results. We adopt the F1-score (weighted average) and accuracy as our classification performance metrics. The F1-score is the harmonic mean between the precision and recall metrics, while the accuracy measures the ratio of correctly classified samples. Table 2 exhibits the classification performance of our approach, k-PPD-ERT, against the distributed ID3 algorithm [12]. We compare our approach against the distributed ID3 [12] since, similar to our approach, it is a state-of-the-art tree-based method that employs SMC techniques for secure aggregation of partial results and addresses classification problems in scenarios where the data is horizontally partitioned. Moreover, the classification performance of the centralized version of ERT is also provided for comparison.

TABLE 2. Classification performance for our proposed method, distributed ID3, and centralized ERT.

Dataset	Metric	k-PPD-ERT	Distributed ID3	ERT
Heart Disease [29]	Accuracy	80.4%	74.5%	80.4%
	F1-Score	80%	74.3%	80%
Breast Cancer [30]	Accuracy	95.3%	91.3%	95.3%
	F1-Score	95.4%	91.3%	95.4%

The k-PPD-ERT and ERT algorithms follow the same learning procedure. This means that, for both algorithms, the same steps for selecting candidate decision nodes and building the decision tree are followed. In our experiments, we set the same seeds for the random functions and the same learning parameters for both algorithms, e.g., the number of candidate decision nodes. Moreover, the datasets are split into train and test sets in the same way with the same random seed, so these sets are the same for both experiments. Therefore, both algorithms result in the same classification performance, i.e., by following the same procedure, setting the same seeds and parameters, and having the same train and test data.

In our experiments, for our approach, k-PPD-ERT, and the ERT algorithm, we learn an ensemble of 25 decision trees. For the number of candidate decision nodes' parameter in the algorithm, we use 5-fold cross-validation on the training set for the model selection (concerning classification performance measured by the F1-score). In the case of the Heart Disease dataset, k-PPD-ERT outperforms the distributed ID3 [12] by up to 5.9%. For the Breast Cancer dataset, our approach outperforms the distributed ID3 by up to 4.1%.

2) CLASSIFICATION PERFORMANCE FOR MENTAL HEALTH DATASETS ASSOCIATED WITH INTROMAT PROJECT

In addition to the widely used public datasets, we also consider the data associated with the Norwegian INTROMAT (INTROducing Mental health through Adaptive Technology) project, i.e., Depresjon dataset [85] and Psykose dataset [86]. We use F1-score (weighted average), Accuracy (ACC), and Matthews Correlation Coefficient (MCC) for measuring the classification performance, which are the metrics used for performance evaluation on these datasets [85], [86]. We consider both the original and augmented data for each dataset. The original data includes the mean and the standard deviation of the activity level along with the proportion of minutes with no activity [85], [86]. The augmented sample reflects the activity level of an individual in a day by locally resampling the raw data from the same individual. The problem related to the difference in the number of recorded days for each individual, which makes the dataset more imbalanced, is addressed by augmentation. Augmentation also addresses

TABLE 3. Classification performance (leave one patient out) of different classification algorithms for mental health datasets associated with the Norwegian INTROMAT project, i.e., Depresjon dataset [85] and Psykose dataset [86].

Algorithms	Depresjon Dataset [85]						Psykose Dataset [86]					
	Augmented Data			Without Augmentation			Augmented Data			Without Augmentation		
	F1-score	ACC	MCC	F1-score	ACC	MCC	F1-score	ACC	MCC	F1-score	ACC	MCC
k-PPD-ERT (Distributed)	76.3%	76.8%	0.518	66.3%	67.0%	0.310	87.9%	88.0%	0.751	81.7%	81.8%	0.623
ID3 (Distributed)	65.1%	65.0%	0.286	65.6%	66.5%	0.296	75.0%	74.8%	0.490	79.3%	79.4%	0.573
ERT (Centralized)	76.3%	76.8%	0.518	66.3%	67.0%	0.310	87.9%	88.0%	0.751	81.7%	81.8%	0.623
Random forest (Centralized)	74.4%	75.1%	0.481	64.3%	64.7%	0.266	90.7%	90.7%	0.807	80.6%	80.7%	0.601
XGBoost (Centralized)	76.2%	76.3%	0.510	64.3%	64.7%	0.265	92.4%	92.5%	0.844	80.7%	80.7%	0.601
Decision Tree (Centralized)	65.7%	65.8%	0.293	60.6%	60.7%	0.191	76.0%	76.0%	0.505	76.1%	76.2%	0.508
Linear SVM (Centralized)	69.5%	69.5%	0.375	68.4%	68.6%	0.349	87.3%	87.2%	0.748	82.8%	82.8%	0.645

TABLE 4. Communication complexity and privacy of different SMC approaches.

Approach	Party	Communication		Total Communication	Number of Colluding Parties
		Send	Receive	(N = number of parties)	
NOSMC	Data Holders Mediator	1 0	0 $N-1$	$(N-1)\times 1 + 1\times(N-1)$	1: mediator has the values with no collusion
STSMC	All	2	2	$N\times(2+2)$	2: neighbor parties
k-PPD-ERT	Data Holders Mediator	1 0	0 $N-1$	$(N-1)\times 1 + 1\times(N-1)$	$k+1$: k data-holder parties and the mediator
Shamir [75]	$k-1$ Parties One Party The Rest	N $N-1$ $N-1$	$N-1$ $N-1+k-1$ $N-1$	$N\times(N-1+N-1)+2\times(k-1)$	k parties (k<N)

the problem of samples with a shorter length, i.e., motor activity signals recorded starting from the middle of the day [87].

We compare our approach against several state-of-the-art machine learning algorithms, including ERT [77], random forest [73], XGBoost [88], Decision Tree [64], and linear SVM algorithm [89]. Table 3 shows the classification performance of different algorithms for the INTROMAT data. The results demonstrate that the proposed approach performs on par or better than state-of-the-art techniques. We also compare our approach against the distributed ID3 [12]. For the Depresjon dataset [85], the k-PPD-ERT technique outperforms distributed ID3 [12] by 0.7% in terms of F1-score, 0.5% in terms of ACC, and 0.014 in terms of MCC for the original data and by 11.2% in terms of F1-score, 11.8% in terms of ACC, and 0.232 in terms of MCC for the augmented data. For the Psykose dataset [86], the k-PPD-ERT technique outperforms distributed ID3 [12] by 2.4% in terms of F1-score, 2.4% in terms of ACC, and 0.05 in terms of MCC for the original data and by 12.9% in terms of F1-score, 13.2% in terms of ACC, and 0.261 in terms of MCC for the augmented data.

3) PRIVACY AND OVERHEAD OF SECURE MULTI-PARTY COMPUTATION TECHNIQUES

We now discuss the privacy and overhead of our proposed approach. We adopt an SMC technique to avoid direct sharing of the vectors representing the combination of record labels for each candidate decision node with other parties and the mediator. We compare the communication overhead and privacy of our adopted SMC technique against three other techniques, including the SMC methods employed in [12], i.e., Shamir's technique [75]. Table 4 presents the communication overhead and privacy evaluation of each approach. In the table, N is the number of parties, and k is a parameter in k-PPD-ERT and Shamir's secret sharing for the minimum number of colluding parties to identify a secret value. The communication overheads in the table are for one round of secure aggregation.

In the first approach (NOSMC), no SMC technique is adopted, and all values are directly shared with the mediator and known to it. This approach has the lowest possible communication cost and number of colluding parties, and, here, it is considered as a baseline. The other approach for the aggregation of partial results is the straightforward SMC (STSMC) approach. In this approach, in the first round, each party aggregates its random mask and its secret value to the received result from the previous party and passes it to the next party, and in the second round, parties subtract their random masks from the aggregated result of the previous round. This method's communication overhead is of the same order as NOSMC, $O(N)$, but it is more robust to collusion. On the other hand, Shamir's secret sharing is an SMC method employed in [12] for secure aggregation. This approach can tolerate the highest number of colluding parties, although it has a high communication overhead, i.e., $O(N^2)$.

Our approach's communication overhead, similar to NOSMC and STSMC, is from order $O(N)$, which is considerably more efficient compared to Shamir's approach with an order of $O(N^2)$. Concerning the number of colluding parties, by adopting our approach, it takes k ($k < N$) data-holder parties and the mediator to collude for identification of the secret values. In our approach, the participation of the mediator for

TABLE 5. The scenarios for our experiments on Amazon's AWS cloud.

	Number of data holders	Mediator location	Data holders locations
Scenario 1	2	SE	CA,DE
Scenario 2	5	SE	CA,DE,US,JP,AU
Scenario 3	10	SE	CA,DE,US,JP,AU,SG,IN,KR,FR,EN
Scenario 4	20	SE	CA,DE,US,JP,AU,SG,IN,KR,FR,EN

collusion is required to reveal a secret value. The mediator is assumed as an honest party in many scenarios, and in the case of a secret value revelation, we know that the mediator has been involved in the collusion. Shamir's secret sharing requires k ($k < N$) parties to collude for identifying a secret value but suffers from scalability and high communication overhead.

4) LATENCY FOR OUR PROOF-OF-CONCEPT IMPLEMENTATION

Finally, we have also implemented our proposed approach on Amazon's AWS cloud to evaluate the latency and scalability of the k-PPD-ERT.[1] We consider four scenarios where we change the number of data-holder parties. We consider four datasets, i.e., Heart [29], Breast [30], Depresjon [85], Psykose [86]. For each dataset, the training data (75% of the dataset) is distributed equally among the data-holder parties. The mediator includes a 2 core 2.40 GHz CPU and 512 MB RAM, runs Ubuntu 20.04, and is located in Sweden. The machines in all other locations include a 1 core 2.40 GHz CPU and 512 MB RAM and run Ubuntu 20.04.

The latency results are shown in Figure 5. In the first scenario, as shown in Table 5, we consider two data-holder parties located in Canada and Germany. Learning one extremely randomized tree through our approach takes 15.9 ± 1.5, 11.8 ± 3.5, 3.5 ± 1.0, 2.4 ± 0.7 seconds for Heart, Breast, Depresjon, and Psykose datasets, respectively. In the second scenario, as shown in Table 5, we consider five data-holder parties located in Canada, Germany, the United States, Japan, and Australia. Learning one extremely randomized tree through our approach takes 43.5 ± 4.1, 32.4 ± 9.6, 9.5 ± 2.7, 6.6 ± 2.0 seconds for Heart, Breast, Depresjon, and Psykose datasets, respectively. In the third scenario, as shown in Table 5, we consider ten data-holder parties located in Canada, Germany, the United States, Japan, Australia, Singapore, India, South Korea, France, and England. Learning one extremely randomized tree through our approach takes 43.8 ± 4.2, 32.6 ± 9.7, 9.6 ± 2.7, 6.7 ± 2.0 seconds for Heart, Breast, Depresjon, and Psykose datasets, respectively. In the fourth scenario, as shown in Table 5, we consider twenty data-holder parties located in Canada, Germany, the United States, Japan, Australia, Singapore, India, South Korea, France, and England, with two parties at each location. Learning one extremely randomized tree through our approach takes 43.6 ± 4.1, 32.5 ± 9.7, 9.6 ± 2.7, 6.8 ± 2.0 seconds for Heart, Breast, Depresjon, and Psykose datasets, respectively.

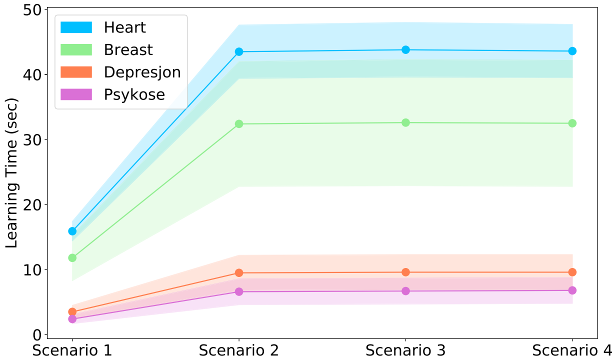


FIGURE 5. The mean and standard deviation of learning time (ten times performed) of one extremely randomized tree through *k*-PPD-ERT for different datasets in several scenarios on Amazon's AWS cloud.

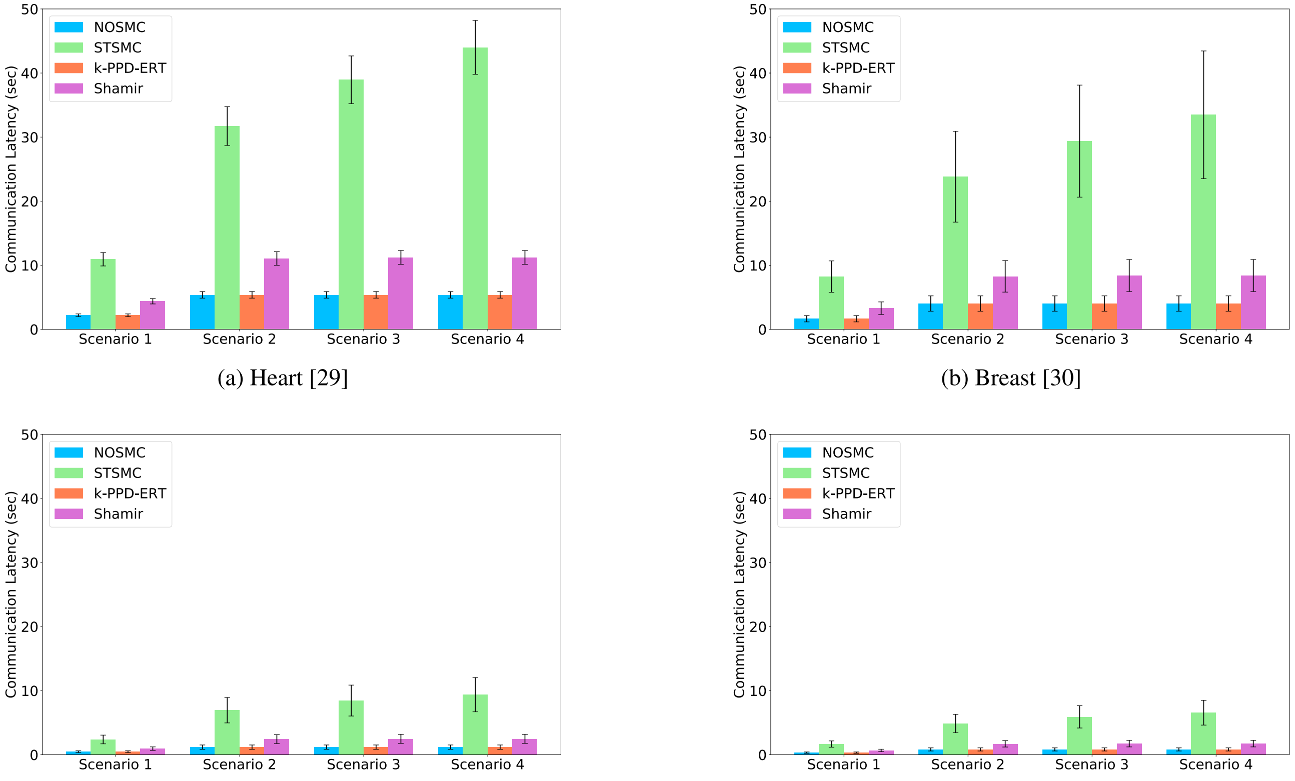
To better understand the reason for the increase and decrease in the latencies reported above and the shape of the graphs in Figure 5, it should be noted that the latency depends on the geographical location of the data holders and communication delays. In scenario two, the latency has increased due to the fact that the bottleneck communication distance between the data holders and the mediator is increased. However, the results in scenario three are similar to scenario two because the bottleneck communication distance remains the same. In scenario four, the slight reduction in the latency is due to the fact that we distribute the data among data-holder parties (each party has fewer data samples to process), and the learning process on each party is performed simultaneously and in parallel, similar to big data analysis. These explain the increase of latencies from scenario one to two and the almost flat shapes of the graphs from scenario two to scenario four in Figure 5.

5) COMMUNICATION LATENCY OF SECURE MULTI-PARTY COMPUTATION TECHNIQUES

We also evaluate the communication latency of one secure aggregation round for each SMC approach based on their algorithms, the location of data holders in each scenario, the volume of packets transferred between parties, and the network bandwidth between parties. This shows to what extent adopting each approach can increase the latency.

In this paper, we consider the propagation and transmission delays for communication latency [90], [91]. The latency of transferring a packet from P_i to P_j is equal to the sum of propagation and transmission delays and is denoted by $L(P_i, P_j)$. The propagation delay is equal to the distance between parties divided by the velocity of signal propagation, which for unguided transmission through air or space is equal to the speed of light [90]. The transmission delay is equal to the number of bits in the packet divided by the rate of transmission. For transmission delay, we divide the volume of the message to be transferred from P_i to P_j by the bandwidth between these parties.

[1] The source code of our implementations is available at https://github.com/AminAminifar/kPPDERT_cloud



(a) Heart [29]

(b) Breast [30]

(c) Depresjon [85]

(d) Psykose [86]

FIGURE 6. The mean and standard deviation of estimated communication latency of different methods for aggregation of secret values in learning one extremely randomized tree (ten times performed) based on different datasets in several scenarios on Amazon's AWS cloud.

The network bandwidth between two Amazon machines is measured as 1.05 Mbits/sec using the iPerf tool [92]. When a packet contains two arrays for true and false branches, each including information for five candidate decision nodes for a binary classification task, the volume of each packet is 384 bytes. The volume of the packet depends on the data, i.e., the number of candidate decision nodes and the number of target classes.

The following are the analysis of communication latency for each method:

- For NOSMC and k-PPD-ERT, all parties (P_i, $\forall i \in \{1, \ldots, n\}$) send one message to the mediator (M) in parallel. Since the messages are sent in parallel, the communication latency is equal to the arrival duration of the last message. Therefore, the communication delay is equal to $\max_i L(P_i, M)$, $i \in \{1, \ldots, n\}$.
- For STSMC, we have two loops of message passing between parties in each round, and finally, the first party sends the result to the mediator. Therefore, the communication delay is equal to $2 \cdot (\sum_{i=1}^{n-1} L(P_i, P_{i+1}) + L(P_n, P_1)) + L(P_1, M)$.
- For Shamir, each round of secure aggregation consists of two parts performed sequentially. In the first part, all data-holder parties send one message to $n - 1$ parties. When all parties receive these messages, they calculate the intermediate results [12] and send them to the mediator. Therefore, the communication delay is equal to $\max_{i,j} L(P_i, P_j)$, $i, j \in \{i, j \in \{1, \ldots, n\} \mid i \neq j\}$ plus $\max_i L(P_i, M)$, $i \in \{1, \ldots, n\}$.

The number of required secure aggregation operations is also recorded for the experiments in Section VI-C4. The mean and standard deviation of the required number of secure aggregation operations for learning one extremely randomized tree (ten times performed) are 98.8 ± 9.4, 73.6 ± 21.9, 22.0 ± 6.2, 15.4 ± 4.5 operations for Heart, Breast, Depresjon, and Psykose datasets, respectively. For estimating the total communication latency of each method for aggregating secret values, the calculated latencies should be multiplied by the number of secure aggregations performed for learning the classification model.

Figure 6 shows the mean and standard deviation of communication latency of different methods for aggregation of secret values for each scenario and each dataset. This figure shows that k-PPD-ERT has the same communication latency as the NOSMC procedure. Shamir's technique has lower communication latency compared to STSMC, but it still has higher communication latency compared to k-PPD-ERT and NOSMC procedures.

It should be noted that the communication latency of these methods should not be confused with the communication overhead presented in Table 4. The orders of communication overhead for NOSMC, STSMC, and k-PPD-ERT are the same and lower than Shamir's technique. However, since in STSMC, we have two loops of message passing between parties that are performed sequentially, this technique has more delay for a secure aggregation operation. Shamir's technique has two rounds for each SMC operation, and in each round, the message passings are performed in parallel, so it has a lower delay compared to STSMC. For NOSMC and k-PPD-ERT, we have one round of message passing that is performed in parallel and has the lowest communication latency.

Finally, we demonstrate that our proposed k-PPD-ERT approach provides a solution for the classification of structured data distributed over multiple sources with privacy-preservation considerations, without performance degradation.

VII. CONCLUSION

In this paper, we present the privacy-preserving distributed extremely randomized trees algorithm for learning without privacy concerns in the healthcare domain. We have evaluated our proposed algorithm extensively using two popular structured healthcare datasets and two mental health datasets associated with the Norwegian INTROducing Mental health through Adaptive Technology (INTROMAT) project. Our approach outperforms the state of the art in distributed tree-based models by up to 11.2% in terms of F1-score, 11.8% in terms of ACC, and 0.232 in terms of MCC for the Depresjon augmented dataset, and by up to 12.9% in terms of F1-score, 13.2% in terms of ACC, and 0.261 in terms of MCC for the Psykose augmented dataset. Moreover, we present the implementation of our technique on Amazon's AWS cloud, as a proof of concept, to evaluate the latency and scalability of our framework. The proposed algorithm has linear overhead with respect to the number of parties and can also handle datasets with missing values. We demonstrated our framework's efficiency in terms of prediction performance, scalability, and overheads, as well as privacy. The proposed framework provides the possibility of developing high-quality and accurate machine learning models without privacy concerns and is expected to contribute to a better healthcare system in the long term. As future work, we plan to explore the possibility of extending the proposed framework to settings where the parties do not follow the honest-but-curious security model, which is beyond the scope of this work.

REFERENCES

[1] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng, "Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network," *Nature Med.*, vol. 25, no. 1, pp. 65–69, Jan. 2019.

[2] S. McKinney *et al.*, "International evaluation of an AI system for breast cancer screening," *Nature*, vol. 577, pp. 89–94, Jan. 2020.

[3] X. Liu, L. Faes, A. U. Kale, S. K. Wagner, D. J. Fu, A. Bruynseels, T. Mahendiran, G. Moraes, M. Shamdas, C. Kern, J. R. Ledsam, M. K. Schmid, K. Balaskas, E. J. Topol, L. M. Bachmann, P. A. Keane, and A. K. Denniston, "A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: A systematic review and meta-analysis," *Lancet Digit. Health*, vol. 1, no. 6, pp. e271–e297, Oct. 2019.

[4] R. Aggarwal, V. Sounderajah, G. Martin, D. S. W. Ting, A. Karthikesalingam, D. King, H. Ashrafian, and A. Darzi, "Diagnostic accuracy of deep learning in medical imaging: A systematic review and meta-analysis," *npj Digit. Med.*, vol. 4, no. 1, p. 65, Dec. 2021.

[5] S. D. Lustgarten, Y. L. Garrison, M. T. Sinnard, and A. W. Flynn, "Digital privacy in mental healthcare: Current issues and recommendations for technology use," *Current Opinion Psychol.*, vol. 36, pp. 25–31, Dec. 2020.

[6] D. Pascual, A. Amirshahi, A. Aminifar, D. Atienza, P. Ryvlin, and R. Wattenhofer, "EpilepsyGAN: Synthetic epileptic brain activities with privacy preservation," *IEEE Trans. Biomed. Eng.*, vol. 68, no. 8, pp. 2435–2446, Aug. 2021.

[7] A. Saeed, F. D. Salim, T. Ozcelebi, and J. Lukkien, "Federated self-supervised learning of multisensor representations for embedded intelligence," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1030–1040, Jan. 2021.

[8] F. Forooghifar, A. Aminifar, and D. Atienza, "Resource-aware distributed epilepsy monitoring using self-awareness from edge to cloud," *IEEE Trans. Biomed. Circuits Syst.*, vol. 13, no. 6, pp. 1338–1350, Dec. 2019.

[9] D. Sopic, A. Aminifar, A. Aminifar, and D. Atienza, "Real-time event-driven classification technique for early detection and prevention of myocardial infarction on wearable systems," *IEEE Trans. Biomed. Circuits Syst.*, vol. 12, no. 5, pp. 982–992, Oct. 2018.

[10] D. Sopic, A. Aminifar, A. Aminifar, and D. Atienza, "Real-time classification technique for early detection and prevention of myocardial infarction on wearable devices," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, Oct. 2017, pp. 1–4.

[11] R. Zanetti, A. Arza, A. Aminifar and D. Atienza, "Real-time EEG-based cognitive workload monitoring on wearable devices," *IEEE Trans. Biomed. Eng.*, vol. 69, no. 1, pp. 265–277, Jan. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9464276, doi: 10.1109/TBME.2021.3092206.

[12] F. Emekci, O. D. Sahin, D. Agrawal, and A. El Abbadi, "Privacy preserving decision tree learning over multiple parties," *Data Knowl. Eng.*, vol. 63, no. 2, pp. 348–361, Nov. 2007.

[13] J. S. Davis and O. Osoba, "Improving privacy preservation policy in the modern information age," *Health Technol.*, vol. 9, no. 1, pp. 65–75, Jan. 2019.

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

[15] P. Jurczyk and L. Xiong, "Distributed anonymization: Achieving privacy for both data subjects and data providers," in *Proc. IFIP Annu. Conf. Data Appl. Secur. Privacy*. Berlin, Germany: Springer, 2009. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-03007-9_13

[16] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.

[17] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond K-anonymity," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, p. 3, 2007.

[18] N. Li, T. Li, and S. Venkatasubramanian, "T-closeness: Privacy beyond K-anonymity and L-diversity," in *Proc. IEEE 23rd Int. Conf. Data Eng.*, Apr. 2007, pp. 106–115.

[19] A. Aminifar, Y. Lamo, K. Pun, and F. Rabbi, "A practical methodology for anonymization of structured health data," in *Proc. 17th Scandin. Conf. Health Informat.*, 2019, pp. 127–133. [Online]. Available: https://ep.liu.se/en/conference-article.aspx?series=ecp&issue=161&Article_No=22

[20] A. Aminifar, F. Rabbi, V. K. I. Pun, and Y. Lamo, "Diversity-aware anonymization for structured health data," in *Proc. 43rd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Nov. 2021, pp. 2148–2154.

[21] *Health Informatics—Pseudonymization*, International Organization for Standardization, Geneva, Switzerland, Standard ISO 25237:2017, Jan. 2017. [Online]. Available: https://www.iso.org/standard/63553.html

[22] C. Dwork, "Differential privacy," in *Proc. 33rd Int. Colloq. Automata, Lang., Program. (ICALP)* (Lecture Notes in Computer Science). Berlin, Germany: Springer-Verlag, 2006. [Online]. Available: https://link.springer.com/chapter/10.1007/11787006_1

[23] M. Kantarcioglu, "A survey of privacy-preserving methods across horizontally partitioned data," in *Privacy-Preserving Data Mining*. Boston, MA, USA: Springer, 2008, pp. 313–335. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-387-70992-5_13

[24] J. Vaidya, "A survey of privacy-preserving methods across vertically partitioned data," in *Privacy-Preserving Data Mining*. Boston, MA, USA: Springer, 2008, pp. 337–358. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-387-70992-5_14

[25] W. Du and Z. Zhan, "Building decision tree classifier on private data," in *Proc. IEEE Int. Conf. Privacy, Secur. Data Mining (CRPIT)*, vol. 14. Australia: Austral. Comput. Soc., 2002, pp. 1–8. [Online]. Available: https://dl.acm.org/doi/10.5555/850782.850784

[26] J. Konečný, H. Brendan McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*.

[27] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.

[28] A. Aminifar, F. Rabbi, and Y. Lamo, "Scalable privacy-preserving distributed extremely randomized trees for structured data with multiple colluding parties," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 2655–2659.

[29] R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, J.-J. Schmid, S. Sandhu, K. H. Guppy, S. Lee, and V. Froelicher, "International application of a new probability algorithm for the diagnosis of coronary artery disease," *Amer. J. Cardiol.*, vol. 64, no. 5, pp. 304–310, Aug. 1989.

[30] O. L. Mangasarian, W. N. Street, and W. H. Wolberg, "Breast cancer diagnosis and prognosis via linear programming," *Oper. Res.*, vol. 43, no. 4, pp. 570–577, Aug. 1995.

[31] *INTROMAT (Introducing Personalized Treatment of Mental Health Problems Using Adaptive Technology)*. Accessed: Dec. 9, 2021. [Online]. Available: https://intromat.no

[32] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist., Mach. Learn. Res.*, PMLR, 2017. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.htm

[33] V. Smith, S. Forte, C. Ma, M. Takáč, M. Jordan, and M. Jaggi, "CoCoA: A general framework for communication-efficient distributed optimization," *J. Mach. Learn. Res.*, vol. 18, p. 230, Apr. 2017.

[34] S. Baghersalimi, T. Teijeiro, D. Atienza, and A. Aminifar, "Personalized real-time federated learning for epileptic seizure detection," *IEEE J. Biomed. Health Informat.*, early access, Jul. 9, 2021, doi: 10.1109/JBHI.2021.3096127.

[35] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2000, pp. 439–450.

[36] S. Agrawal and J. R. Haritsa, "A framework for high-accuracy privacy-preserving mining," in *Proc. 21st Int. Conf. Data Eng. (ICDE)*, Apr. 2005, pp. 193–204.

[37] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, "Privacy preserving mining of association rules," *Inf. Syst.*, vol. 29, no. 4, pp. 343–364, Jun. 2004.

[38] S. Rizvi and J. Haritsa, "Maintaining data privacy in association rule mining," in *Proc. 28th Int. Conf. Very Large Databases (VLDB)*. Amsterdam, The Netherlands: Elsevier, 2002, pp. 682–693.

[39] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *Proc. 3rd IEEE Int. Conf. Data Mining*, Nov. 2003, pp. 99–106.

[40] Z. Huang, W. Du, and B. Chen, "Deriving private information from randomized data," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2005, pp. 37–48.

[41] M. Kantarcioglu and J. Vaidya, "An architecture for privacy-preserving mining of client information," in *Proc. IEEE Int. Conf. Privacy, Secur. Data Mining*, vol. 14, Dec. 2002, pp. 37–42.

[42] Y. Lindell and B. Pinkas, "Privacy preserving data mining," *J. Cryptol.*, vol. 15, no. 3, pp. 177–206, 2002.

[43] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *ACM SIGKDD Explor. Newslett.*, vol. 4, no. 2, pp. 28–34, Dec. 2002.

[44] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1333–1345, May 2018.

[45] J. Vaidya and C. Clifton, "Privacy-preserving outlier detection," in *Proc. 4th IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2004, pp. 233–240.

[46] G. Jagannathan and R. N. Wright, "Privacy-preserving distributed K-means clustering over arbitrarily partitioned data," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2005, pp. 593–599.

[47] X. Lin, C. Clifton, and M. Zhu, "Privacy-preserving clustering with distributed EM mixture modeling," *Knowl. Inf. Syst.*, vol. 8, no. 1, pp. 68–81, Jul. 2005.

[48] H. Yu, X. Jiang, and J. Vaidya, "Privacy-preserving SVM using nonlinear kernels on horizontally partitioned data," in *Proc. ACM Symp. Appl. Comput. (SAC)*, 2006, pp. 603–610.

[49] B. Pinkas, "Cryptographic techniques for privacy-preserving data mining," *ACM SIGKDD Explor. Newslett.*, vol. 4, no. 2, pp. 12–19, Dec. 2002.

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

[51] P. Kairouz *et al.*, "Advances and open problems in federated learning," 2019, *arXiv:1912.04977*.

[52] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," 2019, *arXiv:1907.09693*.

[53] S. Lo, Q. Lu, C. Wang, H. Paik, and L. Zhu, "A systematic literature review on federated machine learning: From a software engineering perspective," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–39, 2021.

[54] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2019, pp. 739–753.

[55] S. Truex, L. Liu, M. E. Gursoy, L. Yu, and W. Wei, "Demystifying membership inference attacks in machine learning as a service," *IEEE Trans. Services Comput.*, vol. 14, no. 6, pp. 2073–2089, Nov. 2021.

[56] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 3454–3469, 2020.

[57] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9530–9539, Oct. 2020.

[58] R. C. Geyer, T. Klein, and M. Nabi, "Differentially private federated learning: A client level perspective," 2017, *arXiv:1712.07557*.

[59] C. Zhuang, T. She, A. Andonian, M. Sobol Mark, and D. Yamins, "Unsupervised learning from video with deep neural embeddings," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9563–9572.

[60] H. Purwins, B. Li, T. Virtanen, J. Schluter, S.-Y. Chang, and T. Sainath, "Deep learning for audio signal processing," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 2, pp. 206–219, May 2019.

[61] A. M. Vartouni, M. Shokri, and M. Teshnehlab, "Auto-threshold deep SVDD for anomaly-based web application firewall," *TechRxiv*, 2021. [Online]. Available: https://www.techrxiv.org/articles/preprint/Auto-Threshold_Deep_SVDD_for_Anomaly-based_Web_Application_Firewall/15135468, doi: 10.36227/techrxiv.15135468.v1.

[62] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable AI for trees," *Nature Mach. Intell.*, vol. 2, no. 1, pp. 56–67, Jan. 2020.

[63] W. Fan, H. Wang, P. S. Yu, and S. Ma, "Is random model better? On its accuracy and efficiency," in *Proc. 3rd IEEE Int. Conf. Data Mining*, Nov. 2003, pp. 51–58.

[64] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.

[65] W. Du and Z. Zhan, "Using randomized response techniques for privacy-preserving data mining," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2003, pp. 505–510.

[66] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, D. Papadopoulos, and Q. Yang, "SecureBoost: A lossless federated learning framework," *IEEE Intell. Syst.*, vol. 36, no. 6, pp. 87–98, Nov. 2021.

[67] Y. Liu, Z. Ma, X. Liu, S. Ma, S. Nepal, R. H. Deng, and K. Ren, "Boosting privately: Federated extreme gradient boosting for mobile crowdsensing," in *Proc. IEEE 40th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Nov. 2020, pp. 1–11.

[68] L. Zhao, L. Ni, S. Hu, Y. Chen, P. Zhou, F. Xiao, and L. Wu, "InPrivate digging: Enabling tree-based distributed data mining with differential privacy," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 2087–2095.

[69] Q. Li, Z. Wen, and B. He, "Practical federated gradient boosting decision trees," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 4642–4649.

[70] J. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.

[71] Y. Liu, Y. Liu, Z. Liu, Y. Liang, C. Meng, J. Zhang, and Y. Zheng, "Federated forest," *IEEE Trans. Big Data*, early access, May 7, 2020, doi: 10.1109/TBDATA.2020.2992755.

[72] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proc. 12th ACM Workshop Artif. Intell. Secur. (AISec)*, 2019, pp. 1–11.

[73] T. Kam Ho, "Random decision forests," in *Proc. 3rd Int. Conf. Document Anal. Recognit.*, Aug. 1995, pp. 278–282.

[74] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, pp. 5–32, Oct. 2001.

[75] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[76] A. Aminifar, F. Rabbi, K. I. Pun, and Y. Lamo, "Privacy preserving distributed extremely randomized trees," in *Proc. 36th Annu. ACM Symp. Appl. Comput.*, Mar. 2021, pp. 1102–1105.

[77] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, Apr. 2006.

[78] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning* (Springer Series in Statistics). New York, NY, USA: Springer, 2001.

[79] Z. Lipton, "The mythos of model interpretability," *Queue*, 2018.

[80] N. D. Condorcet, *Essai Sur l'Application de l'Analyse à la Probabilité des Décisions Rendues à la Pluralité des Voix*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[81] L. Rokach, *Pattern Classification Using Ensemble Methods*. Singapore: World Scientific, 2010.

[82] A. C.-C. Yao, "How to generate and exchange secrets," in *Proc. 27th Annu. Symp. Found. Comput. Sci.*, Oct. 1986, pp. 162–167.

[83] E. Bertino, D. Lin, and W. Jiang, "A survey of quantification of privacy preserving data mining algorithms," in *Privacy-Preserving Data Mining*. Boston, MA, USA: Springer, 2008, pp. 183–205. [Online]. Available: https://link.springer.com/chapter/10.1007/978-0-387-70992-5_8

[84] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2019. [Online]. Available: http://archive.ics.uci.edu/ml and https://archive.ics.uci.edu/ml/citation_policy.html

[85] E. Garcia-Ceja, M. Riegler, P. Jakobsen, J. Tørresen, T. Nordgreen, K. Oedegaard, and O. Fasmer, "Depresjon: A motor activity database of depression episodes in unipolar and bipolar patients," in *Proc. 9th ACM Multimedia Syst. Conf.*, 2018, pp. 472–477.

[86] P. Jakobsen, E. Garcia-Ceja, L. A. Stabell, K. J. Oedegaard, J. O. Berle, V. Thambawita, S. A. Hicks, P. Halvorsen, O. B. Fasmer, and M. A. Riegler, "PSYKOSE: A motor activity database of patients with schizophrenia," in *Proc. IEEE 33rd Int. Symp. Computer-Based Med. Syst. (CBMS)*, Jul. 2020, pp. 303–308.

[87] A. Aminifar, F. Rabbi, K. Pun, and Y. Lamo, "Monitoring motor activity data for detecting patients' depression using data augmentation and privacy-preserving distributed learning," in *Proc. 43rd Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC)*, Nov. 2021, pp. 2163–2169.

[88] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 785–794.

[89] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, Sep. 1995.

[90] W. Stallings, *Data and Computer Communications*. Upper Saddle River, NJ, USA: Prentice-Hall, 2005.

[91] A. Pahlevan, "Multi-objective system-level management of modern green data centers," EPFL, Lausanne, Switzerland, Tech. Rep., 2019. [Online]. Available: https://infoscience.epfl.ch/record/270205?ln=en, doi: 10.5075/epfl-thesis-9457.

[92] *iPerf–The Ultimate Speed Test Tool for TCP, UDP and SCTP*. Accessed: Nov. 30, 2021. [Online]. Available: https://iperf.fr/

AMIN AMINIFAR received the M.Sc. degree in computer engineering from the K. N. Toosi University of Technology, Tehran, Iran, in 2017. He is currently pursuing the Ph.D. degree in computer science with the Western Norway University of Applied Sciences. His current research interests include artificial intelligence and machine learning and their applications, particularly in health, privacy, and security.

MATIN SHOKRI received the M.Sc. degree in computer engineering from the K. N. Toosi University of Technology, Tehran, Iran. He is currently working in machine learning algorithms, especially deep learning in image processing. His research interests include deep learning, reinforcement learning, and ensemble methods.

FAZLE RABBI is currently an Associate Professor. He has long and varied experience with information system development within a large spectrum of domain areas. His research interests include model-based software engineering, data mining, and machine learning, with emphasis on addressing the information science problems in healthcare applications. He is enthusiastic to improving the quality of living through his contribution in healthcare. Earlier, he was involved in academic research for developing reliable workflow management systems for two community-based health programs piloted at Guysborough Antigonish Strait Health Authority (GASHA), Nova Scotia, Canada. He is also interested in innovating new techniques for teaching both in classroom and online platform. Together with researchers from Kenya and Vanderbilt University Medical Center, he developed gamification approach for increasing student motivation and engagement in learning environment.

VIOLET KA I. PUN is currently an Associate Professor at the Western Norway University of Applied Sciences and the SIRIUS Centre, University of Oslo. Her research interests include using formal methods to specify, analyze, and verify the behavior of software programs, especially those running in distributed and concurrent systems. She is active in programming language theory, including language semantics, type systems, deductive verification, and formal logic. She is also interested in digitalization of healthcare domain, data privacy, and self-adaptive patient treatments. She is working on model-based business process planning with tool-supported and automated analyses in terms of formal methods.

YNGVE LAMO received the Ph.D. degree in computer science from the University of Bergen, in 2003, on formal specification of software systems with use of multialgebras. He is currently a Professor with the Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, Bergen, Norway. His research interests include model-based software engineering, formal methods, graph transformations, health informatics, and application of machine learning for analyzing health data.