MLPAM: A Machine Learning and Probabilistic Analysis Based Model for Preserving Security and Privacy in Cloud Environment

Ishu Gupta, *Member, IEEE*, Rishabh Gupta, Ashutosh Kumar Singh, *Senior Member, IEEE*, and Rajkumar Buyya, *Fellow, IEEE*

***Abstract*—The organizational valuable data needs to be shared with multiple parties and stakeholders in a cloud environment for storage, analysis, and data utilization. However, to ensure the security, preserve privacy while sharing the data effectively among various parties have become formidable challenges. In this article, by utilizing encryption, machine learning, and probabilistic approaches, we propose a novel model that supports multiple participants to securely share their data for distinct purposes. The model defines the access policy and communication protocol among the involved multiple untrusted parties to process the owners' data. The proposed model minimizes the risk associated with the leakage by providing a robust mechanism for prevention coupled with detection. The experimental results demonstrate the efficiency of the proposed model for different classifiers over various datasets. The proposed model ensures high accuracy and precision up to 97% and 100% relatively and secures a significant improvement up to 0.01%, 103%, 151%, 87%, 96%, 43%, and 186% for average probability, average success rate, detection rate, accuracy, precision, recall, and specificity, respectively, compared to the prior works that prove its effectiveness.**

***Index Terms*—Cloud computing, data leakage, data privacy, data security, distribution mechanism, machine learning.**

I. Introduction

DATA storage, analysis, and sharing are the essential services required by any organization to upgrade its performance [1]. Most of the businesses have shifted to the cloud due to its several benefits such as minimum upfront cost and maximum scalability for the required services [2]. However, once the data is transferred for storage and computation purposes in the cloud, the owners lose control over their data [3]. Multiple entities may access the data for commercial and/or other purposes after the data is outsourced [4]. It is not possible to fully trust the cloud platform because it is handled by the third party [5]. Therefore, before uploading data onto the cloud, owners first encrypt their data for privacy reasons. Although some conventional encryption techniques are available for the encryption of owners' data, such as symmetric and fully homomorphic cryptography, these techniques are insufficient [6], [7]. However, it becomes difficult to perform the computation over the encrypted data [8]. There arises a necessity to protect the owners' as well as the cloud data while performing the computation effectively. Furthermore, the stored and analyzed data must be shared with the various stakeholders to improve its utility. Although the data is shared among authorized entities, it cannot be assured that data will not be leaked by the receiving entities after obtaining it [9]. Thus, it is essential to protect the data from the entities involved in the communication process. To solve the above-mentioned challenges, we need an effective access control method that supports both the privacy and security of the owners' data. To the best of the author's knowledge, no model exists that solves all the aforementioned challenges. In this regard, we propose a novel Machine Learning and Probabilistic Analysis based Model (MLPAM) for data protection through privacy-preserving data storage and analysis, secure sharing, and identification of guilty entity against data leakage in the cloud environment. The main contributions of MLPAM are summarized as follows.

1) To protect the data with enhanced security, all the entities are considered to be untrusted and MLPAM deals with involved entities by effectively defining an access policy.
2) MLPAM enables multiple data owners to freely share the outsourced data. In order to protect the data from stealing or leakage, the data of each owner is encrypted with a separate key and shared in encrypted form.
3) MLPAM uses two clouds where cloud1 deals with data storage, handling, and sharing whereas cloud2 generates the key for the encryption of owners' data and performs the computation over the data obtained from cloud1 for privacy-preserving classification.
4) An effective distribution mechanism based on an access control is proposed for data distribution among multiple users, that enables to identify the guilty entity and reduces the risk associated with further leakage.
5) A series of experiments are conducted using the widely adopted datasets by researchers to validate the practicality of the proposed model. In addition to this, the comparisons are interpreted among the various a) datasets, b) classifiers, and c) distinctly preprocessed data using ϵ-differential

Manuscript received March 30, 2020; revised October 1, 2020; accepted October 26, 2020. Date of publication November 24, 2020; date of current version August 26, 2021. This work was financially supported by the University Grants Commission (UGC), Government of India. *(Corresponding author: Ishu Gupta.)*

Ishu Gupta, Rishabh Gupta, and Ashutosh Kumar Singh are with the Department of Computer Applications, National Institute of Technology, Kurukshetra, Kurukshetra 136119, India (e-mail: ishugupta23@gmail.com; rishabhgpt66@gmail.com; ashutosh@nitkkr.ac.in).

Rajkumar Buyya is with the Cloud Computing and Distributed Systems Laboratory, School of Computing and Information Systems, University of Melbourne, Melbourne, VIC 3010, Australia (e-mail: rbuyya@unimelb.edu.au).

Digital Object Identifier 10.1109/JSYST.2020.3035666

TABLE I
LIST OF TERMINOLOGIES WITH THEIR EXPLANATORY TERMS

DO_i: data owner; RU_j: request user; TP_k: third-party; CF: classifier
CP: cloud platform; CS: cloud storage; CSP: cloud service provider
CM: classification model; D_i: plain data; $D_i^{N_i}$: encrypted noised data
D_i^E: encrypted data; $D_i^{N_i'}$: plain noised data; $\hat{D}_i^{N_i'}$: preprocessed data
$\hat{D}_{t',i}^{N_i'}$: training data; $\hat{D}_{t'',i}^{N_i'}$: testing data; m: users count; ℓ: leaked dataset
n: number of data objects; k: third-parties count; n^*: count of classes
n^{**}: training objects count; n^{***}: testing objects count; PB_i^K: public key
PV_i^K: private key; SC_j^K: secret key; N_i: noise; N_i^E: encrypted noise
$\mathbb{C}_i^*$: object category; $\mathbb{Z}_i$: demanding userset; $\mathbb{Y}_j$: demanding dataset
L_i: label item; λ: object categories count; T_η: tuple; $A_{\eta'}$: data attribute
S_α: sensitive attribute; Q_β: quasi-identifier attribute; Ω: number of tuples
Δ: attributes count; Δ^*: sensitive attributes count; $\mathcal{I}_\eta^{\mathcal{C}}$: identity coefficient
Δ^{**}: quasi-identifiers count; Υ_j: attributes of RU_j; $\mathcal{O}_j^{\mathcal{F}}$: object flag
TSM_η^F: formative tuple sensitivity measure; W^*: cumulative weight
TSM_η^*: cumulative tuple sensitivity measure; $\mathcal{D}_i^{\mathcal{F}}$:distribution factor
$OSM_{D_i}^F$: formative object sensitivity measure; $\mathcal{S}_i^{\mathcal{I}}$: sensitivity index
$OSM_{D_i}^*$: cumulative object sensitivity measure; $\mathcal{L}_i^{\mathcal{C}}$: limit coefficient
$OSM_{D_i}^{**}$: standardized object sensitivity measure; $\rho^{\mathcal{C}}$: worth coefficient
$P_b\{G_{U_j}\|\mathbb{Y}_j^*\}$: probability of leaking data; $\varpi_{(j,k)}^*$: difference function
$\overline{\varpi}^*$: average success rate; $\min \varpi^*$: detection rate; NG_U: non-guilty user
G_U: guilty user; UE: untrusted entity; P_b: probability; $\mathcal{W}_{\mathcal{F}}$: weight factor
θ: guessing probability; $\mathcal{T}_{\mathcal{H}}$: threshold value; CA: classification accuracy
DA: detection accuracy; DP: detection precision; DR: detection recall
DS: detection specificity; T_E: encryption time; T_D: decryption time

privacy and with the state of the artworks to prove the superiority of MLPAM.

Organization: The related work is discussed in Section II. Section III introduces the system and adversary model along with the problem statement and design goals of MLPAM. The proposed model is entailed in Section IV. Sections V and VI describe the applied encryption mechanism and the introduced classification model, respectively. In Section VII, data is distributed based on a distribution factor that is computed using the parameters demanding usersets and data objects sensitivity discussed in Section VII-A. In Section VIII, multiple probabilistic and performance parameters are evaluated. Performance analysis of MLPAM is conducted in Section IX followed by the summary of the proposed work in Section X. Table I depicts the list of notations with their descriptions that have been used throughout the article.

II. RECENT KEY CONTRIBUTIONS

A. Security Based on Ciphertext-Policy Attribute-Based Encryption (CP-ABE)

Wang *et al.* [10] proposed a File Hierarchy CP-ABE scheme to secure the data in the cloud environment. This scheme utilized an access structure layered model, which can effectively resist Chosen Plaintext Attacks under the assumption of Decisional Bilinear Diffie–Hellman. The computation cost increased dynamically in this scheme when an integrated ciphertext is computed by the data owner. A data access control scheme for cloud storage to achieve a fair key reconstruction in which none of the users send their shares and no one can access the shared data is proposed by Liu *et al.* [11]. The experimental analysis demonstrated that computation delay and communication costs are limited, but the authentication is not effective in this scheme. Liu *et al.* [12] proposed a CP-ABE scheme to reduce the computation cost of the user, as the cost of heavy decryption increases with the complexity of access policy. The performance of the proposed scheme was analyzed by measuring storage overhead and processing power but it lacks in terms of privacy protection. To protect the personal privacy of the user and ensure data confidentiality, Zhang *et al.* [13] proposed a framework of the Hidden access Policy CP-ABE scheme. They designed an identification method to verify the authorized user and completed the decryption process. This scheme provided a constant size private key independent of the number of user attributes and reduced the transmission as well as storage costs. However, it is considered as a weak security model, because, only the "AND" policy is supported by it. In order to improve the efficiency of the policy and file updation dynamically, Li *et al.* [14] proposed a CP-ABE scheme based on the linear secret-sharing schemes (LSSS) matrix access structure in cloud computing. This scheme reduced the computing cost of data owner, communication expense, and storage consumption of the proxy cloud service provider as well as resisted the selected plaintext attacks. But, the time cost in file updation is more which is the major downfall of this scheme.

B. Privacy-Preserving Machine Learning

A Doubly Permuted Homomorphic Encryption (DPHE) based privacy-preserving mechanism that enabled multiparty protected scalar product is proposed by Yonetani *et al.* [15], which reduced the high computational cost. The major disadvantage of DPHE is that only one operation either addition or multiplication is supported at a time. Li *et al.* [16] proposed a scheme for a classifier owner to delegate a remote server and to provide the privacy-preserving classification service for users. A drawback of this scheme is that the interactions of the users were frequently involved while launching a classification query. A data protection scheme is proposed by Li *et al.* [17], which enabled a trainer to train a Naive Bayes classifier over the dataset provided jointly by different data owners. ϵ-differential privacy is utilized in this scheme to preserve the privacy of every owner. In this approach, the collusion is allowed and adversaries had the ability to forge and manipulate the data. To solve the problem of training the model over the encrypted data under multiple keys, a privacy-preserving deep learning model (PDLM) is proposed by Ma *et al.* [18]. The model is trained based on stochastic gradient descent and the feed-forward as well as a back-propagation procedure is performed based on a privacy-preserving calculation toolkit. PDLM reduced the storage overhead but the classification accuracy is less and the computation cost is high in this scheme. Li *et al.* [19] proposed a Privacy-preserving Machine Learning with Multiple data providers scheme to protect the privacy of the datasets. Public key encryption with a double decryption algorithm and ϵ-differential privacy are used to encrypt the datasets of different data providers and the cloud, respectively. However, the proposed solution approached with a high computational cost due to the dependence on integer factorization. Li *et al.* [20] introduced a privacy-conserving outsourced classification in cloud computing framework under various public keys using fully homomorphic encryption proxy technique. But, the data owners and the storage servers are considered in the same trustworthy domain that is no longer applicable in the cloud environment. To avoid information leakage under the substitution-then-comparison attack, a scheme was proposed by Gao *et al.* [21]. By adopting a double-blinding technique to protect data privacy, a privacy-preserving classification mechanism is designed for Naive Bayes and the communication as well as computation overhead are reduced. However, the scheme is not able to achieve the discovery of truth that protects privacy. Hesamifard *et al.* [22] proposed a framework named as CryptoDL for applying deep neural network algorithms to encrypted data. They established neural networking techniques while considering the existing limitations of homomorphic encryption schemes. Although the method works well to secure the private data, the different owners' data are protected using a key that is not practical.

C. Security Based on Probabilistic Analysis

The pioneering work in the area of probabilistic analysis to detect a guilty agent responsible for leaking the data in a cloud environment named as Guilt Agent Model (GAM) is proposed by Papadimitriou and Garcia-Molina [23]. This model is based on statistical analysis where the probability of various agents for being guilty has been assessed. GAM is widely used by several researchers for malicious user detection in a shared data environment. The parameters for guilty agent detection are improved by Dynamic-Threshold-based Information Leaker Identification Scheme (DT-ILIS) in [24] over GAM [23]. This scheme utilized an access control mechanism to distribute the data among authorized entities. Fan *et al.* [25] presented a distribution model for data leakage prevention by considering the guilt probability. This model selected a file allocation plan with minimum overlap between obtained file sets of users to find the leakage sources with high probability. In order to share the cloud data in a secure manner, a data leakage detection model (DLDM) that identified the malicious entity by utilizing an integration of watermarking and probabilistic approach is presented in [26]. To provide stronger security to the shared data, DLDM utilized the cryptography and hashing techniques and protected the confidential information from the unauthorized entity. The advantage of the probabilistic method is that the leaker identification is independent of the alteration or removal in the embedded data, unlike the watermarking technique [23], [27].

The major downfall of the existing work is that the models supported single owners and/or dealt with the single untrusted entity (UE) only, which is not feasible in the real environment. Unlike the existing works, MLPAM establishes a robust mechanism for absolute and efficient data protection in the sharing environment by contemplating all the involved entities as untrusted and ensuring the security and privacy jointly in association with the prevention as well as detection.

III. Problem Formulation

This section characterizes the entities involved in the model with their assigned tasks, all the possible threats that may arise in the protocol, defines the problem and outlines the design goals.

A. System Model

The system model comprises the four entities Data Owners (DO_{id}), Cloud Platform (CP), Request Users (RU_{id}), and Third Party (TP_{id}) that are described as follows.

1) DO_{id}: An entity generating the information and requesting services from CP. DO_{id} encrypts the data prior to uploading it to CP. Since it is believed that DO_{id} cannot leak its own data, but may leak the other owner's data, therefore, DO_{id} is treated as an untrusted entity.
2) CP: An entity that collects all the encrypted data from DO_{id} and offers storing, computing, and sharing facilities to DO_{id} or RU_{id}. CP transforms the ciphertexts sent by DO_{id}, performs certain computations over it, and encrypts the calculated outcome for secure sharing among DO_{id} or RU_{id}. CP trains the obtained information using machine learning algorithms. CP is a semitrusted but untrusted entity in the model as it follows the protocol strictly, but curious to learn the information. In our system model, CP comprises two clouds where cloud1 consists of Cloud Storage (CS) and Cloud Service Provider (CSP), whereas the Classifier (CF) belongs to cloud2. CSP is the only entity that acts as a bridge and applies ϵ-differential privacy, distribution mechanism, and detection mechanism to perform the tasks of data transformation, data distribution, and guilty entity detection.
3) RU_{id}: An entity receiving the data from CP in the encrypted form along with the key. It obtains the usable data by performing the decryption over the received data from CP. In the system model, RU_{id} is treated as an untrusted entity.
4) TP_{id}: An unauthorized and untrusted entity that belongs indirectly to the system. TP_{id} can access the relevant information from a malicious entity or by stealing the dataset from the authorized entity.

B. Adversary Model

CP and RU_{id} are the authorized but untrusted entities in MLPAM having permission to access the data owned by DO_{id}. The following are the possible adversaries in MLPAM that can misuse the data through an unauthorized way.

1) TP_{id} can corrupt the *data owner* ($DO_{id}; id \in [1, n]$) for leaking the data of $DO_{id'}; id' \in [1, n] \wedge id' \neq id$.
2) TP_{id} can convince *Cloud Service Provider* (CSP) to leak the data shared by $DO_{id}; id \in [1, n]$. Or TP_{id} can corrupt *Classifier* (CF) to leak the data shared by CSP.
3) TP_{id} can deal with *Request User* (RU_{id}) in order to leak the data of $DO_{id}; id \in [1, n]$ shared by CSP.
4) TP_{id} can try to access the data by stealing it during communication among DO_{id}, CSP, CF, and RU_{id}.
5) *Third Party* TP_{id} can acquire the data by stealing it from $DO_{id}; id \in [1, n]$.
6) *Third Party* TP_{id} can compromise the data by stealing through any malicious activity from cloud1 or cloud2.
7) *Third Party* TP_{id} can misuse the data after stealing it from RU_{id}.

C. Problem Statement and Design Goals

The multiple *Data Owners* $DO_1, DO_2, \ldots, DO_n$ possess the data $D_1, D_2, \ldots, D_n$ that need to be shared with CP and among a set of *Request Users* $RU_1, RU_2, \ldots, RU_m$. The following are the challenges faced by $DO_1, DO_2, \ldots, DO_n$ during sharing.

1) Data sharing comprises the risks of security, privacy, and leakage. The entities involved in communication (DO_{id}, CP, RU_{id}) can misuse/leak the *shared data* $D_1, D_2, \ldots, D_n$ or unauthorized third party TP_{id} can obtain $D_1, D_2, \ldots, D_n$ by stealing from DO_{id}, CP, and RU_{id} or during communication among DO_{id}, CP, RU_{id}.
2) To protect the object D_i from $DO_{id}; id \in [1, n] \wedge id \neq i$.
3) To protect the owners' data $D_1, D_2, \ldots, D_n$, it is shared in encrypted form $D_1^E, D_2^E, \ldots, D_n^E$, but the computations over $D_1^E, D_2^E, \ldots, D_n^E$ have limited accuracy.
4) If $D_1^E, D_2^E, \ldots, D_n^E$ is shared along with the key, then CP or TP_{id} may obtain $D_1, D_2, \ldots, D_n$.

The following are the design goals of MLPAM based on the aforementioned problem statement and the adversary model.

1) To provide $DO_1, DO_2, \ldots, DO_n$ with an efficient method that allows them to share their data $D_1, D_2, \ldots, D_n$ while preserving security and privacy.
2) To preserve the confidentiality of $D_1, D_2, \ldots, D_n$ via sharing it among the authorized parties only and by protecting the data of DO_i from other owners $DO_{id}; id \in [1, n] \wedge id \neq i$, CP, unprivileged $RU_1, RU_2, \ldots, RU_m$, and TP_{id}. $RU_1, RU_2, \ldots, RU_m$ are entitled to access $D_1, D_2, \ldots, D_n$ based on the distribution mechanism.
3) To perform the computation over $D_1, D_2, \ldots, D_n$ while preserving privacy and with improved accuracy and efficiency.
4) To share $D_1, D_2, \ldots, D_n$ by minimizing the likelihood of leakage and to be capable of detecting the guilty user with improved accuracy.

IV. Proposed Model

The architecture of the proposed model called MLPAM is depicted in Fig. 1. It shows the entities involved along with the communication among them as well as the critical blocks with essential flow among these blocks. Let the *owners* $\mathbb{DO} = \{DO_1, DO_2, \ldots, DO_n\}$ own the *data* $\mathbb{D} = \{D_1, D_2, \ldots, D_n\}$ where the *data object* $D_i \in \mathbb{D}$ is independent and can be of any type and size. $\mathbb{DO}$ need to share $\mathbb{D}$ among authorized parties like Cloud Platform (CP) and various *users* $\mathbb{RU} = \{RU_1, RU_2, \ldots, RU_m\}$ for storage, computation, and performance enhancement, etc., but do not aspire $\mathbb{D}$ to be leaked to an *unauthorized third party* $\mathbb{TP} = \{TP_1, TP_2, \ldots, TP_k\}$. The Classifier ($CF$) generates *private keys* $\mathbb{PV}^{\mathbb{K}} = \{PV_1^K, PV_2^K, \ldots, PV_n^K\}$ and *public keys* $\mathbb{PB}^{\mathbb{K}} = \{PB_1^K, PB_2^K, \ldots, PB_n^K\}$. The keys $PB_1^K, PB_2^K, \ldots, PB_n^K$ are shared with Cloud Service Provider (CSP), which transfers these to $DO_1, DO_2, \ldots, DO_n$, respectively. To make the data private and secure, $DO_1, DO_2, \ldots, DO_n$ procure the encrypted data $\mathbb{D}^{\mathbb{E}} = \{D_1^E, D_2^E, \ldots, D_n^E\}$ by applying an encryption technique along with their individual key

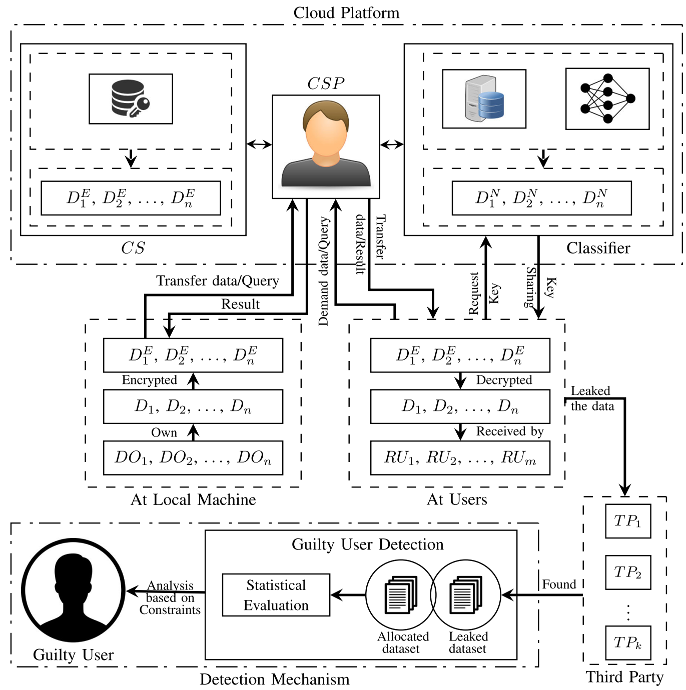

Fig. 1. Proposed MLPAM architecture.

$PB_1^K, PB_2^K, \ldots, PB_n^K$. MLPAM utilizes the CP-ABE scheme to encrypt $D_1, D_2, \ldots, D_n$ because CP-ABE is considered as the most appropriate technology for data confidentiality and control of fine-grained data access. It is a secure and efficient encryption technique that uses attributes of $RU_1, RU_2, \ldots, RU_m$ to encrypt and decrypt $D_1, D_2, \ldots, D_n$ without any loss and promises access control policy determined by $DO_1, DO_2, \ldots, DO_n$ itself as per the attributes of $RU_1, RU_2, \ldots, RU_m$ [28]. $D_1^E, D_2^E, \ldots, D_n^E$ are shared with CSP that stores these in Cloud Storage (CS). CSP transforms the stored $D_1^E, D_2^E, \ldots, D_n^E$ into encrypted noised data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ by adding random noise over it and transmits it to CF for computation. The entity CF decrypts $D_1^N, D_2^N, \ldots, D_n^N$ using $PV_1^K, PV_2^K, \ldots, PV_n^K$ individually and obtains the plain noised data $\mathbb{D}^{\mathbb{N}'} = \{D_1^{N'_1}, D_2^{N'_2}, \ldots, D_n^{N'_n}\}$ to perform the classification over it. CF performs the computation using $D_1^{N'}, D_2^{N'}, \ldots, D_n^{N'}$ and produces a Classification Model (CM). Any query can be made by $DO_1, DO_2, \ldots, DO_n$ or $RU_1, RU_2, \ldots, RU_m$ through CSP. The entity CSP communicates with CF, which receives the results from CM and sends back to CSP. Afterward, CSP delivers the acquired results to the corresponding entity $DO_1, DO_2, \ldots, DO_n$ or $RU_1, RU_2, \ldots, RU_m$. Furthermore, $RU_1, RU_2, \ldots, RU_m$ may request the data from CSP that distributes the data $D_1^E, D_2^E, \ldots, D_n^E$ among $RU_1, RU_2, \ldots, RU_m$ after applying the distribution mechanism. $RU_1, RU_2, \ldots, RU_m$ achieve the plain data $D_1, D_2, \ldots, D_n$ via applying decryption over $D_1^E, D_2^E, \ldots, D_n^E$ along with the corresponding keys $PV_1^K, PV_2^K, \ldots, PV_n^K$, which are obtained via communicating with CF. If any $RU_j \in \mathbb{RU}$ leaks the received data to any $TP_1, TP_2, \ldots, TP_k$ then it is called as Guilty User

(G_U). Allocated datasets are compared with the leaked dataset followed by statistical evaluation and the G_U is identified by analyzing the evaluated parameters.

V. Data Encryption and Decryption

Let a data owner DO_i has data D_i, public key PB_i^K, and defines an access policy Θ_i over the attributes. The data object D_i is encrypted with PB_i^K using the following equation:

$$D_i^E = \{\Theta_i, \tilde{D}_i, D_i^{'}, D_{(x,y)}, D_{(x,y)}^{'}, \check{D}_{(x,y),j}\}. \tag{1}$$

$\tilde{D}_i$ and $D_i^{'}$ are computed by DO_i using the following equation:

$$\tilde{D}_i = D_i e(g,g)^{z s_i} \quad D_i^{'} = g^{s_i} \tag{2}$$

whereas $D_{(x,y)}$ and $D_{(x,y)}^{'}$ are calculated using the following equation:

$$D_{(x,y)} = h^{q_{(x,y)}(0)} \quad D_{(x,y)}^{'} = H(att(x,y))^{q_{(x,y)}(0)} \tag{3}$$

where Θ_i is a policy to access the data D_i, e is the bilinear map denoted by $e : G_O \times G_O \rightarrow G_\Theta$, g be the generator of G_O, and G_O be the bilinear group of prime order p. For each node (x,y) (including the leaf nodes) in Θ_i, a polynomial $q_{(x,y)}$ must be chosen from starting with the root node. s, r, z are the random numbers, whereas a, b are the random exponents that belong to Z_p and $h = g^b$. A hash function H is used to map the attributes. att is a function that denotes the attributes within the tree associated with the leaf nodes. $\check{D}_{(x,y),j}$ for each node (x,y) and $\forall j = 1,2,\ldots,k^*$ is computed to obtain the threshold gate set where k^* is the level of the tree. $DO_i \in \mathbb{DO}$ do not rely on CSP for data access control and $RU_1, RU_2, \ldots, RU_m$ get different decryption privileges according to their different attributes. The encrypted data D_i^E including the access structure Θ_i implicitly is uploaded to CSP by $DO_1, DO_2, \ldots, DO_n$. The attribute sets $\Upsilon_j; j = 1,2,\ldots,m$ are obtained by CSP in the encrypted form along with $SC_1^K, SC_2^K, \ldots, SC_m^K$ from $RU_1, RU_2, \ldots, RU_m$ where SC_j^K denotes the secret key of RU_j related to the attribute set Υ_j. RU_j can decrypt the ciphertext D_i^E and gets the original data D_i only if Υ_j satisfies Θ_i. D_i^E is decrypted using the following equation:

$$D_i = \tilde{D}_i / \left(e(h^{s_i}, g^{(a+r)/b}) / e(g,g)^{r s_i}\right). \tag{4}$$

Fig. 2 portrays the encryption and decryption mechanism of MLPAM where $RU_1, RU_2, \ldots, RU_m$ can access $D_1, D_2, \ldots, D_n$ by decrypting the data $D_1^E, D_2^E, \ldots, D_n^E$ using the corresponding keys $PV_1^K, PV_2^K, \ldots, PV_n^K$ after matching of users' attributes with the access policy determined by $DO_1, DO_2, \ldots, DO_n$.

VI. Data Classification

To enhance the accuracy and efficiency of the computations while preserving privacy, encrypted data $\mathbb{D}^{\mathbb{E}} = \{D_1^E, D_2^E, \ldots, D_n^E\}$ from Cloud Storage (CS) is transformed into noised data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ using ϵ-differential privacy [29], [30]. CSP generates a noise vector $\mathbb{N} = \{N_1, N_2, \ldots, N_n\}$ using a distribution that is encrypted using public keys $\mathbb{PB}^{\mathbb{K}} = \{PB_1^K, PB_2^K, \ldots, PB_n^K\}$

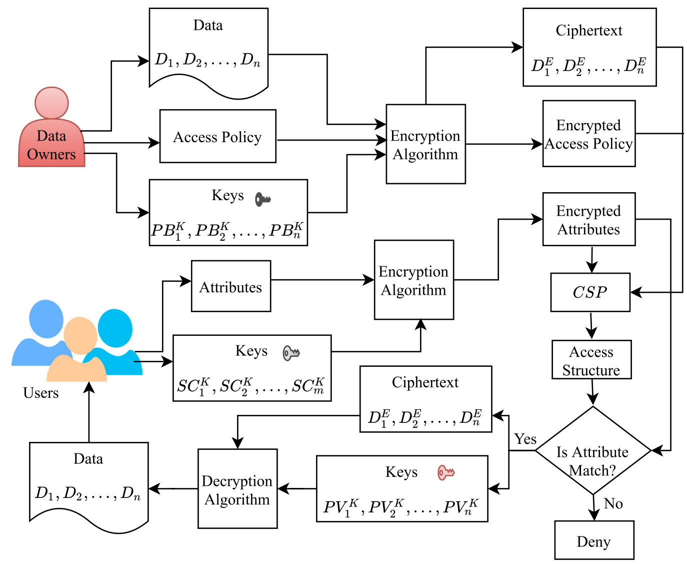

Fig. 2. Data encryption and decryption in MLPAM.

correspondingly and encrypted noise vector $\mathbb{N}^{\mathbb{E}} = \{N_1^E, N_2^E, \ldots, N_n^E\}$ is obtained. The generated data $\mathbb{N}^{\mathbb{E}} = \{N_1^E, N_2^E, \ldots, N_n^E\}$ is added in the corresponding data $\mathbb{D}^{\mathbb{E}} = \{D_1^E, D_2^E, \ldots, D_n^E\}$ as $D_i^{N_i} = D_i^E + N_i^E$ where $i \in [1,n]$ and the resulted data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ are passed to CF. Using the corresponding private keys $\mathbb{PV}^{\mathbb{K}} = \{PV_1^K, PV_2^K, \ldots, PV_n^K\}$, CF decrypts the data $\mathbb{D}^{\mathbb{N}} = \{D_1^{N_1}, D_2^{N_2}, \ldots, D_n^{N_n}\}$ and attains plain noised data $\mathbb{D}^{\mathbb{N}'} = \{D_1^{N_1'}, D_2^{N_2'}, \ldots, D_n^{N_n'}\}$ that undergoes preprocessing to achieve the preprocessed data $\hat{\mathbb{D}}^{\mathbb{N}'} = \{\hat{D}_1^{N_1'}, \hat{D}_2^{N_2'}, \ldots, \hat{D}_n^{N_n'}\}$. Let ith decrypted data $D_i^{N_i'}$ consists of Δ attributes $\mathbb{A} = \{A_1, A_2, \ldots, A_\Delta\}$, it is preprocessed by using the normalization function given in (5), where A_t is the training sample, μ and σ are the mean and the standard deviation of the training sample, respectively

$$\hat{D}_i^{N_i'} = \frac{(A_t - \mu)}{\sigma}. \tag{5}$$

It is known that the data $\hat{\mathbb{D}}^{\mathbb{N}'} = \{\hat{D}_1^{N_1'}, \hat{D}_2^{N_2'}, \ldots, \hat{D}_n^{N_n'}\}$ belongs to $n^* \leq n$ classes $\mathbb{C} = \{\mathbb{C}_1, \mathbb{C}_2, \ldots, \mathbb{C}_{n^*}\}$ where $\cup_{i=1}^{n^*} \mathbb{C}_i = \mathbb{D}$ and $\mathbb{C}_i \cap \mathbb{C}_j = \emptyset \forall i,j = 1,2,\ldots,n^* \wedge i \neq j$. The data $\hat{\mathbb{D}}^{\mathbb{N}'} = \{\hat{D}_1^{N_1'}, \hat{D}_2^{N_2'}, \ldots, \hat{D}_n^{N_n'}\}$ is divided into training data $\hat{\mathbb{D}}_{t'}^{\mathbb{N}'} = \{\hat{D}_{t,'1}^{N_1'}, \hat{D}_{t,'2}^{N_2'}, \ldots, \hat{D}_{t,'n^{**}}^{N_{n^{**}}'}\}$ and testing data $\hat{\mathbb{D}}_{t''}^{\mathbb{N}'} = \{\hat{D}_{t,''1}^{N_1'}, \hat{D}_{t,''2}^{N_2'}, \ldots, \hat{D}_{t,''n^{***}}^{N_{n^{***}}'}\}$ satisfying the following properties: 1) $\hat{\mathbb{D}}_{t'}^{\mathbb{N}'} \cup \hat{\mathbb{D}}_{t''}^{\mathbb{N}'} = \hat{\mathbb{D}}^{\mathbb{N}'}$; 2) $\hat{\mathbb{D}}_{t'}^{\mathbb{N}'} \cap \hat{\mathbb{D}}_{t''}^{\mathbb{N}'} = \emptyset$; 3) $n^{**}, n^{***} \leq n$; and 4) $n^{**} = n \times x$, $n^{***} = n \times (1-x)$, where $x \in \mathbb{Z} \wedge 0 \leq x \leq 1$ for the Classification Model (CM). The training data $\hat{D}_{t,'1}^{N_1'}, \hat{D}_{t,'2}^{N_2'}, \ldots, \hat{D}_{t,'n^{**}}^{N_{n^{**}}'}$ is used to train CM utilizing machine learning algorithms, whereas the testing data $\hat{D}_{t,''1}^{N_1'}, \hat{D}_{t,''2}^{N_2'}, \ldots, \hat{D}_{t,''n^{***}}^{N_{n^{***}}'}$ is used to evaluate the accuracy of CM. During the testing process, data objects $\hat{D}_{t,''1}^{N_1'}, \hat{D}_{t,''2}^{N_2'}, \ldots, \hat{D}_{t,''n^{***}}^{N_{n^{***}}'}$ are given to CM to identify their classes. CM analyzes $\hat{D}_{t,''1}^{N_1'}, \hat{D}_{t,''2}^{N_2'}, \ldots, \hat{D}_{t,''n^{***}}^{N_{n^{***}}'}$ and produces

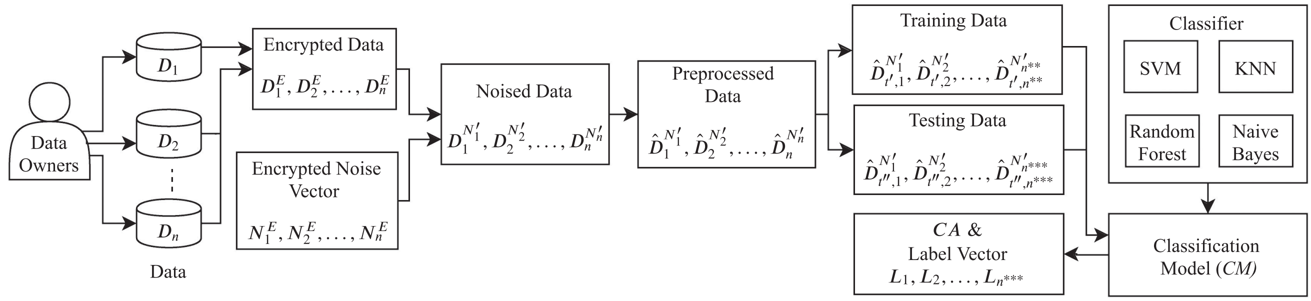

Fig. 3. Workflow of data classification.

a *Label Vector* $\mathbb{L} = \{L_1, L_2, \ldots, L_{n^{***}}\}$ as an output, where $L_{i'} \in \mathbb{L}$ specifies $\mathbb{C}_i \in \mathbb{C}$ to which $\hat{D}_{t,''i'}^{N_{i'}'} \in \hat{\mathbb{D}}_{t''}^{N'}$ pertains. The *Classification Accuracy* (CA) is measured using (6), where CN signifies the number of correctly classified items and TN implies the total number of test items. The stepwise process for classification of data $D_1, D_2, \ldots, D_n$ is depicted in Fig. 3

$$CA = \frac{CN}{TN}. \tag{6}$$

VII. Data Distribution

Let $\mathbb{D} = \{D_1, D_2, \ldots, D_n\}$ is the dataset consisting n independent data objects in the relational form owned by n different owners $\mathbb{DO} = \{DO_1, DO_2, \ldots, DO_n\}$, which are stored on cloud storage CS in encrypted form by CSP. The stepwise process along with essential blocks for the data distribution is presented in Fig. 4.

A. Demanding Usersets and Sensitivity Computation

The m distinct users $RU_1, RU_2, \ldots, RU_m$ send the *demanding datasets* $\mathbb{Y}_1, \mathbb{Y}_2, \ldots, \mathbb{Y}_m$ where $\mathbb{Y}_j \subseteq \mathbb{D}$, $\cup_{j=1}^{m} \mathbb{Y}_j \subseteq \mathbb{D}$, and $\mathbb{Y}_j \cap \mathbb{Y}_{j'} \subseteq \mathbb{D} \forall j, j' = 1, 2, \ldots, m \wedge j \neq j'$ that are used to compute *demanding usersets* $\mathbb{Z}_1, \mathbb{Z}_2, \ldots, \mathbb{Z}_n$ for each data object D_i using $\mathbb{Z}_i = \{RU_j | D_i \in \mathbb{Y}_j\} \forall i = 1, 2, \ldots, n$.

Let the object D_i has Ω tuples $\mathbb{T} = \{T_1, T_2, \ldots, T_\Omega\}$ and Δ attributes $\mathbb{A} = \{A_1, A_2, \ldots, A_\Delta\}$, out of which Δ^* are sensitive attributes $\mathbb{S} = \{S_1, S_2, \ldots, S_{\Delta^*}\}$ and Δ^{**} are quasi-identifier attributes $\mathbb{Q} = \{Q_1, Q_2, \ldots, Q_{\Delta^{**}}\}$. A *Grading Function* assigns a weight $0 \leq W \leq 1$ to object D_i, sensitive attributes $S_\alpha; (\alpha = 1, 2, \ldots, \Delta^*)$ of D_i, quasi-identifier attributes $Q_\beta; (\beta = 1, 2, \ldots, \Delta^{**})$ of D_i, every possible value V_γ of S_α, and each possible value V_δ of Q_β as per their sensitivity by satisfying the following properties: 1) $0 \leq W_{D_i} \leq 1$; 2) $W_{S_1} + W_{S_2} + \cdots + W_{S_\alpha} = 0.9$; 3) $W_{Q_1} + W_{Q_2} + \cdots + W_{Q_\beta} = 0.1$; 4) $0 \leq W_{V_\gamma}^{\alpha} \leq 1$; 5) $0 \leq W_{V_\delta}^{\beta} \leq 0.1$, where $W_{V_\gamma}^{\alpha}$ and $W_{V_\delta}^{\beta}$ signify the weight of possible value of S_α and Q_β attribute, respectively. For a large number of possible values $V_\gamma \in S_\alpha$ or $V_\delta \in Q_\beta$, W is assigned to V_γ or V_δ by classifying the domain of S_α or Q_β relatively. The *cumulative weight* $W^* \in [0, 1] \wedge W^* \in \mathbb{R}_{\geq 0}$ for the possible values of S_α and Q_β attribute is computed using $W_{V_\gamma}^{\alpha*} = S_\alpha \times W_{V_\gamma}^{\alpha}$ and $W_{V_\delta}^{\beta*} = Q_\beta \times W_{V_\delta}^{\beta}$, respectively. The *Formative Tuple Sensitivity Measure* $TSM_\eta^F \in \mathbb{R}_{\geq 0}$ is computed for each tuple $\eta \in$ $D_i (1 \leq \eta \leq \Omega)$ using (7) where $W_\eta^{\alpha*}$ and $W_\eta^{\beta*}$ represent the weight assigned to the ηth tuple of S_αth and Q_βth attribute, respectively

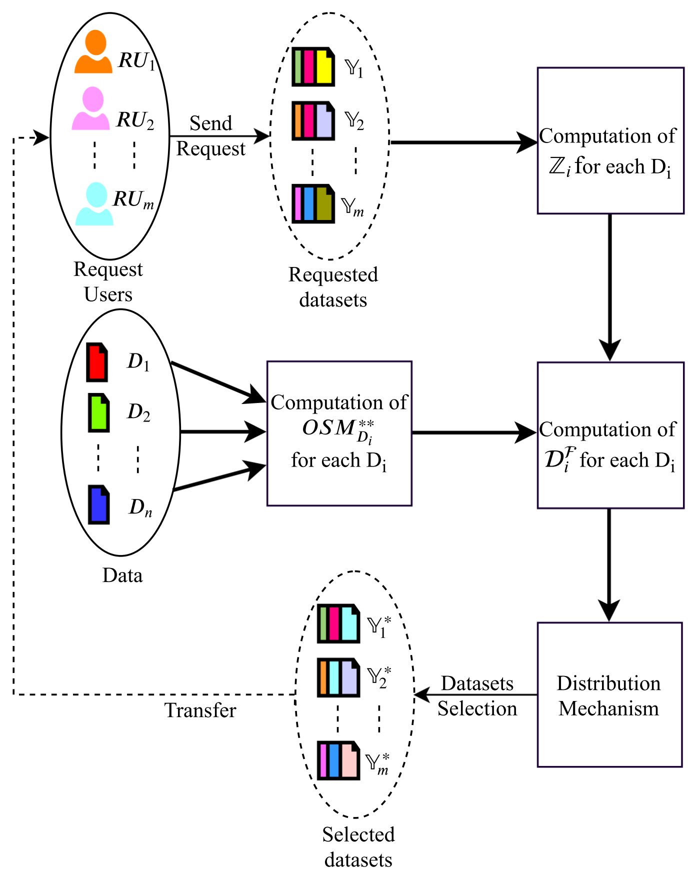

Fig. 4. Distribution process.

$$TSM_\eta^F = \sum_{\alpha=1}^{\Delta^*} W_\eta^{\alpha*} + \sum_{\beta=1}^{\Delta^{**}} W_\eta^{\beta*} \quad \forall \eta = 1, 2, \ldots, \Omega. \tag{7}$$

The *Cumulative Tuple Sensitivity Measure* $TSM_\eta^* \in \mathbb{R}_{\geq 0}$ is obtained by computing the ratio of TSM_η^F to the *Identity Coefficient* $\mathcal{I}_\eta^{\mathcal{C}} \in [1, \Omega] \wedge \mathcal{I}_\eta^{\mathcal{C}} \in \mathbb{Z}^+$ as shown in (8), where $\mathcal{I}_\eta^{\mathcal{C}}$ is the number of repetitions as a unit of quasi-identifier's values

$V_\delta^1, V_\delta^2, V_\delta^{\Delta^{**}}$ of tuple η in D_i

$$TSM_\eta^* = \frac{TSM_\eta^F}{\mathcal{I}_\eta^{\mathcal{C}}}. \tag{8}$$

The evaluation of *Formative Object Sensitivity Measure* $OSM_{D_i}^F \in \mathbb{R}_{\geq 0}$ followed by the assessment of *Cumulative Object Sensitivity Measure* $OSM_{D_i}^* \in \mathbb{R}_{\geq 0}$ is performed in (9) and (10), respectively

$$OSM_{D_i}^F = \sum_{\eta=1}^{\Omega} TSM_\eta^* \tag{9}$$

$$OSM_{D_i}^* = W_{D_i} \times OSM_{D_i}^F. \tag{10}$$

The *Standardized Object Sensitivity Measure* $OSM_{D_i}^{**} \in \mathbb{R}_{\geq 0} \wedge OSM_{D_i}^{**} \in [0,1]$ is calculated using (11) where $\rho^{\mathcal{C}} \in \mathbb{R}_{\geq 0} \wedge \rho^{\mathcal{C}} \in [0,1]$ is the *Worth Coefficient*. $OSM_{D_i}^{**}$ and $\rho^{\mathcal{C}}$ imply the sensitivity of the object D_i and worth of the object having maximum sensitivity, respectively

$$OSM_{D_i}^{**} = \frac{OSM_{D_i}^*}{\max_{i=1,2,\ldots,n} OSM_{D_i}^* + (1 - \rho^{\mathcal{C}})}. \tag{11}$$

B. Distribution Mechanism

The objects $D_1, D_2, \ldots, D_n$ are categorized into $\lambda \leq n$ classes $\mathbb{C}^* = \{\mathbb{C}_1^*, \mathbb{C}_2^*, \ldots, \mathbb{C}_\lambda^*\}$ having property $\cup_{i=1}^{\lambda} \mathbb{C}_i^* = \mathbb{D}$, $\mathbb{C}_i^* \cap \mathbb{C}_{i'}^* = \emptyset \forall i, i' = 1, 2, \ldots, \lambda \wedge i \neq i' \wedge \mathbb{C}_{i,i'}^* \subseteq \mathbb{D}$ by classifying the range of $OSM_{D_i}^{**}$ completely and disjointly. The range of the ith category is $(i-1)\zeta \leq \mathbb{C}_i^* < i\zeta$ and $(\lambda - 1)\zeta \leq \mathbb{C}_\lambda^* \leq 1$ where $\zeta = \frac{1}{\lambda}$. A *Sensitivity Index* $\mathcal{S}_i^{\mathcal{I}} \in \mathbb{R}_{\geq 0} \wedge \mathcal{S}_i^{\mathcal{I}} \in [0,1]$ using (12) and a *Limit Coefficient* $\mathcal{L}_i^{\mathcal{C}} \in \mathbb{Z}_{\geq 0} \wedge \mathcal{L}_i^{\mathcal{C}} \in [0, 100]$ are assigned to the ith category $\mathbb{C}_i^* \in \mathbb{C}^* \forall i = 1, 2, \ldots, \lambda$

$$\mathcal{S}_i^{\mathcal{I}} = \begin{cases} 1, & i = \lambda \\ Round\left[\left((i-1) \times \left(\frac{1}{\lambda - 1}\right)\right), 2\right], & \text{otherwise.} \end{cases} \tag{12}$$

$\mathcal{L}_i^{\mathcal{C}}$ indicates the lower limit in the percentage of users for D_i allocation. $\mathcal{S}^{\mathcal{I}}$ and $\mathcal{L}^{\mathcal{C}}$ assigned to class $\mathbb{C}_i^*$ are $\mathcal{S}^{\mathcal{I}}$ and $\mathcal{L}^{\mathcal{C}}$ of all $D_i \in \mathbb{C}_i^*$. The *Distribution Factor* $\mathcal{D}_i^{\mathcal{F}} \in [0, |\mathbb{RU}|] \wedge \mathcal{D}_i^{\mathcal{F}} \in \mathbb{Z}_{\geq 0}$ for each D_i is computed employing (13) that defines the count of users for D_i allocation

$$\mathcal{D}_i^{\mathcal{F}} = \lceil * \min\left(1, \left(1 - \mathcal{S}_i^{\mathcal{I}} + \frac{\mathcal{L}_i^{\mathcal{C}}}{100}\right)\right) \times |\mathbb{Z}_i|. \tag{13}$$

CSP selects the datasets $\mathbb{Y}_1^*, \mathbb{Y}_2^*, \ldots, \mathbb{Y}_m^*$ for distribution among $RU_1, RU_2, \ldots, RU_m$ that minimizes the risks associated with data leakage from $\prod_{i=1}^{n} \binom{\mathbb{Z}_i}{\mathcal{D}_i^{\mathcal{F}}}$ possible datasets allocations. The operational summary for MLPAM data distribution is delineated in Algorithm 1. *Object Flag* $\mathcal{O}_j^{\mathcal{F}}$ is initiated to 1 for every RU_j that indicates the request of RU_j to be processed. The *user selection* is followed by *object allocation* and the process is repeated until all the requests are processed. The steps for user selection are depicted in Algorithm 2, where the $\aleph^{th}$ user is selected on a rotation basis having $\mathcal{O}_\aleph^{\mathcal{F}}$ less or equal to the number of requests by $RU_\aleph$. For the selected user $\aleph$, initiating from *object-identity* equals object flag to the total number of requests, an object D_i is allocated with a positive data allocation factor, which is reduced by 1 after the object allocation.

Algorithm 1: MLPAM Distribution Mechanism.

Input: $n, m, \mathbb{Y}_1, \mathbb{Y}_2, \ldots, \mathbb{Y}_m, \mathbb{Z}_i, \mathcal{D}_i^{\mathcal{F}} \forall i = 1, 2, \ldots, n$
Output: $\mathbb{Y}_1^*, \mathbb{Y}_2^*, \ldots, \mathbb{Y}_m^*$
Initialize: $\mathbb{Y}_1^* \leftarrow \emptyset, \mathbb{Y}_2^* \leftarrow \emptyset, \ldots, \mathbb{Y}_m^* \leftarrow \emptyset$, $\mathcal{O}_j^{\mathcal{F}} \leftarrow 1 \forall j = 1, 2, \ldots, m$
while $\sum_{i=1}^{n} \mathbb{Z}_i > 0$**do**
  $\aleph \leftarrow$ SELECT_USER$m, |\mathbb{Y}|, \mathcal{O}^{\mathcal{F}}$
  for $\uplus = \mathcal{O}_\aleph^{\mathcal{F}}; \uplus \leq |\mathbb{Y}_\aleph|; \uplus + +$ **do**
    if $\mathcal{D}^{\mathcal{F}}[\mathbb{Y}[\aleph][\uplus]] > 0$ **then**
      $\mathbb{Y}_\aleph^* = \mathbb{Y}_\aleph^* \cup \{\mathbb{Y}[\aleph][\uplus]\}; \mathcal{D}^{\mathcal{F}}[\mathbb{Y}[\aleph][\uplus]] - -; \mathcal{O}_\aleph^{\mathcal{F}} + +; \sum_{i=1}^{n} \mathbb{Z}_i - -$
      BREAK
    else
      $\mathcal{O}_\aleph^{\mathcal{F}} + +; \sum_{i=1}^{n} \mathbb{Z}_i - -$
    end if
  end for
end while

Algorithm 2 MLPAM User Selection

Initialize: $\aleph^* = 1, \aleph^{**} = 0$
FunctionSELECT_USER$m, |\mathbb{Y}|, \mathcal{O}^{\mathcal{F}}$
  $(\aleph^* == m + 1) \Rightarrow (\aleph^* = 1)$
  If $(\mathcal{O}_{\aleph^*}^{\mathcal{F}} \leq |\mathbb{Y}_{\aleph^*}|)$ **?** $\aleph^{**} = \aleph^*; \aleph^* + +$ **:** $\aleph^* + +; \aleph^{**} =$ SELECT_USER$m, |\mathbb{Y}|, \mathcal{O}^{\mathcal{F}}$
  return $\aleph^{**}$
end function

VIII. Guilty User Detection

If any RU_j leaks the dataset $\ell \subseteq \mathbb{D}$ to an unauthorized party $TP_k \in \mathbb{TP}$, then the following parameters are calculated for the identification of Guilty User (G_U): 1) *Probability* (P_b) *of* RU_j *for being* G_U ($P_b\{G_{U_j}|\ell\}) \forall j = \{1, 2, \ldots, m\}$; 2) *difference function* ($\varpi_{(j,k)}^*(G_U)) \forall j, k = \{1, 2, \ldots, m\}$; 3) *average success rate* ($\overline{\varpi}^*$); 4) *detection rate* ($\min \varpi^*$) using (14)–(17), respectively [24], where θ is the probability of stealing any $D_i \in \ell$ by $TP_k \in \mathbb{TP}$ either from $DO_i \in \mathbb{DO}$ or $RU_j \in \mathbb{RU}$ having access to D_i. It is believed that the obtained $D_i \in \mathbb{D}$ by $TP_k \in \mathbb{TP}$ in case of 1) leakage through $DO_{i'} \in \mathbb{DO}$ where $i' \neq i$, CSP, and CF 2) stealing from cloud1, cloud2, and communication among $DO_i \in \mathbb{DO}, CSP \in CP, CF \in CP$, and $RU_j \in \mathbb{RU}$ will not be usable since it is in encrypted D_i^E (using distinct public keys $PB_1^K, PB_2^K, \ldots, PB_n^K$) or noised D_i^N form

$$P_b\left\{G_{U_j}|\ell\right\} = 1 - \prod\nolimits_{\forall D_i \in \left(\ell \cap \mathbb{Y}_j^*\right)} \left(1 - \frac{(1-\theta)}{\mathcal{D}_i^{\mathcal{F}}}\right) \tag{14}$$

$$\varpi_{(j,k)}^*(G_U) = P_b\left\{G_{U_j}|\mathbb{Y}_j^*\right\} - P_b\left\{G_{U_k}|\mathbb{Y}_j^*\right\} \tag{15}$$

$$\overline{\varpi}^* = \frac{\sum_{\substack{j,k=\{1,2,\ldots,m\} \\ j \neq k}} \varpi_{(j,k)}^*(G_U)}{m(m-1)} \tag{16}$$

$$\min \varpi^* = \min_{\substack{j,k=\{1,2,\ldots,m\} \\ j \neq k}} \varpi_{(j,k)}^*(G_U). \tag{17}$$

A $RU_j \in \mathbb{RU}$ fulfilling the criteria $\max_{j=1,2,\ldots,m} P_b\{G_{U_j}|\ell\} - P_b\{G_{U_j}|\ell\} \leq \mathcal{T}_{\mathcal{H}}$ is declared as G_U where $\mathcal{T}_{\mathcal{H}}$ signifies the threshold value. Furthermore, the parameters *Detection*

Accuracy (DA), *Detection Precision* (DP), *Detection Recall* (DR), and *Detection Specificity* (DS) are computed using (18)–(21), respectively, to prove the effectiveness of MLPAM by classifying the users as Guilty User (G_U) and Nonguilty User (NG_U) and via assessing the outcome of MLPAM, whether the user is guilty or nonguilty. In these equations, $|NG_U^A = NG_U^E|$ specifies the number of test cases having Actual Nonguilty Users (NG_U^A) equal to the Estimated Nonguilty Users (NG_U^E), $|G_U^A = G_U^E|$ is the term representing the count of test cases where Actual Guilty Users (G_U^A) are identical to the Estimated Guilty Users (G_U^E), $|G_U^A = NG_U^E|$ demonstrates the number of test cases in which actual guilty users are estimated as nonguilty users, whereas $|NG_U^A = G_U^E|$ deals with the number of test cases when actual nonguilty users are estimated as guilty users

$$DA = \frac{|NG_U^A = NG_U^E| + |G_U^A = G_U^E|}{|NG_U^A = NG_U^E| + |G_U^A = NG_U^E| + |G_U^A = G_U^E| + |NG_U^A = G_U^E|} \tag{18}$$

$$DP = \frac{|G_U^A = G_U^E|}{|G_U^A = G_U^E| + |NG_U^A = G_U^E|} \tag{19}$$

$$DR = \frac{|G_U^A = G_U^E|}{|G_U^A = G_U^E| + |G_U^A = NG_U^E|} \tag{20}$$

$$DS = \frac{|NG_U^A = NG_U^E|}{|NG_U^A = NG_U^E| + |NG_U^A = G_U^E|}. \tag{21}$$

The computational and space complexities are $O(\max_{1\leq j\leq m} |\Upsilon_j|)$, $O(1)$; $O((n^{**})^3)$, $O((n^{**})^2)$; $O(\sum_{j=1}^{m} |\mathbb{Y}_j|)$, $O(\sum_{j=1}^{m} |\mathbb{Y}_j|)$; $O(m^2 \max_{1\leq j\leq m} |\mathbb{Y}_j^*|)$, $O(m^2)$ for various phases data encryption and decryption, data classification, data distribution, and guilty user detection of MLPAM, respectively. The complexity analysis of MLPAM implies that the data is protected by the aid of endurable time and space, which establishes its potency.

IX. Performance Evaluation

A. Experimental Setup

A series of experiments have been conducted over four different datasets Glass, Iris, Wine, and Balance Scale with 10, 4, 13, 4 attributes and 214, 150, 178, 625 instances, respectively, that are taken from the UCI Machine Learning Repository [31] to train CM using machine learning algorithms. The four different classifiers Support Vector Machine (SVM), Random Forest, K-Nearest Neighbor (K-NN), and Naive Bayes have been used to train CM over the training data. These experiments are performed on Intel Core i7-7700 CPU@3.60 GHz eight-core processor with Ubuntu 14.04-amd64 operating system, 8 GB RAM machine using Python 2.73 for encryption and machine learning, and C++ 12.1 for guilty user detection.

B. Computation Time for Encryption/Decryption

The computation time for the encryption and decryption processes over the various datasets is shown in Fig. 5(a) and (b), respectively. It is observed that the time costs of the encryption and decryption grow linearly with respect to the number of attributes associated with the access policy. Furthermore, the comparison among different datasets has been performed in terms of encryption time (T_E) and decryption time (T_D). It is found that both encryption and decryption time varies with respect to the dataset for the fixed number of attributes. For instance, BS has maximum encryption time for 2 and 5 attributes, but it is not true for other numbers of attributes. However, Iris has minimum encryption time for 2 and 3 attributes, which is not true for 4 and 5 attributes. BS has minimum decryption time for all number (2–5) of attributes, whereas wine has maximum decryption time for all attributes excluding 3. CP-ABE is effective to reduce the computation cost because the user can determine the matching result without the interaction with the initiator.

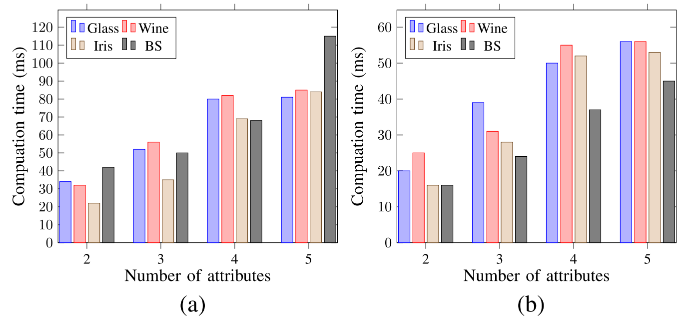

Fig. 5. Computation time (ms) for various datasets: (a) T_E and (b) T_D.

C. Accuracy of Classification Model

From the complete dataset, 9/10 of the data is used as training data, whereas the rest of the data is taken as testing data. The machine learning is performed over both clean and noised data. To generate the noised data, we have used the Gaussian and the randomly generated mechanisms with the value of privacy level 0.1. The outcome of the noised data is compared against the clean data to find the variations. Furthermore, a comparison is performed among Gaussian and Random noised data to find the superior one. The outcome of CM is measured using the testing data and the Classification Accuracy (CA) is computed. Fig. 6(a)–(d) shows the CA achieved by CM of MLPAM over Clean, Gaussian noised, and Random noised data and also depicts the comparison among Glass, Iris, Wine, and BS datasets for SVM, Random Forest, KNN, and Nave Bayes classifier, respectively. It is observed that CA of the noised data is less compared to the clean data in the case of all the four classifiers because of the noise addition but still, CA is nearly equal for noised data and also provides more security compared to the clean data. Furthermore, out of the two noised added data, the Gaussian noised data outperforms over the Random noised data in the case of all the four classifiers. The performance of datasets and classifiers in descending order are Iris, Wine, BS, Glass; and SVM, Random Forest, Naive Bayes, K-NN, respectively. Out of the four datasets, the Iris dataset outperforms the rest of the three dataset for all the four classifiers. For the Clean and Gaussian noised data, the SVM classifier outperforms the

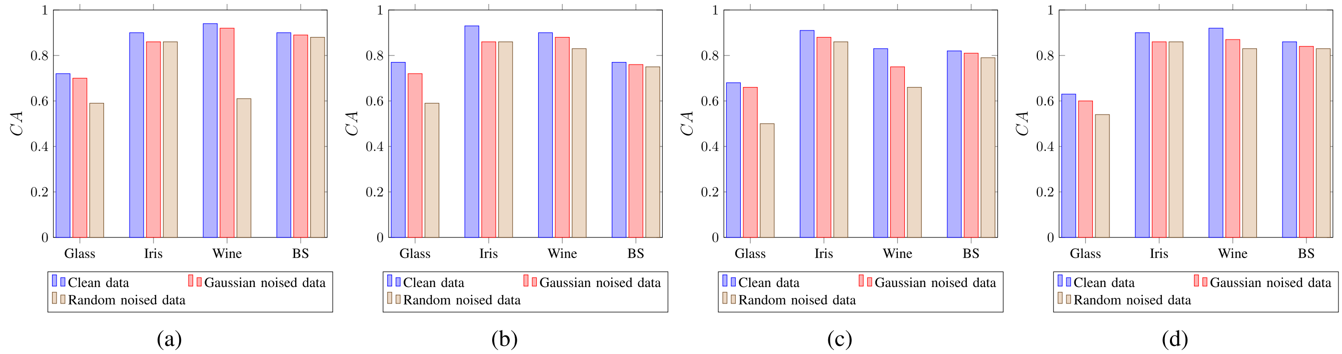

Fig. 6. Accuracy of CM in MLPAM for (a) SVM, (b) Random Forest, (c) K-NN, and (d) Naive Bayes classifier.

rest three, whereas for the Random noised data, Naive Bayes outperforms over the other classifiers. As an aggregate, the SVM classifier outperforms the rest three classifiers in MLPAM due to the application of kernel trick and considerable optimal margin gap between separating hyperplanes during classification, which results in better performance.

TABLE II
AVERAGE PROBABILITY COMPARISONS FOR DIFFERENT θ

$\mathcal{W}_{\mathcal{F}}$	$\sum_{j=1}^{m} P_b\left\{G_{U_j}\lvert\mathbb{Y}_j^*\right\}/\lvert\mathbb{RU}\rvert$					
	$\theta = 0.3$			$\theta = 0.5$		
	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM
1	0.999949	0.999963	0.999985	0.998442	0.998731	0.999184
1.44	0.999998	0.999999	1	0.999855	0.999885	0.999932
1.975	0.999984	0.999990	1	0.999539	0.999704	0.999961
2.59	0.999999	0.999999	1	0.999909	0.999938	0.999983
2.89	0.999998	0.999999	1	0.99992	0.999939	0.999969
3.12	1	1	1	0.999961	0.999968	0.99998
3.38	0.999995	0.999997	1	0.999852	0.999905	0.999987
3.84	0.999998	0.999999	1	0.999921	0.999942	0.999975
4.435	0.999999	0.999999	1	0.999936	0.999955	0.999984
4.97	0.999999	0.999999	1	0.999939	0.999952	0.999972
5.55	0.999999	0.999999	1	0.999958	0.999967	0.99998
5.98	1	1	1	0.999966	0.999970	0.999976
6.485	0.999999	0.999999	1	0.999961	0.999965	0.99997
6.935	0.999999	0.999999	1	0.999961	0.999966	0.999975
7.42	1	1	1	0.999965	0.999969	0.999976
7.915	0.999999	0.999999	1	0.99996	0.999965	0.999974
8.345	0.999999	0.999999	1	0.999963	0.999967	0.999973
8.905	0.999999	0.999999	1	0.999963	0.999967	0.999972
9.4	0.999999	0.999999	0.999999	0.999963	0.999964	0.999961
9.73	1	1	1	0.999965	0.999969	0.999976
Avg	0.999996	0.999997	0.999999	0.999845	0.999879	0.999933

D. Parameters for Guilty User Detection

Five hundred data objects are shared among ten users and the requests of $RU_1, RU_2, \ldots, RU_m$ are generated randomly in MLPAM. $\lambda = 11, \mathcal{L}_i^{\mathcal{C}} = 0.1 \forall \mathbb{C}_i^* \in \mathbb{C}^*, \theta \in \{0, 0.1, 0.3, 0.5\}$, and $\mathcal{T}_{\mathcal{H}} = 0.0E+00, 0.0E+00, 1.0E-06, 7.75E-05$ has taken into consideration throughout the experiments. The performance is assessed with respect to the *Weight Factor*, which is calculated as $\mathcal{W}_{\mathcal{F}} = \frac{\sum_{j=1}^{m} \mathbb{Y}_j}{|\mathbb{D}|}$. The experimental results are compared with GAM [23] and DT-ILIS [24] via implementing these on the same platform.

The average probability $\left(\frac{\sum_{j=1}^{m} P_b\{G_{U_j}|\mathbb{Y}_j^*\}}{|\mathbb{RU}|}\right)$ when all RU_j have leaked their allocated datasets $\mathbb{Y}_1^*, \mathbb{Y}_2^*, \ldots, \mathbb{Y}_m^*$, Average Success Rate ($\overline{\varpi}^*$), and Detection Rate ($\min \varpi^*$) for the proposed and the comparable schemes are computed with respect to $\mathcal{W}_{\mathcal{F}}$ at different $\theta = 0, 0.1, 0.3, 0.5$ in Tables II–IV, respectively. $\frac{\sum_{j=1}^{m} P_b\{G_{U_j}|\mathbb{Y}_j^*\}}{|\mathbb{RU}|} = 1$ is noted for all three GAM [23], DT-ILIS [24], and MLPAM $\forall \mathcal{W}_{\mathcal{F}}$ and $\theta = 0, 0.1$, whereas the values of $\frac{\sum_{j=1}^{m} P_b\{G_{U_j}|\mathbb{Y}_j^*\}}{|\mathbb{RU}|}$ are depicted in Table II for $\theta = 0.3, 0.5$. The following are the observations from Table II.

1) Probability to detect G_U is very high $\forall \mathcal{W}_{\mathcal{F}}, \theta$.
2) $\frac{\sum_{j=1}^{m} P_b\{G_{U_j}|\mathbb{Y}_j^*\}}{|\mathbb{RU}|}$ decreases with respect to θ as chances of stealing rather than leaking the data become high with increment in θ.
3) Probability of the proposed and compared schemes is nearly the same, but, in the proposed scheme, the difference between the probabilities of G_U and NG_U is high (see Tables III and IV) that makes the scheme capable to identify G_U with high accuracy.

In Tables III and IV, the values of $\overline{\varpi}^*$ and $\min \varpi^*$ decrease with respect to $\mathcal{W}_{\mathcal{F}}$ since the overlapping among the datasets $\mathbb{Y}_1^*, \mathbb{Y}_2^*, \ldots, \mathbb{Y}_m^*$ raise with increment in $\mathcal{W}_{\mathcal{F}}$. Furthermore, $\overline{\varpi}^*$ and $\min \varpi^*$ increase with respect to θ due to increment in the probabilities difference of G_U and NG_U. The Detection Accuracy (DA), Detection Precision (DP), Detection Recall (DR), and Detection Specificity (DS) achieved by MLPAM with respect to θ, and comparison against [23], [24] are shown in Fig. 7(a)–(d), relatively. MLPAM secures DA 80%, 97%, 97%, 96%, DP 72%, 95%, 100%, 99%, DR 100%, 99%, 93%, 92%, and DS 60%, 95%, 100%, 99% for $\theta = 0, 0.1, 0.3, 0.5$, respectively, that are very high and acceptable over existing methods [23], [24]. An average for all the three parameters $\frac{\sum_{j=1}^{m} P_b\{G_{U_j}|\mathbb{Y}_j^*\}}{|\mathbb{RU}|}$, $\overline{\varpi}^*$, and $\min \varpi^*$ is calculated individually for each θ. The improvement attained by MLPAM over [23] and [24] for each parameter individually with respect to θ is depicted in Table V. MLPAM achieves relative improvement up to 0.0088064%, 102.83%, 151.31%, 86.54%, 96.08%, 43.08%, 185.71% for $\frac{\sum_{j=1}^{m} P_b\{G_{U_j}|\mathbb{Y}_j^*\}}{|\mathbb{RU}|}$, $\overline{\varpi}^*$, $\min \varpi^*$, DA, DP, DR, and DS, respectively, which supports its effectiveness. Moreover, Table VI depicts the comparison of complexities and it is indicated that both the computational and space complexities are the least in MLPAM as compared to GAM [23] and DT-ILIS [24] due to the effectual data allocation strategy in the proposed model. Additionally, we have performed a comprehensive feature analysis along with a comparison of MLPAM against the state of the artworks [1], [13], [18], [23], [24]. It can be seen from

TABLE III
AVERAGE SUCCESS RATE COMPARISONS FOR DIFFERENT θ

$\mathcal{W}_{\mathcal{F}}$	$\overline{\varpi}^*$											
	$\theta=0$			$\theta=0.1$			$\theta=0.3$			$\theta=0.5$		
	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM
1	4.26425E-01	4.68821E-01	6.19136E-01	4.65160E-01	5.05377E-01	6.51352E-01	5.52823E-01	5.91106E-01	7.19269E-01	6.54565E-01	6.86046E-01	7.91441E-01
1.44	2.44173E-01	2.90987E-01	4.56962E-01	2.81179E-01	3.27278E-01	4.94599E-01	3.72952E-01	4.20423E-01	5.79346E-01	4.94575E-01	5.36789E-01	6.78115E-01
1.975	9.02605E-02	1.27853E-01	2.61136E-01	1.14095E-01	1.53710E-01	2.97499E-01	1.82971E-01	2.30145E-01	3.88077E-01	2.94785E-01	3.43878E-01	5.08234E-01
2.59	2.91696E-02	4.82628E-02	1.15957E-01	4.15896E-02	6.39306E-02	1.45020E-01	8.38887E-02	1.16323E-01	2.24908E-01	1.68513E-01	2.09317E-01	3.45922E-01
2.89	2.12756E-02	3.89133E-02	1.01447E-01	3.00196E-02	5.05604E-02	1.25116E-01	6.13568E-02	9.17428E-02	1.93470E-01	1.30238E-01	1.70372E-01	3.04732E-01
3.12	1.58841E-02	2.63845E-02	6.36131E-02	2.24081E-02	3.57463E-02	8.41590E-02	4.69216E-02	6.98373E-02	1.46555E-01	1.05155E-01	1.39421E-01	2.54139E-01
3.38	5.79726E-03	1.19686E-02	3.38487E-02	9.60572E-03	1.80400E-02	4.86535E-02	2.63282E-02	4.29839E-02	9.87442E-02	7.26829E-02	1.01059E-01	1.96057E-01
3.84	2.64569E-03	7.64570E-03	2.53730E-02	4.68082E-03	1.16148E-02	3.67825E-02	1.47277E-02	2.90569E-02	7.70286E-02	4.71334E-02	7.32714E-02	1.60777E-01
4.435	6.22644E-04	2.97342E-03	1.13080E-02	1.28749E-03	4.82851E-03	1.76811E-02	5.49990E-03	1.41897E-02	4.32817E-02	2.36426E-02	4.26207E-02	1.06156E-01
4.97	1.81591E-04	1.02395E-03	4.01048E-03	4.21779E-04	1.90237E-03	7.27637E-03	2.29826E-03	7.09883E-03	2.31703E-02	1.26940E-02	2.60749E-02	7.08719E-02
5.55	3.84815E-05	4.63603E-04	1.97085E-03	1.08238E-04	8.96766E-04	3.75883E-03	8.37296E-04	3.74015E-03	1.34584E-02	6.33137E-03	1.57301E-02	4.71955E-02
5.98	1.33822E-05	1.91425E-04	8.22667E-04	4.17421E-05	4.14530E-04	1.76761E-03	3.99437E-04	2.10758E-03	7.82614E-03	3.75032E-03	1.04707E-02	3.29695E-02
6.485	3.64439E-06	1.03825E-04	4.59013E-04	1.32783E-05	2.33603E-04	1.03330E-03	1.69908E-04	1.29852E-03	5.07693E-03	2.06665E-03	7.10709E-03	2.39816E-02
6.935	1.28093E-06	5.81012E-05	2.59555E-04	5.18344E-06	1.38185E-04	6.20933E-04	8.17373E-05	8.51877E-04	3.43017E-03	1.21814E-03	5.11217E-03	1.81487E-02
7.42	3.95276E-07	3.28281E-05	1.47817E-04	1.81283E-06	8.24327E-05	3.75053E-04	3.62444E-05	5.63463E-04	2.32850E-03	6.77465E-04	3.69691E-03	1.38055E-02
7.915	1.71268E-07	1.91212E-05	8.63072E-05	8.29886E-07	5.14511E-05	2.28927E-04	1.87836E-05	3.75566E-04	1.57001E-03	4.00196E-04	2.69274E-03	1.03678E-02
8.345	5.30878E-08	8.59690E-06	3.88886E-05	2.94127E-07	2.49159E-05	1.14284E-04	8.24863E-06	2.22693E-04	9.40614E-04	2.18360E-04	1.84417E-03	7.28712E-03
8.905	1.41117E-08	5.94830E-06	2.69877E-05	9.04168E-08	1.73652E-05	8.00663E-05	3.01856E-06	1.59679E-04	6.84150E-04	9.82375E-05	1.37124E-03	5.63303E-03
9.4	4.30221E-09	6.22106E-06	2.82623E-05	3.12577E-08	1.76779E-05	8.17288E-05	1.03644E-06	1.53993E-04	6.66063E-04	3.94567E-05	1.24572E-03	5.28410E-03
9.73	1.40350E-09	5.84718E-06	2.65731E-05	1.18432E-08	1.64655E-05	7.61864E-05	2.88686E-07	1.41988E-04	6.16372E-04	1.31493E-05	1.14175E-03	4.92010E-03
Avg	4.18246E-02	5.12864E-02	8.48329E-02	4.85350E-02	5.87472E-02	9.58138E-02	6.75662E-02	8.11261E-02	1.26522E-01	1.00940E-01	1.18963E-01	1.79302E-01

TABLE IV
DETECTION RATE COMPARISONS FOR DIFFERENT θ

$\mathcal{W}_{\mathcal{F}}$	$\min\varpi^*$											
	$\theta=0$			$\theta=0.1$			$\theta=0.3$			$\theta=0.5$		
	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM	GAM [23]	DT-ILIS [24]	MLPAM
1	6.66667E-02	8.85777E-02	1.11111E-01	9.41970E-02	1.18348E-01	1.48225E-01	1.76194E-01	2.04328E-01	2.48333E-01	3.07548E-01	3.33156E-01	3.90156E-01
1.44	1.56250E-02	4.07886E-02	6.66667E-02	2.69309E-02	5.69988E-02	9.41970E-02	7.14114E-02	1.12277E-01	1.76194E-01	1.68055E-01	2.11309E-01	3.07584E-01
1.975	1.08507E-03	1.08209E-02	2.08333E-02	2.58264E-03	1.71760E-02	3.52299E-02	1.24611E-02	4.22940E-02	8.89556E-02	5.07251E-02	9.63022E-02	1.97748E-01
2.59	5.29819E-05	6.35505E-04	1.23457E-03	1.78939E-04	1.42052E-03	2.95653E-03	1.67448E-03	6.54894E-03	1.41731E-02	1.26410E-02	2.61766E-02	5.63041E-02
2.89	1.33333E-04	8.28402E-04	1.54321E-03	3.89651E-04	1.78349E-03	3.50786E-03	2.87047E-03	7.83620E-03	1.56031E-02	1.78947E-02	3.06404E-02	5.90098E-02
3.12	3.08571E-05	1.82106E-03	3.66211E-03	1.07285E-04	3.28923E-03	7.22572E-03	1.09243E-03	1.06030E-02	2.54784E-02	9.17778E-03	3.12879E-02	8.05008E-02
3.38	6.58012E-05	1.84438E-03	3.67347E-03	2.08247E-04	3.44980E-03	7.46004E-03	1.73389E-03	1.16108E-02	2.70593E-02	1.22736E-02	3.50207E-02	8.56515E-02
3.84	1.26510E-05	3.67498E-04	7.32422E-04	4.68093E-05	8.02021E-04	1.73632E-03	5.53916E-04	3.72503E-03	8.68499E-03	5.53386E-03	1.56233E-02	3.80805E-02
4.435	5.49604E-06	7.34792E-05	1.43393E-04	2.15442E-05	2.01805E-04	4.24813E-04	2.93456E-04	1.40540E-03	3.14460E-03	3.46155E-03	8.38226E-03	1.93348E-02
4.97	2.97148E-06	2.19890E-04	4.42969E-04	1.22623E-05	4.92451E-04	1.08651E-03	1.87117E-04	2.41556E-03	5.90108E-03	2.49224E-03	1.05533E-02	2.84956E-02
5.55	1.04134E-06	3.53323E-05	7.05970E-05	4.82326E-06	9.65779E-05	2.10091E-04	9.16909E-05	7.08362E-04	1.67290E-03	1.49967E-03	4.67817E-03	1.17529E-02
5.98	6.34989E-07	5.50997E-05	1.11111E-04	2.97597E-06	1.41406E-04	3.12663E-04	5.98956E-05	9.11841E-04	2.24437E-03	1.07157E-03	5.18509E-03	1.43410E-02
6.485	1.65420E-07	2.47134E-05	4.99584E-05	8.94993E-07	7.07790E-05	1.57235E-04	2.37169E-05	5.48260E-04	1.36870E-03	5.51000E-04	3.57062E-03	1.02917E-02
6.935	6.58861E-08	7.52567E-06	1.51973E-05	3.90110E-07	2.50238E-05	5.54991E-05	1.23766E-05	2.54640E-04	6.33566E-04	3.41717E-04	2.12187E-03	6.08413E-03
7.42	3.38254E-08	3.86959E-06	7.81429E-06	2.12706E-07	1.30978E-05	2.90385E-05	7.48903E-06	1.45048E-04	3.60205E-04	2.29962E-04	1.37186E-03	3.91352E-03
7.915	8.67690E-09	2.57028E-06	5.20463E-06	6.24784E-08	9.15822E-06	2.04109E-05	2.79981E-06	1.10639E-04	2.79311E-04	1.10248E-04	1.10678E-03	3.32487E-03
8.345	6.27778E-09	8.40479E-07	1.69837E-06	4.65965E-08	3.38930E-06	7.52468E-06	2.06610E-06	5.17782E-05	1.29533E-04	7.99395E-05	6.44044E-04	1.89963E-03
8.905	1.65535E-09	7.35178E-07	1.48953E-06	1.39955E-08	2.88760E-06	6.44263E-06	6.17608E-07	4.32498E-05	1.09931E-04	2.97665E-05	5.35728E-04	1.66190E-03
9.4	4.73229E-10	3.15535E-08	6.35164E-08	4.53450E-09	1.82971E-07	4.03721E-07	8.65908E-08	4.76164E-06	1.20739E-05	3.36011E-06	9.69838E-05	3.05372E-04
9.73	5.13611E-10	2.62506E-08	5.27184E-08	4.85844E-09	1.55635E-07	3.42167E-07	1.45086E-08	4.78280E-06	1.22409E-05	9.06221E-09	1.08654E-04	3.50477E-04
Avg	4.18414E-03	7.30541E-03	1.05153E-02	6.23424E-03	1.02162E-02	1.51425E-02	1.34337E-02	2.02914E-02	3.10175E-02	2.96860E-02	4.08936E-02	6.58395E-02

TABLE V
IMPROVEMENT SECURED BY MLPAM (IN % TERM)

Models	θ	P_b	$\overline{\varpi}^*$	$\min\varpi^*$	DA	DP	DR	DS
GAM [23]	0	0	102.83	151.31	35.59	30.91	4.17	185.71
	0.1	0	97.41	142.89	64.41	69.64	22.22	156.76
	0.3	0.00036	87.26	130.89	86.54	96.08	43.08	163.16
	0.5	0.0088064	77.63	121.79	71.43	83.33	24.32	160.53
DT-ILIS [24]	0	0	65.41	43.94	25	22.03	3.09	93.55
	0.1	0	63.09	48.22	42.65	48.44	16.47	86.27
	0.3	0.00022	55.96	52.86	46.97	56.25	29.17	66.67
	0.5	0.0054007	50.72	61	37.14	50	13.58	70.69

TABLE VI
COMPUTATIONAL AND SPACE COMPLEXITY COMPARISON

Models	Computational Complexity	Space Complexity
GAM [23]	$O\left(nm\sum_{j=1}^{m}\lvert\mathbb{Y}_j\rvert\right)$	$O\left(nm+\sum_{j=1}^{m}\lvert\mathbb{Y}_j\rvert\right)$
DT-ILIS [24]	$O\left(n\sum_{j=1}^{m}\lvert\mathbb{Y}_j\rvert\right)$	$O\left(n+\sum_{j=1}^{m}\lvert\mathbb{Y}_j\rvert\right)$
MLPAM	$O\left(\sum_{j=1}^{m}\lvert\mathbb{Y}_j\rvert\right)$	$O\left(\sum_{j=1}^{m}\lvert\mathbb{Y}_j\rvert\right)$

Table VII that MLPAM is the only model that synchronously supports multiple untrusted entities, owners, users, and ensures as well as significantly enhances several indispensable features simultaneously; therefore, MLPAM performance is better than the existing models [1], [13], [18], [23], [24].

E. Security Analysis

In our system model, all the authorized entities DO_{id}, RU_{id}, CSP, CF, and an unauthorized entity TP_{id} are deemed as untrusted; and MLPAM protected the data from every involved entity. To protect the data of an owner $DO_{id}; id \in [1,n]$ against a) leakage by other owners $DO_{id'}; id' \in [1,n] \wedge id' \neq id$ or CSP to TP_{id} b) stealing by TP_{id} from cloud1$\big/DO_{id}; id \in [1,n]\big/RU_{id}$ or during communication among DO_{id}, CSP, CF,

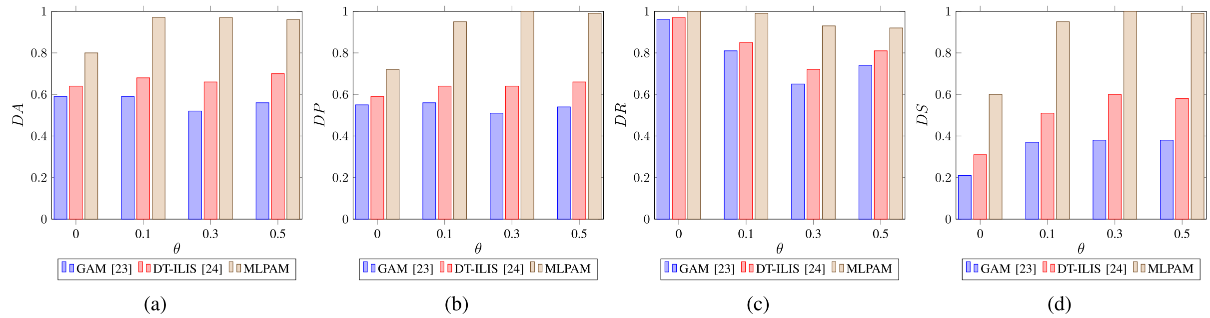

Fig. 7. (a) Detection Accuracy, (b) Detection Precision, (c) Detection Recall, (d) Detection Specificity secured by MLPAM.

TABLE VII
FEATURES ANALYSIS COMPARED TO EXISTING MODELS

Models	$\lvert UE \rvert$	$\lvert DO \rvert$	$\lvert RU \rvert$	T_E (ms)	T_D (ms)	CA	P_b	ϖ^*	$\min \varpi^*$	DA	DP	DR	DS
GAM [23]	single	single	multiple	×	×	×	0.999845	4.18246E-02	4.18414E-03	0.52	0.51	0.81	0.38
DT-ILIS [24]	single	single	multiple	×	×	×	0.999879	5.12864E-02	7.30541E-03	0.66	0.64	0.85	0.60
ABDS [13]	single	single	multiple	140	87	×	×	×	×	×	×	×	×
P-MOD [1]	single	multiple	multiple	1785	359	×	×	×	×	×	×	×	×
PDLM [18]	single	multiple	no	170	100	0.89	×	×	×	×	×	×	×
MLPAM	multiple	multiple	multiple	84	53	0.92	0.999933	8.48329E-02	1.05153E-02	0.97	1	0.99	1

and RU_{id}, the data is encrypted with a distinct key and shared in encrypted form. Furthermore, for protecting the data against leakage by CF or stealing by TP_{id} from cloud2, while performing the analysis with high accuracy, the data is shared in noised form and MLPAM achieved a significant classification accuracy up to 92%. However, if any RU_{id} leaks the data intentionally to TP_{id} or somehow, TP_{id} becomes successful in stealing the data from $DO_{id}; id \in [1, n]/RU_{id}$, then the data is protected through leaker identification via performing probabilistic analysis. The experimental results signified that MLPAM is capable of recognizing a G_U effectively by securing up to 1 P_b, 0.791441 ϖ^*, 0.390156 $\min \varpi^*$, 97% DA, 100% DP, 100% DR, and 100% DS that validates its robust security.

X. Conclusion and Future Work

This article proposed a novel model named MLPAM for effective data protection in a real Cloud environment. To provide the stronger security, all the involved entities are considered to be untrusted and a robust mechanism is provided in the model by exploring every possible threat that may arise during data flow among the involved parties. MLPAM presented an effective sharing protocol to mitigate the loss due to data leakage. An influential distribution mechanism is proposed for data allocation and to detect a guilty entity with high confidence. The evident experimental results depicted that the guilty entity can be distinguished easily in the proposed scheme, which proves its effectiveness. MLPAM attained a significant improvement up to 186% over the existing works and simultaneously secured significant Detection Accuracy, Precision, Recall, and Specificity compared to the prior works that support its high performance. The comprehensive analysis and performance of the model over the well-known datasets and comparison with the existing works demonstrated that MLPAM is more secure, efficient, and optimal. MLPAM lays a foundation for future secure and efficient data sharing and management in multiple environments like Internet of Things, Big Data, etc. Furthermore, the request users might become capable of acquiring the data objects that are not allocated among these users through the use of shared keys. The emerged issue is referred to as future work and can be resolved by employing the set of distinct keys for the data objects.

References

[1] E. Zaghloul, K. Zhou, and J. Ren, "P-MOD: Secure privilege-based multilevel organizational data-sharing in cloud computing," *IEEE Trans. Big Data*, vol. 6, no. 4, pp. 804–815, Dec. 2020.

[2] S. Xu, G. Yang, Y. Mu, and R. H. Deng, "Secure fine-grained access control and data sharing for dynamic groups in the cloud," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 8, pp. 2101–2113, Aug. 2018.

[3] W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 2, pp. 331–346, Feb. 2019.

[4] M. Ali *et al.*, "SeDaSC: Secure data sharing in clouds," *IEEE Syst. J.*, vol. 11, no. 2, pp. 395–404, Jun. 2017.

[5] Z. Zhu and R. Jiang, "A secure anti-collusion data sharing scheme for dynamic groups in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 40–50, Jan. 2016.

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

[7] J. Wei, W. Liu, and X. Hu, "Secure data sharing in cloud computing using revocable-storage identity-based encryption," *IEEE Trans. Cloud Comput.*, vol. 6, no. 4, pp. 1136–1148, Oct.–Dec. 2018.

[8] Z. Fu, L. Xia, X. Sun, A. X. Liu, and G. Xie, "Semantic-aware searching over encrypted data for cloud computing," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 9, pp. 2359–2371, Sep. 2018.

[9] I. Gupta, N. Singh, and A. K. Singh, "Layer-based privacy and security architecture for cloud data sharing," *J. Commun. Softw. Syst.*, vol. 15, no. 2, pp. 173–185, 2019.

[10] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie, "An efficient file hierarchy attribute-based encryption scheme in cloud computing," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 6, pp. 1265–1277, Jun. 2016.

[11] H. Liu, X. Li, M. Xu, R. Mo, and J. Ma, "A fair data access control towards rational users in cloud storage," *Inf. Sci.*, vol. 418/419, pp. 258–271, 2017.

[12] Z. Liu, Z. L. Jiang, X. Wang, and S. Yiu, "Practical attribute-based encryption: Outsourcing decryption, attribute revocation and policy updating," *J. Netw. Comput. Appl.*, vol. 108, pp. 112–123, 2018.

[13] L. Zhang, Y. Cui, and Y. Mu, "Improving security and privacy attribute based data sharing in cloud computing," *IEEE Syst. J.*, vol. 14, no. 1, pp. 387–397, Mar. 2020.

[14] J. Li *et al.*, "An efficient attribute-based encryption scheme with policy update and file update in cloud computing," *IEEE Trans. Ind. Inform.*, vol. 15, no. 12, pp. 6500–6509, Dec. 2019.

[15] R. Yonetani, V. Naresh Boddeti, K. M. Kitani, and Y. Sato, "Privacy-preserving visual learning using doubly permuted homomorphic encryption," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2040–2050.

[16] T. Li, Z. Huang, P. Li, Z. Liu, and C. Jia, "Outsourced privacy-preserving classification service over encrypted data," *J. Netw. Comput. Appl.*, vol. 106, pp. 100–110, 2018.

[17] T. Li, J. Li, Z. Liu, P. Li, and C. Jia, "Differentially private naive Bayes learning over multiple data sources," *Inf. Sci.*, vol. 444, pp. 89–104, 2018.

[18] X. Ma, J. Ma, H. Li, Q. Jiang, and S. Gao, "PDLM: Privacy-preserving deep learning model on cloud with multiple keys," *IEEE Trans. Serv. Comput.*, vol. 14, no. 4, pp. 1251–1263, Jul.-Aug. 2021.

[19] P. Li, T. Li, H. Ye, J. Li, X. Chen, and Y. Xiang, "Privacy-preserving machine learning with multiple data providers," *Future Gener. Comput. Syst.*, vol. 87, pp. 341–350, 2018.

[20] P. Li, J. Li, Z. Huang, C.-Z. Gao, W.-B. Chen, and K. Chen, "Privacy-preserving outsourced classification in cloud computing," *Cluster Comput.*, vol. 21, no. 1, pp. 277–286, 2018.

[21] C.-z. Gao, Q. Cheng, P. He, W. Susilo, and J. Li, "Privacy-preserving naive Bayes classifiers secure against the substitution-then-comparison attack," *Inf. Sci.*, vol. 444, pp. 72–88, 2018.

[22] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service," *Proc. Privacy Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, 2018.

[23] P. Papadimitriou and H. Garcia-Molina, "Data leakage detection," *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 1, pp. 51–63, Jan. 2011.

[24] I. Gupta and A. K. Singh, "Dynamic threshold based information leaker identification scheme," *Inf. Process. Lett.*, vol. 147, pp. 69–73, 2019.

[25] Y. Fan, Y. Rongwei, W. Lina, and M. Xiaoyan, "A distribution model for data leakage prevention," in *Proc. Int. Conf. Mechatronic Sci., Elect. Eng. Comput.*, 2013, pp. 2617–2620.

[26] I. Gupta and A. K. Singh, "An integrated approach for data leaker detection in cloud environment," *J. Inf. Sci. Eng.*, vol. 36, no. 5, pp. 993–1005, 2020.

[27] M. Backes, N. Grimm, and A. Kate, "Data lineage in malicious environments," *IEEE Trans. Depend. Secure Comput.*, vol. 13, no. 2, pp. 178–191, Mar./Apr. 2016.

[28] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, 2007, pp. 321–334.

[29] C. Dwork and A. Smith, "Differential privacy for statistics: What we know and what we want to learn," *J. Privacy Confidentiality*, vol. 1, no. 2, pp. 135–154, 2010.

[30] C. Dwork, "Differential privacy," in *Encyclopedia of Cryptography and Security*. New York, NY, USA: Springer, 2011, pp. 338–340.

[31] H. Inoue and H. Narihisa, "Self-organizing neural grove and its applications," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2005, vol. 2, pp. 1205–1210.

Ishu Gupta (Member, IEEE) received the BCA and MCA (Gold Medalist) degrees in computer science from Kurukshetra University, Kurukshetra, India, in 2012 and 2015, respectively. She is currently working toward the Ph.D. degree in computer science with the Department of Computer Applications, National Institute of Technology, Kurukshetra, Kurukshetra, India.

She is awarded the Senior Research Fellowship by the University Grants Commission, Government of India. Her research interests include the areas of cloud computing, machine learning, and information security and privacy.

Miss Gupta has more than 25 publications in peer-reviewed journals, conferences, and was the recipient of the excellent paper award twice.

Rishabh Gupta received the MCA degree in computer science from the Guru Jambheshwar University of Science and Technology, Hisar, India, in 2015. He is currently working toward the Ph.D. degree in computer science from National Institute of Technology, Kurukshetra, Kurukshetra, India.

His research interests include cloud computing, deep learning, big data, and parallel processing.

Ashutosh Kumar Singh (Senior Member, IEEE) received the Ph.D. degree in electronics engineering from the Indian Institute of Technology BHU, Varanasi, India, in 2000.

He is currently a Professor and Head with the Department of Computer Applications, National Institute of Technology, Kurukshetra, Kurukshetra, India. He has more than 20 years of research and teaching experience in various universities in India, U.K., and Malaysia. He is a Postdoctoral Researcher from the Department of Computer Science, University of Bristol, Bristol, U.K. He has authored/coauthored more than 250 research papers and 8 books. His research interests include verification, design and testing of digital circuits, data science, cloud computing, machine learning, security, etc.

Rajkumar Buyya (Fellow, IEEE) received the Ph.D. degree in computer science and software engineering from Monash University, Melbourne, Australia, in 2002.

He is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems Laboratory, University of Melbourne, Melbourne, VIC, Australia. He is also the Founding CEO of Manjrasoft, a spin-off company of the University, commercializing its innovations in cloud computing. He has authored more than 650 publications and seven textbooks including *Mastering Cloud Computing* published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese, and international markets, respectively. He is one of the highly cited authors in computer science and software engineering worldwide (H-index=138, G-index=307, over 100 000 citations).