A Privacy-Preserving Approach for Big Data Mining using RainForest with Federated Learning

Dipu Saha[1], Mainul Karim[1], Suriya Phongmoo[2] and Dewan Md. Farid[1]

***Abstract*—Federated Learning (FL) in Machine Learning (ML) has become very popular nowadays because it trains classifiers across multiple decentralized devices without transferring data to a central server. It is a decentralized learning approach that amalgamates several different nodes into one. In 2016, Google introduced the concept of FL when the use and misuse of personal data were gaining global attention. In this paper, we have proposed a privacy-preserving approach for mining big personalized data employing scalable decision tree induction with FL. The concept of the RainForest framework refers to addressing big data challenges via a Decision Tree (DT) classifier. The proposed method does not share the data or personal information with the central server. It only transfers the local models' parameter values to the central server. Each individual device or node trains its own local DT classifier and shares the prior and conditional probability values with the central server. We have tried to simulate the proposed concept using five benchmark datasets. The results of the evaluation indicate that the model exhibits exceptional performance and accuracy.**

Big Data; Machine Learning; Federated Learning; Decision Tree; RainForest Learning

I. INTRODUCTION

Throughout the last several decades, ML has evolved from a laboratory curiosity to a real technology with extensive commercial use. This pervasive and powerful form of artificial intelligence is changing every industry by analyzing big data, identifying patterns in big data, and making decisions without human intervention. A 2020 Deloitte survey [1] found that 67% of companies are using ML, and 97% are planning to use it in the next year. But due to constant changes in technology and strict government regulations, many questions are being raised about the standard process of ML. Following the implementation of the General Data Protection Regulation in the European Union in 2018, an agreement is mandatory before an institution or organization can use someone's private data [2]. However, since the client's data is shared directly with the developer in a single framework in the standard ML process, there is a possibility of it being misused by the developer himself or stolen by an intruder. Also, clients regularly interact with the server to complete the forecasting process, which negatively impacts the user experience owing to accessibility, network latency, power consumption, and other unanticipated concerns. Moreover, standard ML algorithms have challenges in the application of big data since the data is either too large to store in a centralized location or too complicated to be dealt with by typical data-processing application software [3]. Google came up with FL in 2016 to solve these problems [4]. The goal of this evolving ML project is to solve the problem of data privacy. The core idea behind FL is decentralized learning, where client data is never shared with or sent to a central server. Clients are regarded nodes in a network and may communicate with one another either directly or through a server [5]. Individual participants execute their own training processes. The model only transmits its learnt parameters (weights, biases, etc.) to the server where a master model is constructed. The quantity of data exchanged is minimized along with any associated privacy problems using this method [6]. Even the central server can operate with relatively little power and storage [7].

The most well-known techniques for representing classifiers for data classification are thought to be DT classifiers. It uses a tree-based approach, where every route starting at the root is defined by a data-separating sequence until a Boolean result is reached at the leaf node [8], [9]. It excels at handling large and complex datasets without necessitating a sophisticated parameter structure. It shows how nodes and connections in a network of knowledge relate to each other in a hierarchical way. Nodes stand in for purposes when relations are employed to categorize [10]. The RainForest Tree Model is a form of ensemble learning approach that entails the construction of several decision trees during training, followed by the use of their aggregate predictions when making final outputs [11]. The model is named after the rainforest ecosystem, as it mimics the natural diversity and complexity of such environments by using many diverse decision trees. Each DT in the ensemble is trained on a different subset of the training data and may use different features and decision rules. During prediction, the RainForest model aggregates the output of each DT to make a final decision. It is a powerful and flexible algorithm that is well-suited for big data analysis due to its ability to handle large and complex datasets, parallel processing, ensemble learning, and feature importance.

Based on our previous research work [12], we present the notion of incorporating FL into the RainForest model in

We appreciate the support for this research received from the a2i Innovation Fund of Innov-A-Thon 2018 (Ideabank ID No.: 12502) from a2i-Access to Information Program – II, Information & Communication TechnologyDivision, Government of the People's Republic of Bangladesh and Institute for Advanced Research (IAR), United International University (Project Code: UIU/IAR/01/2021/SE/23).

[1]Dipu Saha, Mainul Karim and Dewan Md. Farid with the Department of Computer Science and Engineering, United International University, United City, Madani Avenue, Badda, Dhaka 1212, Bangladesh dsaha211072@mscse.uiu.ac.bd, mkarim191076@bscse.uiu.ac.bd, dewanfarid@cse.uiu.ac.bd

[2]Suriya Phongmoo is with the Department of Industrial Engineering, Faculty of Engineering, Chiang Mai University, Chiang Mai 50200, Thailand suriya_p@cmu.ac.th

this study. By implementing this, privacy, performance, and scalability of the RainForest model can be improved in several ways, as follows:

- **Data privacy:** As FL allows multiple parties to collaborate on training a ML model without sharing their data with each other, this can be particularly beneficial for datasets that contain sensitive or confidential information, such as medical or financial data.
- **Distributed data:** FL can be used to train the RainForest tree model on distributed data sources, such as devices or sensors. In Internet of Things (IoT) applications and similar use cases where data is created locally but has to be processed in real time without being transferred to a central server, this approach might be helpful.
- **Model aggregation:** FL can be used to aggregate the individual models trained on different devices or data sources and combine them into a more accurate and robust global model. Since the individual models can be trained on distinct subsets of the data and features, this can help alleviate the problem of overfitting in the RainForest tree model.
- **Model personalization:** FL can also be used to personalize the RainForest tree model for individual users or devices by training separate models for each user or device based on their specific data and preferences. This may help increase the model's precision and applicability to various users or devices.
- **Communication cost:** Since FL only requires trained statistical models to be sent to the server rather than complete data, communication costs are drastically reduced.

The content of this paper is organized as follows: In section II, there is a discussion of related works. Section III introduces the framework and specific technical details of the proposed methodology, whereas section IV provides a synopsis of the experimental results. Lastly, section V concludes the paper, mentioning the advantages and suggesting future research directions.

II. Related Works

Since FL is still in its early stages, there is a lot of research and discussion on its practical application. Alferaidi et al. explained FL in their paper [13] by discussing its architectural and operational mechanisms, classification based on independent research objects and their different distribution characteristics, optimization strategies, current development status, and notable difficulties and solutions. In order to provide general readers with an understanding, a study [14] on the prevalence of FL in big data applications and services was published by Gadekalu et al. In their study [15], Pati et al. established the FL's performance at such size and complexity as a paradigm shift for multisite collaborations, therefore eliminating the need for data sharing. Zhao et al. developed an enhanced FL algorithm in their paper [16] for the mining of industrial big data to protect the confidentiality of data and users. At first, they lessened the privacy leakage by updating just a subset of participants and parameters in each iteration. In addition, they used a proxy server to shield clients' identities from the FL server and lighten its communication load. Furthermore, they used a Gaussian technique for differential privacy on shared parameters to provide complete confidentiality and implement a self-termination mechanism. All of their statements have been validated by experimental evidence. Even their approach is more resilient than other similar schemes and adds nothing to the computational expense. A common problem with data abundance is receiving irrelevant information. In their paper [17], Doku et al. proposed a strategy that combines the advantages of the blockchain with FL to retain the most relevant data discovered by users when engaging with their devices. They devised a unique network sharing method known as Interest Group. Interest groups consist of nodes that have similar interests. They did so by introducing a decentralized method of data storage that decreases the risk associated with centralized data storage and by proposing a consensus process known as the Proof of Common Interest. It assures the legitimacy of data by ensuring that data submitted to the ledger is relevant.

Aïvodji et al. generated an all-new architecture [18], IOTFLA, for FL-based smart homes, where ECIPAP should be implemented for secure data aggregation. This enables scenarios in which all smart home devices can use FL, none can, or a combination of both. This allows for a well-functioning smart home network with improved efficiency over time in a privacy-preserving manner by ensuring that only updates made to the models are transferred to allow the global model's update. Zhang & Jiang proposed a data augmentation strategy [6] called FedDA, based on WGAN-GP, to overcome the small amount of overlapped data. Learning the characteristics of just a handful of overlap data and the abundant nonoverlap data allows for the generation of a greater number of training samples when using this technique. It also passively enhances data quality, reduces iterations, and improves outcomes. Tsai et al. proposed a vision-based IoT technology [19] based on FL to boost productivity and optimize revenues for small and medium-sized firms. In order to locate and recognize edge nodes, it examines regional low-level properties. An embedded camera is used to construct the edge node, which monitors machines and processes data locally. The results from their test revealed that after 50 days of continuous operation, the proposed technique reduced the delay by nearly 1.4 hours. Leroy et al. came up with a method [20] based on FL that will let voice-assistance apps recognize wake words without putting sensitive user data at risk. They conducted an extensive empirical study of the federated averaging algorithm based on a crowdsourced dataset. Specifically, they showed experimentally that the number of communication rounds needed to attain goal performance can be drastically reduced by switching to an adaptive averaging technique inspired by the Adam optimizer rather than the traditional weighted model averaging.

III. Methodology

A. Federated Learning

The increasing computational power of even small devices nowadays makes frequent implementations of FL possible. It is possible to carry it out in either a centralized [21] or decentralized [22] configuration, depending on the preferences. In the centralized FL architecture, clients train an initial model sent from the central server on their own devices with their local data. Only the encrypted gradients of those trained models are sent back to the central server, with which the server updates the model by performing aggregation or pruning as needed. Participants carry out the iterative training procedure based on this new global model. This procedure goes on until the target performance level is reached or the allotted number of communication rounds is exceeded [23]. Fig. 1 shows the basic structure of the centralized FL. To protect one's anonymity even further, a secure aggregation mechanism was devised where the server pairs up clients with others in a buddy system, and their data is aggregated with some random values before being forwarded to the central server, therefore increasing their anonymity. Clearly aware of the values transmitted to the buddies, the server cancels them out in order to retrieve the payload. This trick obfuscates the data while it is in transit to the server. In a centralized FL setup, the mode of data flow between the server and the edge devices might be synchronous or asynchronous. The central server is in charge of notifying the edge devices about the commencement of the training process in a synchronous environment [4]. The server holds off sending the revised settings until all of the edge devices, or a substantial portion, forward their updated parameters. When working in an asynchronous environment, an edge device may initiate a fresh training pass whenever necessary [24]. The latest parameters made accessible from all edge devices are combined by the central server when it wishes to create a new global model.

In the decentralized FL architecture, the integration of a central server is not required. Instead, the devices communicate directly with each other to aggregate their model updates. This can be done using various protocols, such as gossip algorithms, where devices randomly exchange model updates with each other, or hierarchical clustering, where devices are organized into clusters and updates are aggregated within each cluster before being sent to other clusters. As the number of connected devices increases, the complexity of the communication process also grows [25].

Data is usually stored in different nodes or entities in the form of a feature matrix. Typically, the data will have numerous instances, and the clients will be on the horizontal axis with their characteristics on the vertical. According to data partition mode, FL is commonly classified as Horizontal FL, Vertical FL, and Federated transfer learning [26]. Horizontal FL takes use of the fact that even while no two sets of data are same in sample space, there is often overlap in the attributes of data located at different nodes. By contrast, Vertical FL is used in settings where the nodes share a

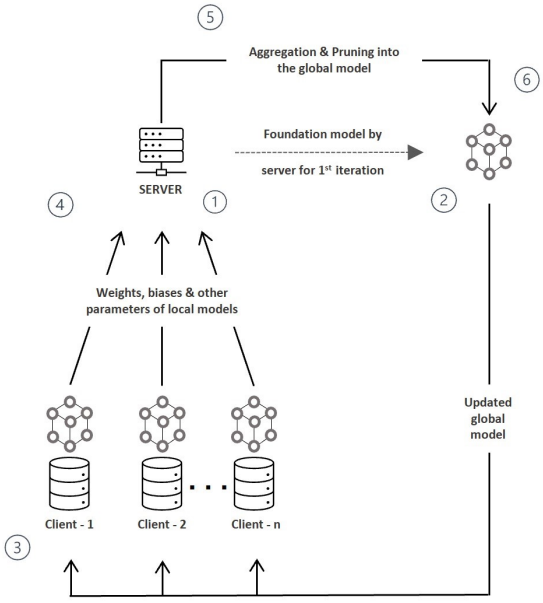

Fig. 1. A basic structure of Centralized Federated Learning.

similar feature space but a dissimilar sample space. Unlike these two, sometimes data shares relatively small samples and feature spaces. Consequently, a shortage of data labels and low quality of data are the key issues in this situation. To improve learning outcomes, Federated transfer learning facilitates shifting information from the source domain to the target domain.

B. Decision Tree

Behind DT's popularity, the main reason is its explainability. Anyone can understand this, even if they are from a non-CS background, and generate some rules from it. It works with any data format and can even deal with missing values. As less pre-processing is required, it is also computationally fast. It uses multiple algorithms like ID3, ID4, ID5, C4.5, C4.8, C5.0, MARS, GUIDE, CTREE, CRUISE, QUEST, etc. to split a node into sub-nodes [8]. The selection of the appropriate algorithm is determined by the nature of the desired outcomes. When there are several characteristics in a dataset, it might be difficult to decide which ones belong at the top and which ones should be nested deeper. Randomly selecting an approach might have inaccurate and unfavorable effects. Researchers worked on this issue and came up with various solutions. A few criteria were offered, including Entropy, Information Gain, Gini Index, Gain Ratio, Squared Error, etc. The Squared Error is used for the regression cases.

$$Squared\ Error = (Predicted\ value - Real\ value)^2 \quad (1)$$

Entropy is the measurement of disorder in a set. It is used for classification cases. Entropy has a fixed value that ranges

from 0 to 1. The closer the Entropy value is to 0, the better the splitting, and the closer it is to 1, the worse it is.

$$H\left(q\right) = -\sum_{i=1}^{k} p_i \log_2 p_i \tag{2}$$

Information Gain computes the difference between the Entropy before splitting and the average Entropy after splitting of the dataset based on given attribute values. Among all the attributes, the higher the Information Gain, the higher it will go in the tree.

$$Gain\left(q, V\right) = H\left(q\right) - \sum_{i=1}^{|V|} \frac{N_i}{N_q} H\left(i\right) \tag{3}$$

In practical life, many times unique classifications are seen that are misleading. For these, the Entropy becomes 0. In that case, many unimportant attributes will be selected. In such a particular situation, the Gain Ratio is applied rather than the Information Gain. Gain Ratio is obtained by dividing the Information Gain by the Split Information.

$$Gain\ Ratio = \frac{Gain\left(q, V\right)}{-\sum_{i=1}^{|V|} \frac{N_i}{N_q} \log_2 \frac{N_i}{N_q}} \tag{4}$$

The Gini index, used for classification cases, calculates the likelihood that a randomly selected element from the set would be erroneously identified if it were randomly assigned a label based on how those labels are distributed in the subset. The attribute with the lowest Gini Index will sit higher, followed by the rest in order.

$$Gini\left(q\right) = \sum_{i=1}^{k} p_i \left(1 - p_i\right) \tag{5}$$

C. RainForest Method

In this era of big data, data is too large and complex to be dealt with. Of course, that huge amount of data cannot be stored in a small memory or in any centralized location. At this point, the RainForest Model concept comes in handy. It is applicable to all DT algorithms and may adjust itself to the amount of main memory that is available [27]. To represent the training instances at each node of the tree, the approach keeps an AVC-set (Attribute-Value, Class-label). The AVC set of a given attribute A at a given node N provides the number of examples labeled with a certain class for each possible value of A. Fig. 2 shows the basic architecture of the RainForest model. For example, if we consider Table I as the subdata set d_1, then the AVC-set of the temperature attribute of d_1 will be considered as mentioned in Table II.

D. Proposed Framework

This subsection illustrates the proposed approach for addressing privacy concerns and managing big data with the RainForest model through the application of FL. Initially, the central server leverages a simple k-means clustering method to partition the vast amount of data it has stored into several subsets. Subsequently, the server dispatches a unique subset

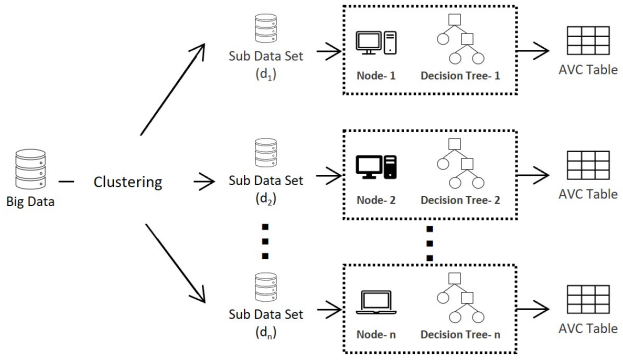

Fig. 2. A basic structure of RainForest model.

and an initial model to each local node, where the nodes utilize their own data in combination with the provided subset to train the model. Each node constructs a DT by implementing probabilistic C4.8 and producing an AVC table for each attribute of the tree. All AVC tables are then transmitted back to the central server, where they are combined through matrix addition to form a global DT. The resultant model is then conveyed back to the local nodes as the new initial model for the subsequent iteration. Meanwhile, new data will be generated on the local nodes, which will be employed to train the next iteration. This iterative process continues until the specified criteria are met. Fig. 3 shows the whole process of the proposed method.

IV. Experiments

A. Datasets & Experimental Setup

The complexity of our proposed model raises doubts about its accuracy compared to traditional algorithms. Therefore, we have divided our experimental setup into two parts. The experiments were conducted on a machine equipped with an Intel(R) Core (TM) i5-6500 CPU @ 3.20 GHz and 12 GB of RAM, using the Visual Studio IDE to implement our solution in Python. Our experiment involved five distinct datasets obtained from the KAGGLE website, which are described in Table III. The initial phase of our experiment involved preprocessing the datasets and measuring their accuracy using two traditional algorithms, namely the Decision Tree and Random Forest. We then proceeded to the second phase, where we evaluated the accuracy of our proposed model. To conduct this phase, we utilized the Flower Framework, a powerful tool for experimental analysis in FL. This framework enables researchers and developers to test and implement various FL algorithms and evaluate their performance under different scenarios by creating multiple virtual nodes on a single computer. For this experiment, we set up one main server and four local nodes. The final readings were obtained after waiting for up to three iterations.

B. Results and Discussion

By implementing the proposed model and comparing its results against traditional algorithms using multiple datasets

TABLE I
A SUB DATASET OF THE PLAYING TENNIS

Day	Temperature	Humidity	Wind	Play
Day 1	Hot	High	Weak	No
Day 2	Hot	High	Strong	No
Day 5	Mild	High	Weak	No
Day 8	Cold	Normal	Weak	Yes
Day 9	Mild	Normal	Strong	Yes

TABLE II
AVC-SET OF THE TEMPERATURE ATTRIBUTE OF TABLE I

Temperature	Yes	No
Hot	0	2
Mild	1	1
Cold	1	0

and replicating the process multiple times, we aimed to provide a robust and reliable assessment of its accuracy. Table IV presents the accuracy outcomes of all algorithms for each dataset. Notably, the proposed model demonstrated superior accuracy compared to the other two models for the Titanic, Iris Species, and Palmer Archipelago datasets. For the breast cancer dataset, the proposed model yielded an enhanced accuracy rate in comparison to the Decision Tree, and it achieved the same accuracy rate as the Random Forest. However, for the stroke prediction dataset, both the proposed model and the Decision Tree produced the same accuracy, whereas the Random Forest achieved 4% higher. These findings indicate that the proposed model exhibits a promising approach for enhancing the accuracy of classification tasks in contrast to conventional models, particularly when dealing with big data. Even if the accuracy is equal or marginally lower, it still holds promise for privacy preservation or personalized modeling purposes that traditional algorithms cannot.

V. CONCLUSION

This paper shows the merging of two different concepts, Federated Learning and RainForest, which offer several advantages when working together. In terms of privacy, time, memory, and power consumption, this approach is comparatively more efficient. Moreover, the model shows better performance due to the use of big data. In the future, there is scope to further improve the proposed concept of using FL in the RainForest model. One possible avenue for future research is to investigate ways to enhance the privacy and security of the data transmitted from edge devices during the FL process. This could involve exploring new encryption techniques or developing more secure communication protocols. Additionally, there may be opportunities to optimize the FL process itself, perhaps by exploring new aggregation techniques or investigating the impact of different hyperparameters on model performance. Finally, it would be valuable to compare the proposed concept with other emerging techniques, such as Differential privacy, Secure multiparty computation, Homomorphic encryption, and determine its relative strengths and weaknesses in different contexts.

TABLE III
DATASET DESCRIPTION

Datasets	No. of attributes	Attribute types	No. of instances	No. of output
Titanic	28	Mixed	1309	2
Iris Species	6	Mixed	150	3
Breast Cancer	32	Mixed	569	2
Stroke Prediction	12	Mixed	5110	2
Palmer Archipelago	17	Mixed	344	3

TABLE IV
ACCURACY

Datasets	Decision Tree	Random Forest	Proposed Model
Titanic	0.82	0.86	0.88
Iris Species	0.93	0.96	0.97
Breast Cancer	0.94	0.95	0.95
Stroke Prediction	0.90	0.94	0.90
Palmer Archipelago	0.95	0.98	0.99

REFERENCES

[1] B. Ammanath, D. Jarvis, and S. Hupfer, "Thriving in the era of pervasive ai," *Deloitte, rep*, 2020.

[2] C. J. Bennett, "The european general data protection regulation: An instrument for the globalization of privacy standards?" *Information Polity*, vol. 23, no. 2, pp. 239–246, 2018.

[3] D. M. Farid, M. A. Al-Mamun, B. Manderick, and A. Nowé, "An adaptive rule-based classifier for mining big biological data," *Expert Systems with Applications*, vol. 64, pp. 305–316, December 2016.

[4] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[5] X. Qu, J. Wang, and J. Xiao, "Quantization and knowledge distillation for efficient federated learning on edge devices," in *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2020, pp. 967–972.

[6] J. Zhang and Y. Jiang, "A data augmentation method for vertical federated learning," *Wireless Communications and Mobile Computing*, vol. 2022, 2022.

[7] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed federated learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1146–1159, 2019.

[8] D. M. Farid, L. Zhang, C. M. Rahman, M. A. Hossain, and R. Strachan, "Hybrid decision tree and naïve bayes classifiers for multi-class classification tasks," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1937–1946, March 2014.

[9] F.-J. Yang, "An extended idea about decision trees," in *2019 International Conference on Computational Science and Computational Intelligence (CSCI)*. IEEE, 2019, pp. 349–354.

[10] A. Shamim, H. Hussain, and M. U. Shaikh, "A framework for generation of rules from decision tree and decision table," in *2010 International Conference on Information and Emerging Technologies*. IEEE, 2010, pp. 1–6.

[11] J. Gehrke, R. Ramakrishnan, and V. Ganti, "Rainforest—a framework for fast decision tree construction of large datasets," *Data Mining and Knowledge Discovery*, vol. 4, pp. 127–162, 2000.

[12] M. Karim, N. Kundu, D. Saha, S. Kabir, S. A. Mim, and D. M. Farid, "Implementing federated learning based on rainforest model," in *IEEE 8th International Conference for Convergence in Technology (I2CT)*, Pune (Maharashtra), India, April 2023, pp. 1–6.

[13] A. Alferaidi, K. Yadav, Y. Alharbi, W. Viriyasitavat, S. Kautish, and G. Dhiman, "Federated learning algorithms to optimize the client and cost selections," *Mathematical Problems in Engineering*, vol. 2022, 2022.

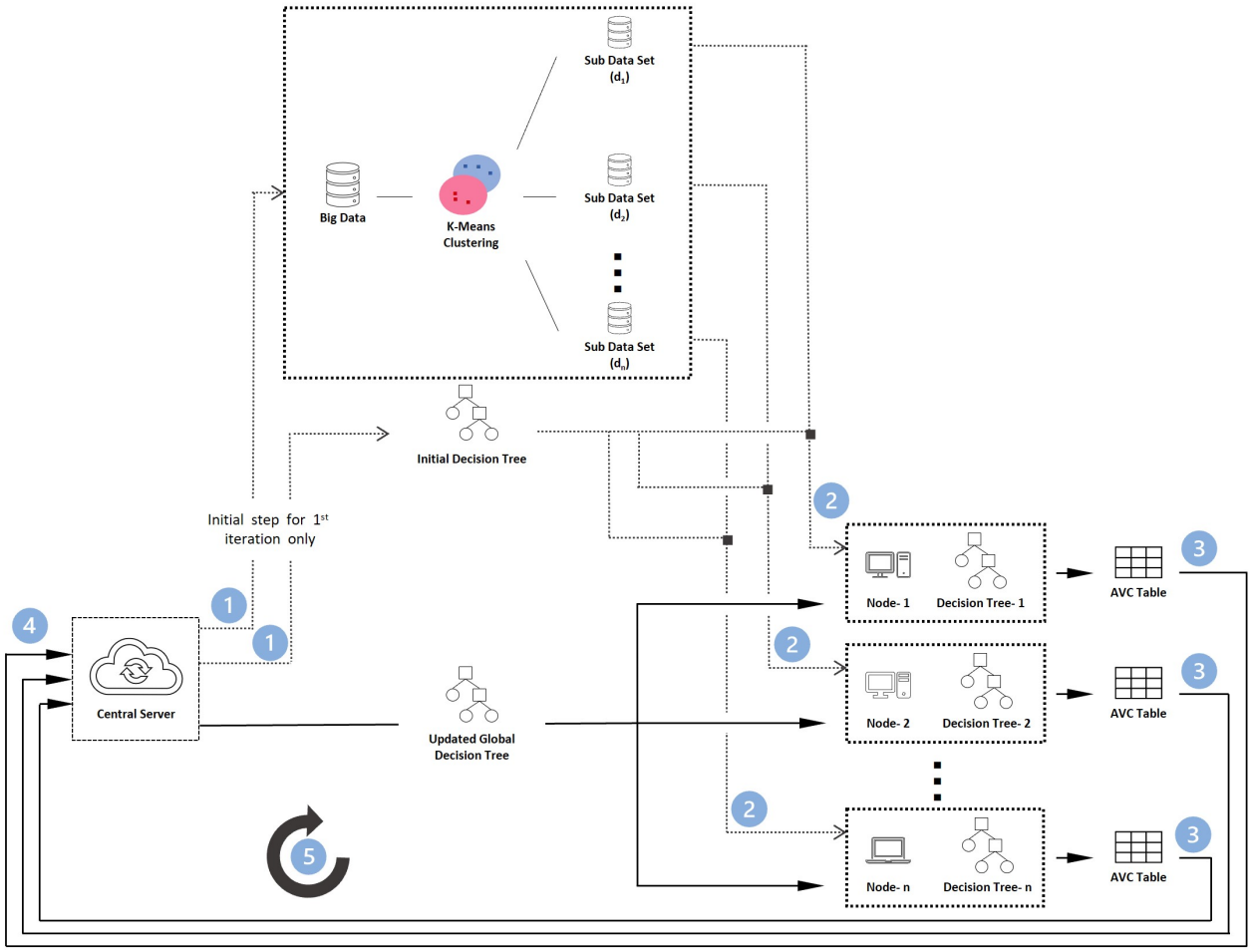

Fig. 3. Proposed Framework: Big Data Mining using RainForest with Federated Learning.

[14] T. R. Gadekallu, Q.-V. Pham, T. Huynh-The, S. Bhattacharya, P. K. R. Maddikunta, and M. Liyanage, "Federated learning for big data: A survey on opportunities, applications, and future directions," *arXiv preprint arXiv:2110.04160*, 2021.

[15] S. Pati, U. Baid, B. Edwards, M. Sheller, S.-H. Wang, G. A. Reina, P. Foley, A. Gruzdev, D. Karkada, C. Davatzikos *et al.*, "Federated learning enables big data for rare cancer boundary detection," *Nature communications*, vol. 13, no. 1, p. 7346, 2022.

[16] B. Zhao, K. Fan, K. Yang, Z. Wang, H. Li, and Y. Yang, "Anonymous and privacy-preserving federated learning with industrial big data," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6314–6323, 2021.

[17] R. Doku, D. B. Rawat, and C. Liu, "Towards federated learning approach to determine data relevance in big data," in *2019 IEEE 20th international conference on information reuse and integration for data science (IRI)*. IEEE, 2019, pp. 184–192.

[18] U. M. Aïvodji, S. Gambs, and A. Martin, "Iotfla: A secured and privacy-preserving smart home architecture implementing federated learning," in *2019 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2019, pp. 175–180.

[19] Y.-H. Tsai, D.-M. Chang, and T.-C. Hsu, "Edge computing based on federated learning for machine monitoring," *Applied Sciences*, vol. 12, no. 10, p. 5178, 2022.

[20] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, "Federated learning for keyword spotting," in *ICASSP 2019-2019 IEEE*

[21] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6341–6345.
"Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[22] C. Korkmaz, H. E. Kocas, A. Uysal, A. Masry, O. Ozkasap, and B. Akgun, "Chain fl: decentralized federated machine learning via blockchain," in *2020 Second international conference on blockchain computing and applications (BCCA)*. IEEE, 2020, pp. 140–146.

[23] M. Duan, D. Liu, X. Chen, Y. Tan, J. Ren, L. Qiao, and L. Liang, "Astraea: Self-balancing federated learning for improving classification accuracy of mobile deep learning applications," in *2019 IEEE 37th international conference on computer design (ICCD)*. IEEE, 2019, pp. 246–254.

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

[25] M. Joshi, A. Pal, and M. Sankarasubbu, "Federated learning for healthcare domain-pipeline, applications and challenges," *ACM Transactions on Computing for Healthcare*, vol. 3, no. 4, pp. 1–36, 2022.

[26] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[27] S. Sabah, S. Z. B. Anwar, S. Afroze, M. A. Azad, S. Shatabda, and D. M. Farid, "Big data with decision tree induction," in *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)*. IEEE, 2019, pp. 1–6.