SN Computer Science (2024) 5:355
https://doi.org/10.1007/s42979-024-02657-z

ORIGINAL RESEARCH

Data Confidentiality in Machine Learning: Exploring Multivariate Regression and Its Application on Encrypted Medical Data

Eric Affum[1] · **Marian Enchill**[2]

Received: 22 July 2022 / Accepted: 26 January 2024

Abstract
In this work, we present a concise literature review on the application of multivariate regression using gradient descent. We employed gradient descent, which provides an optimal approach for minimizing the cost function in a regression model for error estimation, and further modelled a multivariate regression algorithm to perform regression analysis over encrypted data. To encrypt the dataset, we modified the original integer homomorphic encryption scheme into a new scheme for dataset encryption to achieve an efficient and secure data encryption and decryption operation. Since homomorphic operations do not support division, we devised a division-free gradient descent multivariate regression over cost-effective VHE encrypted training samples with high regression accuracy. We conducted simulations to compare least-squares and gradient descent with and without division. We also proved the applicability of machine learning for modeling encrypted datasets based on breast cancer datasets, with the focus on determining breast cancer potential patients. In our system, encrypted datasets should not interrupt the learning task, nor should the learning task reveal sensitive information to unauthorized users.

Keywords Multivariate regression · Least squares · Gradient descent · Homomorphic encryption scheme

Introduction

Safeguarding the confidentiality of sensitive medical data, including personally Identifiable Medical Information, Health Conditions, Treatment History, HIV/AIDS Status, Genetic and Biometric Data, emerges as a paramount concern within the realm of machine learning research. This emphasis arises from the ongoing imperative to preserve data privacy in various real-world scenarios. Particularly noteworthy is the healthcare sector's heightened emphasis on utilizing confidential patient information for predictive insights. Similarly, within the field of biology, gene predictions underscore the necessity for precise classification grounded in functional attributes. The current landscape reflects a significant dedication to advancing machine learning tasks, especially in cases where these services are operated through commercial providers' infrastructure. Renowned examples include the Google Prediction API, Microsoft Azure Machine Learning, RxNLP's API, Graph Lab, and Ersatz Labs. These efforts must harmonize with well-established regulatory structures like the Health Insurance Portability and Accountability Act (HIPAA) of August 21, 1996, and the General Data Protection Regulation (GDPR) of May 25, 2018. These regulations provide explicit guidance for systematically categorizing sensitive data, handling sensitive data, and ensuring privacy and security are resolutely maintained. Their intrinsic provisions reflect an unwavering dedication to preserving data confidentiality, fostering a climate of responsible data management among healthcare practitioners and researchers.

On cloud servers, machine learning allows users to train and deploy models. Users may use these models to generate predictions once they have been launched, and they do not have to bother about the service or the maintenance of the models. Again, it allows remote accessibility, easy expansion, is environmentally friendly, less expensive, and so on. On the other hand, the client must pay for every prediction made by the model. In a broader logic, it permits a model of machine learning as a service, where there is a separation between the data owner, the content provider, and the cloud

✉ Eric Affum
Affrico23@yahoo.com

Marian Enchill
enchil.M@yahoo.com

[1] University of Mines and Technology, Tarkwa, Ghana

[2] University of Education, Winneba, Ghana

Published online: 28 March 2024

service provider. In the health sector, every client may want to use a model that makes a prediction about his health, but patients are often hesitant to disclose their sensitive medical data when using a predictive health model due to concerns about privacy, stigma, potential consequences for employment and insurance, personal preferences, mistrust in data security, and legal and ethical considerations.

It is ideal to expect that the physician and the patient both execute a protocol in which the patient gets his health status prediction and neither the health worker nor the patient learns anything else about the other's information. In spite of the enormous advantages offered by cloud data services, such as cost-effectiveness, scalability, accessibility, and flexibility, organizations must remain vigilant about data security and privacy to ensure a secure and reliable cloud computing environment. The apprehension of data confidentiality and privacy cannot be overlooked when outsourcing computational tasks. The fact that the data owner and the cloud server are no longer in the same trusted domain may put the unencrypted data at risk [1]. Cloud computing relies on fundamental security processes, including authentication, data encryption, security patching, monitoring, policies, and more. These mechanisms ensure authorized access, data protection, vulnerability management, and regulatory compliance. Despite these measures, challenges exist, such as shared responsibility models, insider threats, data exposure, application vulnerabilities, and legal and compliance risks. In the cloud, users must protect their sensitive data before entrusting it to cloud providers. The cloud may leak data to unauthorized entities or may even be hacked [2]. One way to combat unauthorized access is to encrypt all the sensitive data before outsourcing it to a cloud server. Data encryption ensures confidentiality, control, and compliance with data protection regulations, reinforcing the need for proactive encryption practices. Encrypting data before outsourcing it to the cloud adds additional layer of security, ensures data confidentiality, and protects it from unauthorized, and even if the cloud provider is curious (as per the "cloud is honest but curious" concept), they cannot access the data without the user's decryption key. With this, only the appropriate data owner who has the private key can decrypt and access the data. However, outsourced encrypted data may have limitations in its utilization. A fully developed encryption scheme that allows arbitrary operations on ciphertexts even without knowing the secret key is called a "fully homomorphic encryption" (FHE) scheme [1, 3–8]. Rivest, Adleman, and Dertouzos [3] presented this concept, initially dubbed a "privacy homomorphism," immediately after Rivest, Adleman, and Shamir invented RSA. The scheme's mathematical features are such that multiplying the public key encrypted forms of two numbers, $M1$ and $M2$, indicated as $Epk(M1)$ and $Epk(M2)$, results in the encrypted form of the product of the two integers, which is $Epk(M1, M2)$. As shown in [9], decrypting this product will provide the true solution to the true product of two numbers. To enable the wide adoption of this paradigm, it is desirable to achieve secure computation on sensitive client data records without a breach of confidentiality. Assuming that a supervised learning algorithm consists of training phases and prediction phases, the protocol first learns a model m from a data set of labeled samples during the training phase, and then runs a regression K over a previously undiscovered feature vector v, by using model m to generate a prediction Y during the prediction phase (v, m). The fact that output or feature vector v and the model m are kept secret from some of the involved parties is critical.

Previous work on privacy-preserving machine learning may be categorized into two privacy-preserving approaches: training and classification approaches . These papers [10–12] demonstrated how to conduct machine learning statistical analysis on homomorphic schemes' encrypted data. The authors of [10] presented the training of many machine learning models over homomorphic encrypted data. They concentrate on a few straightforward classifiers, such as the linear means classifier. More complicated algorithms, such as support vector machines, are not discussed in these publications. They also provide private classification, but with limited security models that allow the customer to know more about the model but not just the final results. Moreover, using FHE alone to make the final comparison is costly, a challenge that may be solved using an interactive setup. A learning model was designed in [12], in which a third party can execute a medical prediction algorithm over a patient's encrypted dataset using fully homomorphic encryption. Anyone in their system, along with the patient, is aware of the prediction model, and their algorithm merely shields the patient's input data from the cloud. Such systems reveal more information to patients than simply the predicted result. The researchers employed a mix of varied-party and fully homomorphic encryption techniques to create a two-party computing framework. This enables learning techniques such as hyperplane decisions, Naive Bayes, and binary decision trees. The basic procedures remain the same, but extensive dialog between two "honest yet curious" parties is necessary here. Researchers [13] studied the topic of statistical analysis on encrypted data. They concentrated on two objectives in particular: determining the mean and variance of univariate and multivariate data and conducting linear regression on a multi-dimensional encrypted corpus. It proves that fully homomorphic encryption may be used for large-scale statistical research. Batch processing and encryption on a per-element and per-block basis are rather costly in this highly decentralized scenario [14]. However, it must extend the ciphertext from an integer to a vector, making attempts to make this approach more efficient challenging

[15]. As a result, improving computing efficiency in homomorphic evaluation while reducing ciphertext expansion has remained a challenging and unresolved subject.

Related Works

Homomorphic Encryption's Evolution

Homomorphic encryption (HE) generates ciphertext that is purposefully malleable, enabling operational processes on encrypted data. The fundamental RSA encryption technique [3] maintains multiplicative homomorphism. In 1999, Paillier et al. [16] introduced an encryption system with an additive homomorphic characteristic. There has been some progress in this research sector since Gentry's breakthrough [17] in 2009. Regev's encryption system [18] was released the same year as proof-of-concept evidence for a HE method employing learning with errors (LWE) and random linear codes. Lyubashevsky et al. further developed learning with error problems over rings in 2010 [19]. Since then, a variety of HE techniques with shorter bootstrapping times have been presented [20]. In 2016, Chillotti et al. [21] modified the HE technique and reduced the bootstrapping time to less than 0:1 s. Zhou and Wornell [22] proposed VHE, which encrypts integer vectors to enable bounded degree computation of arbitrary polynomials in the encrypted domain. The basic VHE is insecure enough to allow an attacker to obtain the secret key or plain vector from the key-switching matrix or ciphertext, correspondingly [23]. Moreover, the components in each vector and matrix are encoded as binary strings in the underlying VHE. This work seeks to address some of the efficiency issues raised in [22].

Machine Learning Over Encrypted Data

Hall et al. [24] proved that a fully secure linear regression approach based on the HE is feasible for use on a tolerably massive dataset shared by multiple parties. They used the Paillier HE scheme [16] for their regression analysis. In our research, we deemed this a non-federated data model, in which data is stored in the cloud. Later that year, Naehrig et al. [10] demonstrated the feasibility of applying machine learning algorithms over encrypted data, using a leveled homomorphic encryption system for a non-federated data model. They also showed the HE system, which is based on the level of arithmetic computations.

Furthermore, a variety of techniques for deriving classification models utilizing Yao's garbled circuits [25] had been taken into account. Bost et al. [26] postulated various techniques for successfully implementing machine learning algorithms on encrypted data by employing HE schemes in 2015. They introduced two models for developing the classifier, one based on data from the HE system and the other on Yao's protocol. In recent times, Gilad et al. [27] demonstrated that neural networks can be applied to encrypted data. Given the complexity and limitations of the highly regarded HE scheme, they limited their work to low-degree polynomials.

In 2018, Hesamifard et al. introduced a system for machine learning as a service, in which they leveraged client–server interaction to compute the model. Moreover, Sadat et al. [28] and Jiang et al. [29] developed their frameworks for linear and logistic regression utilizing the same HE approaches. Nevertheless, in their system, they additionally utilized the secure hardware SGX [30]. On the other side, in our work, we developed the framework to be non-interactive. Table 1 shows a quick description of relevant research in the field of machine learning over homomorphic encrypted data.

Our Contribution

In this work, we present a cost-effective vector homomorphic encryption (CE-VHE) scheme based on [22] to achieve efficient and secure data encryption and decryption operations. We also presented an efficient and secure multivariate regression approach over encrypted medical datasets. We

Table 1 Summary of related works of machine learning algorithms on encrypted data

Scheme	Year	ML algorithm	HE scheme	Data model	Garbled circuit	Parallel
Hall et al. [24]	2011	Regression analysis	Pillar	Federated	No	No
Naehrig et al. [10]	2012	Linear means classifier, Fisher's linear discriminant classifier	Leveled	Federated	No	No
Bost et al. [26]	2015	Linear means classifier, Fisher's linear discriminant classifier	Fully	Federated	Yes	No
Sadat et al. [28]	2018	Linear regression, logistic regression	Somewhat	Federated (SGX)	No	No
Jiang et al. [29]	2018	Logistic regression	Somewhat	Federated (SGX)	No	Yes
Toufique et al. [31]	2018	Linear regression	Somewhat	Centralized	No	Yes
Our	2023	Multivariate regression	VHE	Centralized	No	Yes

demonstrated how multivariate regression analysis can be used to predict the value of one or more responses from a set of attributes. Gradient descent was used to find the cost function or the solution. Gradient descent, which gives an optimized approach for minimizing the cost function, is used in machine learning for error detection minimization. Because homomorphic encryption operations do not support division, we designed a division free gradient descent for predicting regression solutions. Specifically, the main contributions are as follows:

- We modified the VHE [22] scheme by developing a new key transformation process using an invertible matrix in key generation and incorporating an error vector in the ciphertext, and then proposed a new VHE called CE-VHE, which has been proven to accomplish semantic security. CE-VHE distinguishes itself by having a small encryption key size and a minimal execution time for data encryption while also supporting homomorphic operations and efficient linear transformation.
- A secure and efficient multivariate regression solution over encrypted data is accomplished by incorporating CE-VHE and a division-free gradient descent multivariate regression solution. Given that homomorphic encryption lacks support for gradient descent because of its division operations, an ingenious solution has been devised. A division-free gradient descent algorithm has been formulated and applied in our work. The accuracy of our division-free gradient descent is determined using the least squares model result (an exact or accurate solution) as a benchmark. The division-free gradient decent approach made it possible to execute a machine learning algorithm over a homomorphic encryption scheme that does not accept division. Our division-free gradient descent algorithm also helped to achieve efficient and accurate regression over homomorphic encrypted data with minimal execution time and communication cost.
- We implemented our systems and carried out a performance assessment. The results obtained demonstrate that the new CE-VHE is secured against both classical and quantum security attacks, has a smaller public key size, and has higher data encryption efficiency than the original scheme.

Organization

The rest of our work is as follows: In "Related works", related works are discussed. In "Preliminaries of the study", the preliminaries of the work are presented. System modeling and problem formulation are described in "System Model and problem formulation". The proposed encryption scheme is described in "Cost effective integer vector homomorphic encryption". In "Multivariate regression over encrypted data", we presented our proposed machine learning task over encrypted data. The experiment and the results are presented in "Experiment and results". In "Evaluation and performance analysis", we present the security and performance analysis of the proposed scheme. Finally, in "Conclusion", we give the conclusion to the study and outline future research.

Preliminaries of the Study

In this section, we present linear regression, multivariate linear regression, least squares, gradient descent, and a homomorphic encryption scheme.

Regression Analysis

Linear regression: Denote a set of x as input variables and a set of output variables as y. The regression model input of one or more independent variable x(weight, smoking habit, and body mass index of a person), will give the output, a dependent variable of y(blood pressure).

Multivariate Linear Regression [32]: The multivariate model assumes the multiple linear regression model to hold for each component of the response vector, that means, one assumes for the jth dependent variable. The fundamental linear model is:

$$y_{ij} = x_i^T \beta_{(i)} + \varepsilon_{ij} \tag{1}$$

$$= \beta_{0j} + x_{i1}\beta_{ij} + \cdots + x_{ip}\beta_{pj} + \varepsilon_{ij,i=1\ldots n}$$

where ε_{ij} denotes noise variable with $E(\varepsilon)$ and β_0 as vector parameters. It is essential that the vector parameters depend on j, which refers to the jth component of the response vector.

Computing coefficient: There are so many models used to compute coefficient or analyzing regression. To compute vector β_0, we will consider the least square and gradient descent approach.

Least Squares: As stated in [33], Least squares is the algorithm that solves the Empirical risk minimization (ERM) problem for the hypothesis class of linear regression predictors with respect to the squared loss. Given a training set S, and using the homogenous version of LS, the general model is given by:

$$Q(\beta) = (y - X\beta)^T (y - X\beta) \sum_{i=1}^{n} \left(y_i - X_i\beta\right)^T \left(y_i - X_i\beta\right) \tag{2}$$

which for full rank matrix X yields the least squares estimator:

$$\beta = \left(X^T X\right)^{-1} X^T y \tag{3}$$

Gradient descent: The simplest approach of using gradient information is to choose weight to comprise a step-in direction of negative gradient, so that

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w^{\tau}), \quad \eta > 0 \tag{4}$$

Gradient descent [34] is iterative with the initial estimate w_1 of unknown parameter vector. From the initial estimate, we note that the direction of steepest descent from this point to follow the negative -∇E of the objective function evaluated at w_1. Let the hypothesis function be given as

$$h\beta(x) = \beta^T x = \sum_{i=0}^{n} \beta_i x_i \tag{5}$$

And the batch gradient descent update rules denoted as:

$$\beta_j := \beta_j - \alpha \frac{1}{m} \sum_{i=1}^{n} \left(h\beta\left(x^{(i)}\right) - y^{(i)}\right) (\text{for all } j) \tag{6}$$

where α which is the learning rate is gradient descent between the ranges of $0 \leq a \leq 1$

Recall that the cost function is defined as:

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h\beta\left(x^{(i)}\right) - y^{(i)}\right)^2 \tag{7}$$

The gradient descent could be obtained as:

$$\nabla E = \frac{1}{m} (X^T((x * \beta) - y) \tag{8}$$

And therefore, the coefficient β is given as:

$$\beta = \beta - \alpha * \nabla \mathrm{E} \tag{9}$$

Homomorphic Encryption Scheme

Let P and S be a pair of keys, where public key P and secret key S are keys for encryption and decryption, respectively. If a cryptosystem characteristic can map the computation over ciphertext to the corresponding plaintext without knowing the decryption key, it is known as homomorphic. An integer vector homomorphic scheme is employed in the suggested schema, which permits computations, including secure addition, linear transformation, and weighted product. Given a public key as P, the message m and error term$e$, the encryption is obtained as $c = P \cdot m + e$ and for the secret key as$S$, the decryption is obtained as: $m = S \cdot \frac{c}{w}$, where w is a large integer. The decryption process will be successful if the size of the error is $|e| < \frac{w}{2}$.

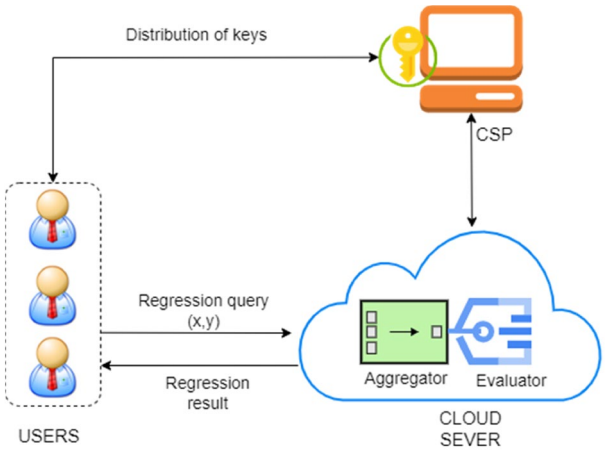


Fig. 1 System model

System Model and Problem Formulation

System Model and Entity

This system consists of collecting health information and performing machine learning tasks to generate information that will assist patients and healthcare professionals in diagnosing patients and also predict their risk of being infected by a particular health problem. In Fig. 1, we identify three entities, which consist of e cloud server, and the cryptography service provider. They are described below.

- The Data Owner is the client interested in the Cloud Service and is in charge of the information being processed.
- Cryptography Service Providers initialize the system and provide setup parameters and keys to the various users and the server.
- The Cloud Server consists of an aggregator and an evaluator. It might be controlled by a third party, an accomplice organization, or even the organization itself, off-premises or in some stand-alone setting. Perform various computations on the data sent by the data owner and also provide the model.

Our framework is intended for some clients to send information to a focal server. The server analyses the contributed data and generates a model, which can then be used for expectation or proposal errands. All the more particularly, every client has a private record containing two variables $x_1 \in \mathbb{R}^d$ and $y_1 \in \mathbb{R}$. The server wishes to perform computation for the model, such that $\beta \in \mathbb{R}^d$. The purpose of this work is to ensure that the server should not learn anything about x and y which is $y_1 \approx \beta^T x_i$ revealed by β.

Threat Model

Personal health records of data owners, on the other hand, are subject to being tampered with or corrupted since their components, such as computers, networks, applications, and devices, may introduce major security flaws. We are most certainly not worried about a noxious saver who is attempting to corrupt the computation in the hope of creating an improper result. The data sent between the patient device and the servers could be eavesdropped or seized by hackers, and the cloud is suspected of purposely or mistakenly exposing the preserved data. At times, servers try to learn information about private data sent by clients. However, if a PHR enters the wrong hands, it might be mishandled, faked, and publicly published, jeopardizing the data owner's identity. The saver, even in its malicious condition, should not be able to learn anything about the client-contributed data or the learning result output by the algorithm.

Dataset Processing

There are many different cancer datasets available, each with their own unique characteristics and features. However, in general, cancer datasets are collections of information about patients with cancer, including their demographic information, medical history, and tumor characteristics.

The homomorphic encryption scheme has some limitations, including the inability to perform floating-point calculations, the lack of divisional homomorphism, and the requirement for larger ciphertext modulus sizes for larger inputs. Taking these restrictions into account, we preprocessed our data as follows:

1. Higher values Attributes: For characteristics with higher values, we evaluated the attribute's variance and examined its effect on the regression model's correctness. We regarded a characteristic as an outlier and eliminated it if it had a detrimental effect on accuracy. We condensed the amount to two digits if the effect was favorable. The two-digit decrease was necessary to fit within the ciphertext modulus and prevent overflow.
2. Relatively small values: For characteristics with smaller values (less than one), we additionally evaluated the attribute's variance and accurately verified its relationship. If an attribute has a beneficial effect, we scale it up so that its minimum value is an integer; if not, we eliminate it.
3. Round to the nearest integer: Rounding numbers to the nearest integer is the final step-in data preparation. The homomorphic encryption scheme does not handle floating-point values, which is the cause.

System's Goal

Our systems should be able to conduct multivariate linear machine learning tasks on encrypted data while achieving these goals.

1. Regression Accuracy: The multivariate regression precision on ciphertext training samples should be reasonably high to facilitate correct medical decision-making for the health system. The accuracy of learning over encrypted data samples should be nearly the same as that of the original data.
2. High Efficiency: The regression on ciphertexts must be performed effectively on large datasets on the server, and the communication and computation time introduced by multivariate regression ought to be minimal.
3. Security guarantee: to ensure that the server learns nothing about the data contributed by the data owners, as well as the learning algorithm's output result to the client. However, the above goal should be accomplished with the least computational overhead. In this concept, we accept that it is in the saver's ideal interest to create the right model.

Our System Scenario

We conducted multivariate regression on the unencrypted and encrypted datasets. Multivariate regression helps provide a more accurate and comprehensive analysis of data, leading to better predictions and decision-making. Since the fundamental healthcare data is confidential, the data provider must encrypt it before sending it to the cloud for storage. The cloud server will conduct the linear regression method to produce a classification model after storing the encrypted data. The produced model will be encrypted, and the cloud server will have no knowledge of it. As a result, it meets our criterion that semi-honest opponents have no knowledge of not just the data but also the created model. Following model generation, the cloud server will transmit the encrypted model to the data provider, who will retrieve the model. The scenarios that we considered in this work are shown in Fig. 2a and b. As depicted in Fig. 2a, users compute most of the data locally, including all the division parts, before sending them to the server. They aggregate all their respective input variables. Using gradient descent without division, the server extracts the model linear best fit to the data submitted by the client. An important aspect of this work is the scenario demonstrated in Fig. 2b. After the users have locally computed all their input various variables, each user encrypts their data and sends the encrypted data to a third party (a server which has an aggregator and an evaluator), which runs the learning algorithm and extracts β. This describes the model's linear best fit to the encrypted data

submitted by the client. At a certain point in time, the server may interact with a trusted service provider who can be trusted not to conspire with the server to cause any malicious act. The users compute the $R_i = X^T X, a = X^T y, i - 1, \cdots, n$. The CSP provides the users with their key pairs and encrypts $R'_i = Enc(R_i), a'_i = Enc(a_i)$ and sends them to the server. The server aggregates the encrypted data inputted and contributed by the users and the server uses it to learn the model β without learning anything about the model.

The Approach Used

The input matrix x_i and the vector y_i, where $i = 1, \cdots, n$ are variables from each client, the server aggregates these variables and transfers them to the evaluator. The evaluator regresses to fine the linear relationship between the input and the output by the client, as obtained in multivariate linear regression. To obtain the model in the gradient descent approach, we will compute a matrix $C = X^T X$ and a vector $m = X^T y$. Here we aim at solving linear systems of equations using the gradient descent method. The straight-forward method, the least square method for finding the cost function, is $E(w) = ||Cw - d||^2/2$, which provides the exact solution. However, since homomorphic encryption schemes' operations do not support division, we use the least square method as a benchmark, and then we use the approximate solution of gradient descent without division to demonstrate how regression can be conducted on ciphertext. Let's consider gradient descent $\nabla_w E(w) = Cw - d$, with a fixed leaning rate k an identity matrix I, and $w_0 = 0$. $R = I - K \sum_{i=c}^{n} C_i$ and $a = k \sum_{i=1}^{n} m$ are calculate and the results are used to execute the recursion equation $w_{j+1} = Rw - a$ to find the approximate solution for the model. To ensure data confidentiality, users encrypt their sensitive data with an efficient vector homomorphic encryption scheme before it is sent to the cloud for regression. The user computes most of the division parts, $R_i, a_i I, w_0$ and then encrypts them to obtain $R'_i, a'_i I', w'_0$. Finally, we compute a recursive algorithm based on gradient descent over encrypted data to obtain the model.

Cost-Effective Integer Vector Homomorphic Encryption

The encryption scheme considered for our implementation is a normal modification of the scheme [22], which is based on the PVW [35], from binary vectors to integer vectors. This homomorphic encryption is different from previous schemes. It allows computation on encrypted data while maintaining the secrecy of the function. The initial IVHE [22] is not robust enough to withstand present threats, which allow an adversary to retrieve the raw vector or private key from the encrypted text or key transformation matrix, accordingly [36]. Furthermore, the components within every matrix or vector are encoded as binary strings in the initial IVHE. Significant dimensional vector operations are conducted little by little, which necessitates the use of huge public keys to encrypt data, leading to high computation complexity. Because the initial IVHE uses a binary to represent vectors and matrices, computations on increased vectors incur substantial computational and transmission overhead. As a consequence, in this work, a new transformation matrix using an invertible matrix in key generation is constructed to prohibit private key discovery, and an error

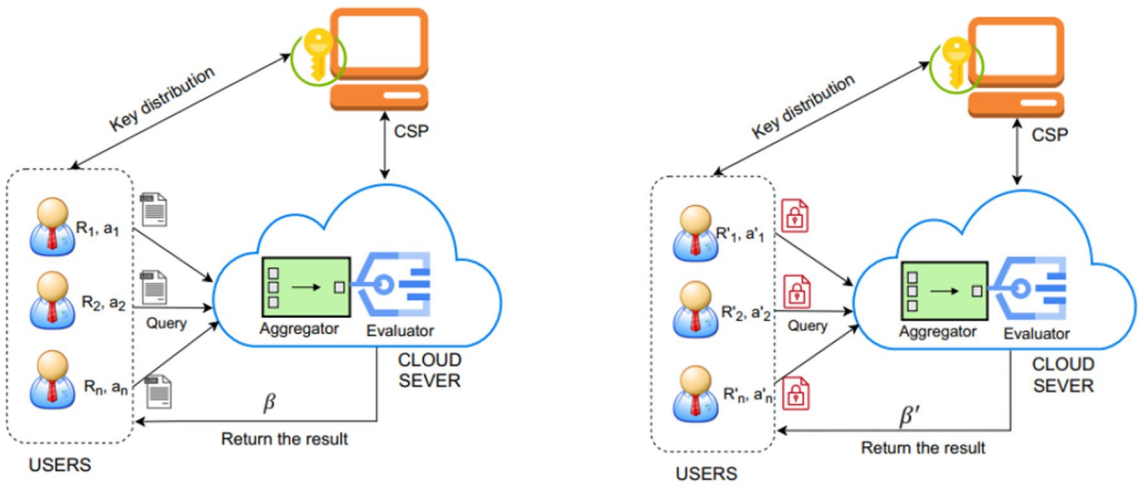


Fig. 2 Learning over encrypted and unencrypted data using gradient descent

vector is employed to obscure the raw vector in data encryption. As a result, the plaintext vector and decryption keys cannot be reconstructed from the encrypted message and key-switching matrix. This mechanism is secured against plaintext attack due to its indistinguishability. Our scheme consists of five polynomial algorithms, which are described below:

VH.Setup(λ): Given secret parameter as λ, the algorithm selects random parameters $m, n, p, q, w \in \mathbb{Z}$, where $m < n$ and $q \ll p$, selects discrete Gaussian noise distribution $\chi \in \mathbb{Z}_q$ with a standard deviation δ. It generates two identity matrices $I_1, I_2 \in \mathbb{Z}_p^{n\times n}$, where $I_1 I_2 = I$ and outputs the public parameters as $PP = (p, q, m, w, n, \chi)$

VH.KeyGen(Param): This algorithm first executes the Transformation(S_0) which takes input of the initial key S_0 and output(S, P). The transformation algorithm does the following: Generates these matrices:$M_s, M_m \leftarrow \mathbb{Z}_q^{n\times n}$,$D \leftarrow \chi^{m\times(n'-m)}, A \leftarrow \chi^{(n'-m)\times n}$,$S_d = [I, D] \in \mathbb{Z}^{m\times n'}$, where I is an identity matrix of m and $P_d = \begin{bmatrix} S_0 - DA \\ A \end{bmatrix} \in \mathbb{Z}_q^{n'\times n}$ and computes $S = S_d M_s$ and$P = M_m P_d$. Finally, it outputs the public key as $P \in \mathbb{Z}_q^{m\times n}$ and Secret keys as $S \in \mathbb{Z}_q^{n\times m}$, where $S, P = xI$.

VH.Enc(b, P): The algorithm takes an input of the public parameter$P$, the dataset, message$b$, and selects a random error term $e \leftarrow \chi^n$ and output the ciphertext text as $c = Pb + e$

VH.Decrypt(c,S): The algorithm takes an input of the encrypted message c and secret key S and extract the messages as $b = \left[\frac{Sc}{x}\right] q$

Correctness of CE-VHE Scheme

The correctness of is ensured by the equation below:

$$b = \left[\frac{Sc}{x}\right]_q = \left[\frac{S(Pb+e)}{x}\right] = b + \left[\frac{Sc}{x}\right]_q. \tag{10}$$

To recover the dataset, message b, the following conditions should be satisfied; the secret key S must be a accurate, and $|Se| < \left(\frac{x}{2}\right)$ must be satisfied. That is, we obtain $n|S|\,|e < (\frac{x}{2})$. Hence, we represent upper bound of $|e|$ as $E < \frac{x}{2n|S|}$. We obtain an upper bound of $|e|$ as compared to the original scheme of $E < \frac{x}{2}$. This encryption scheme can support three types of fundamental operations on integer vectors in encrypted domain. In a secure environment, the encrypted integer vectors of any polynomial on integers that does not exceed a specific degree may be calculated properly and efficiently.

Security Proof

The proposed efficient CE-VHE is secured under learning with error as stated in theorem 1 The sketch of our security proof model is shown below.

Theorem 1: The CE-VHE scheme is semantic security assuming that the LWE problem is intractable.

Proof: The CE-VHE mechanism realizes one-way security. The ciphertext of c is computed as $c - Pb + e$. Therefore, to solve one-way security issue of a given c is mandated to answer the plaintext b which is equivalent to breaking the LWE problem. Thus, to compute b when given $c \approx Pb$, where is P is random matrix. Hence one-way security can be reduced to LWE problem. Given $P = M_m P_d$, the value of M_m and P_t with the size depending on D and A is required to be large enough to ensure randomness. Also, each row of M_m should not depend on other rows.

Therefore, the security of our CE-VHE is IND-CPA secured. In the IND-CPA security notion, the attacker has a chance to adaptively request ciphertext from the oracle many times, where the encryption oracle responds to the attacker at any time. The oracle receives two plaintexts a_0 and a_1 and outputs a challenging ciphertext c which is either the ciphertext of a_0 or a_1. If the adversary is able to solve one-way security by outputting the message b from the ciphertext $c = Pb + e$, then the security notion IND-CPA cannot be attained. As luck would have it, our scheme is one-way security secured. Also, given the difficulty of separating a and b with regards to c, it is similar to differentiate between $a_0 - a_0$ and $a_0 - a_1$ depending on the homomorphic characteristic. To be explicit, it is similar to differentiate between the nonzero vector x and zero vector 0. Given ciphetext $c_0, c_1 = \left(P_0 + e_0, Pb + e_1\right)$ where $b \neq 0$ and $e_0, e_1 \leftarrow \chi^n$, the ciphertext c_0 and c_1 cannot be distinguished. Hence, the scheme CE-VHE gratifies IND-CPA. Hence CE-VHE is semantic secured.

Multivariate Regression Over Encrypted Data

In this chapter, the proposed multivariate regression analysis models and protocols that address the issue of machine learning, and its confidentiality are presented. The protocols for least square estimation, which provides the exact solution, and gradient descent estimation protocols for both with and without division, which provide an approximate solution, were conducted. The performance analysis of the proposed division-free gradient descent machine learning task over encrypted data is also discussed.

Data Description

We utilized the Wisconsin Diagnostic Breast Cancer (WDBC) database from the UCI library to train machine learning models [37]. The set of data comprises 569 instances made up of 10 attributes (ID, diagnosis, and 9 real-valued input features). Table 2 depicts the data representation of ten real-valued features.

Accurate Solution by Least Square Model

Let x_i be input variables and the output variable y_i, where $i = 1, \cdots, n$. These are variables from each client. The server aggregates these variables and regress to find the linear relationship between the input and the output by the client as obtain in multivariate linear regression. To obtain the model w in least square approach, the server first computes a matrix $C = X^T X$ and a vector $m = X^T y$ and the further compute C_i and m_i such that $c = \sum_{i=1}^{n} C_i$ and $m = \sum_{i=1}^{n} m_i$. Accurate solution w is computed by least squares model.

$$w = C^{-1} m \tag{11}$$

Approximate Solution by Gradient Descent Model with Division

Here we aim at solving linear systems of equation using gradient descent method. The straightforward method is the use of least square for finding the cost function is $E(w) = \| Cw - \frac{d\|^2}{2}$. However, it has poor extrapolation qualities and is impractical when working with big batches of datasets and sophisticated algorithms such as Support Vector Machine [38, 39]. As a result, we use gradient descent, which provides an optimized approach for minimizing the cost function and use least square as a benchmark. Then the gradient descent is $\nabla_w E(w) = Cw - d$. The system model in Fig. 2 gives the general overview of how this model works. Let's consider gradient descent with a fixed leaning rate k identity matrix I, and $w_0 = 0$. The users compute C and m as above and set $R = I - kC$ and$a = km$, and send them to the server. The recursion equation $w_{j+1} = Rw_j + a$ is used to find the approximate solution for the model.

Table 2 Breast cancer dataset representation

Number	Attributes	Domain
1	Sample code	ID number
2	Clump thickness'	1–10
3	Uniformity of cell size	1–10
….	….	….
….	….	….
10	Diagnosis	Benign, Malignant

Approximate Solution, Gradient Descent Model Without Division

To avoid division, which is restricted by homomorphic basic operation properties, gradient descent without division is computed in order to achieve homomorphic encryption scheme requirements. Users locally compute all the parameters, which involves division before encryption. The server should not compute any divisions. Each users locally does the following: Compute $C = X^T X$ and $m = X^T y$ and then set $C = \sum_{i=1}^{n} C_i$ $m = \sum_{i=1}^{n} m_i$. Compute $R = I - k \sum_{i=1}^{n} C_i$ and $a = \sum_{i=1}^{n} m_i$. Finally, the users compute $R_i = kC_i$ and $a_i = km_i$ and these are sent to the server for aggregation. The server finally computes the solution using the recursion equation $w_{j+1} = Rw_j + a$.

Multivariate Machine Learning Task Over Encrypted Data

To ensure data confidentiality while finding the approximate solution, the model in Sect. (5.2) will be considered. Each user locally computes $C = X^T X$ and $m = X^T y$. The following parameters are also computed locally by the use: $R = I - k \sum_{i=1}^{n} C_i$, $a = \sum_{i=1}^{n} m_i$, and $a_i = km_i$. The user encrypts R_i, a_i, I, w_0 using vector integer homomorphic encryption scheme as $R_i' = Enc(R_i)$,$a_i' = Enc(a_i)$, $I' = Enc(I)$ and $w_0' = Enc(w_0)$. Each user sends R_i', a_i', I_i' and w_i' to the server. The sever then aggregates $a' = \sum_{i=1}^{n} a_i'$ and $R' = I' - \sum_{i=1}^{n} R_i'$ and finally compute the approximate solution using the recursion equation $w_0' = R' w_0' + a'$.

Multivariate Machine Learning Protocols

Based on the scenarios in the model presented above, the protocols involved in finding the solution to the problem in this work are shown below.

First Protocol: The Exact Solution and Approximate Solution with Division

This protocol demonstrates how the exact solution, and the approximate solution can be computed. This protocol consists of preparation, setup and score calculation.

- *Preparation*

1. The server computes all the necessary specifications requested by the service provider. These include dimension of the input data and the number of bit representations for integer numbers.
2. Client also processes their dataset.

- *SetUp(x, y)*:
  Users locally compute matrix $C = X^T X$ and $m = X^T y$ vector.
- ***Upload***(C, m)
  Users send C and m to the server.
- ***ScoreCalculation (C, m):*** The server does the following:

1. Computes aggregation of C_i and m_i such that $m = \sum_{i=1}^{n} m_i$ and $C = \sum_{i=1}^{n} C_i$,
2. Solves multivariate linear regression as follows:

   Option 1: Accurate solution w by least squares

   $w = C^{-1} m$

   Option 2: Approximate solution w by gradient descent with division.
   - Computes $R = I - kC$ and $a = km$ where I is identity matrix, k is the learning rate.
   - Computes $R = I - k \sum_{i=1}^{n} C_i$ and $a = k \sum_{i=1}^{n} m_i$
   -The solution w is solved by computing recursion equation $w_{i+1} = Rw_j + a$. Where $w_0 = 0$
3. Return the learning result w

Second Protocol: Gradient Descent without Division

This protocol presents how the approximate solution can be solved using gradient descent without division. Steps consist of ***Preparation, Setup, Upload and Score Calculation.***

- ***Preparation:*** The client does the following:

1. Locally compute matrix $C = X^T X$ and vector $m = X^T y$
2. Compute $R = I - k \sum_{i=1}^{n} C_i$ and $a = k \sum_{i=1}^{n} m_i$.
3. Set $R_i = kC_i$ and $a_i = km_i R_i$

- ***Setup:*** The server provides all the required specification such as input data size and the number of bits to represent integer and fractional number to the service provider based on a request
- ***Upload***$(\boldsymbol{R_i, a_i, w_i})$:
  Clients send R_i and a_i to the server for aggregation.
  Where $R = I - \sum_{i=1}^{n} R_i = I - \sum_{i=1}^{n} kC_i = I - kC$ and $a = \sum_{i=1}^{n} a_i = \sum_{i=1}^{n} km_i = km$
- ***-ScoreCalcualtion:*** The server does the following:

1. The server firstly aggregate users uploaded datasets R_i and a_i
2. Computes w by solving the recursion of the equation, $w_{j+1} = Rw_j + a$ where $w_0 = 0$.
3. Returns the learning result as w

Third Protocol: Multivariate Task over Encrypted Data

This protocol shows how a multivariate machine learning task is computed on encrypted data. This protocol uses gradient descent without division. The protocol consists of ***Preparation, VHE.SetUp, VHE.Enc, Upload, ScoreCalculation and VHE.Dec***.

- ***Preparation***: The client does the following:

1. Compute matrix $C = X^T X$ and vector $m = X^T y$.
2. Compute $R = I - k \sum_{i=1}^{n} C_i$ and $a = k \sum_{i=1}^{n} m$
3. Set $R_i = kC_i$ and $a_i = km_i$

- ***EVH.SetUp***(λ): This algorithm is executed by the CSP. It takes input of security parameters λ and generate a secret S key and public M key for users based on CE-VHE scheme.
  All the required specifications such as data size and bits representation of integers are provided by the server.
- **EVH.Enc:** The clients locally encrypt R_i, a_i, I, w_0 by running the encryption algorithm of CE-VHE to output $R'_i = Enc(R_i), a'_i = Enc(a_i), I' = Enc(I)$ and $w_0 = Enc(w_0)$
- **Upload**(R', w', a'):
  The client sends these ciphertexts $R'_i, a'_i w'_i$ to the server.
- ***ScoreCalcualtion***(R', w', a')

1. The server first aggregates users input $R' = I' - \sum_{i=1}^{n} R'_i$ and $a' = \sum_{i=1}^{n} a'_i$
2. Compute w' by running the recursion equation $w'_{j+1} = R' w'_j + a'$.
3. Output the encrypted result w' to the user.

- ***EVHDec***(w', S'): This algorithm is execute by the user. The algorithm takes input of the ciphertext w' and secret key S' and output the learning result w

Experiments and Results

In this section, we conduct two sorts of experiments. First and foremost, we simulated the modified version of [22] and compared some matrices with the modified version. Second, an experiment was conducted to analyze and compare multivariate linear regression using the least squares model, gradient descent with division, and gradient descent without division and, more importantly, to demonstrate how multivariate learning tasks could be performed on encrypted data without jeopardizing sensitive data confidentiality. To

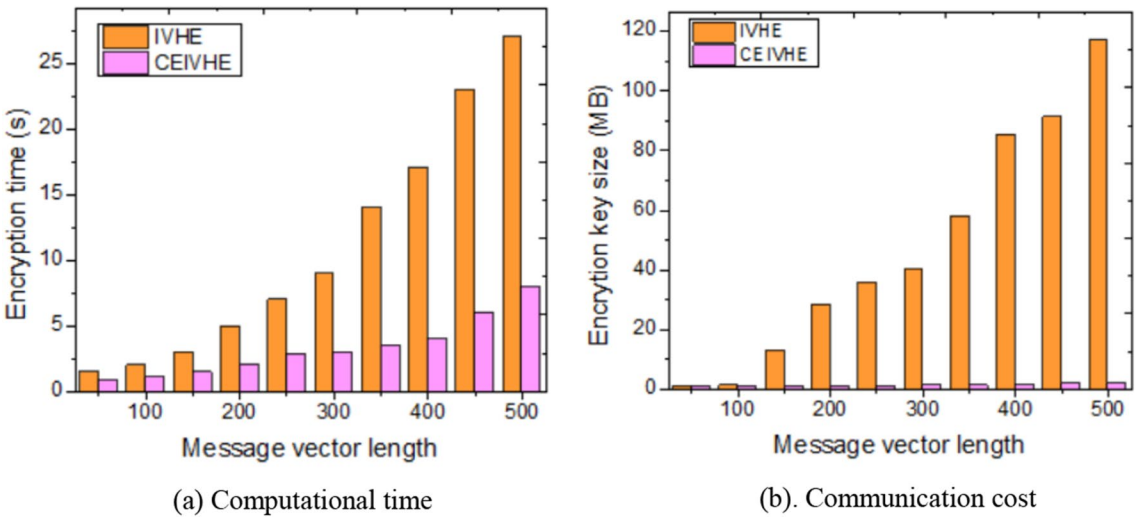
demonstrate the effectiveness of this research, the multivariate datasets that were obtained from [37] were used. The implementation of our CE-VHE scheme was conducted using Python programming language on an Intel i7-8700 processor at 2.53 GHz and 8 GB RAM running Windows 10 operating system, and the execution on the server was carried out on an E5-2430 with 24-GB running Ubuntu version 16.04. The learning task algorithm (gradient descent with and without division) and the last squares were first modeled and tested. The purpose of this study is to find out whether the practicality of a machine learning algorithm based on multivariate regression with division-free gradient descent over encrypted data is possible. This work demonstrates the effectiveness and the performance of multivariate regression over encrypted data. In this regard, the sizes of the features ranged from 2 to 10. The dimension of the dataset used also varied between an array of 10 and 100. For the purpose of comparison to determine computational cost, the analysis was conducted on both plaintext and ciphertext to demonstrate the extent of the computational cost of an operation for the ciphertext of the dataset. The performance results are summarized below.

Simulation Parameters

Two requirements should be satisfied while setting the parameters. These are the following: To begin with, it must ensure security; specifically, the parameters must ensure a desirable level of security against known homomorphic encryption scheme threats. Second, it must ensure that the data is accurate. When given a specified computation, the encryption algorithm must be able to accurately execute the computation without the error terms in the ciphertexts being too large, resulting in the result being incorrectly decrypted. The parameter selection for this vector homomorphic scheme is based on the size of the vector or matrix needed to be encrypted. The selection of the two main secrete parameters follows after the work in [40, 41]. Taking a vector length of 10, $w = w^{30}$ and $l = 100$ and a matrix of 100 by 10, w was set as, $w < 2^{60}$ and $l = 1000$.

Performance analysis of the CE-VHE scheme

As shown in Fig. 3a and b, the public key size of the CE-VHE mechanism increases vehemently as the size of the message vector also increases. However, our scheme recorded a very low increase in public key size. We also realized a higher level of encryption efficiency as compared to the CE-VHE scheme. This remarkable improvement has been achieved based on the efficiency of the key transformation process. The key transformation process of the CE-VHE is computed as $P_0 = \frac{xI - DA + E}{A}$, where P_0 is the same as actual public parameter, xI in P_0 has a size of $n'l \times m$ in binary and the ciphertext of b under the key P is output as $c = Pb$. However, with CE-VHE, instead of using the original key transformation process, we computed our key transformation process as $P_t = \frac{xI - TA}{A}$, where xI in P_t has a size of $n' \times m$ which is a unite value. The public key is output as $P = M_m P_d$ and the ciphertext of b is output as $c = Pb + e$. Since CE-VHE does not compute matrices and vectors in binary, we realized a shorter key size, which automatically reduces ciphertext computation time. Hence, the performance of our scheme outperformed the original CE-VHE in terms of public key size and the ciphertext computation cost.



(a) Computational time (b). Communication cost

Fig. 3 Comparative analysis in terms of encryption time and encryption size

Simulation of Gradient Descent Multivariate Linear Regression

Three sets of experiments were conducted in this section. Multivariate linear regression using least squares, gradient descent with division, and gradient descent without division. We used iterations 4 and a learning rate of 0.01 to conduct the gradient descent. The results are multiplied by 100. Different numbers of features were used for both the training and testing data.

Comparative Analysis of Least Square and Gradient Descent with Division Solution

In Fig. 4, we compared least square (the exact solution) and gradient descent with division. Using least square usually does not require any feature scaling and will give an exact solution in the calculation, but the gradient descent requires future scaling and also requires a loop until convergence. However, Euclidian norm of the solution gave the same result. Figure 5 shows the result obtained from the exact solution and gradient descent without division (the approximate solution). Analysis indicates that the error between the exact solution and the approximate solution increases as the dimension of the dataset increases. To make the approximate solution more practical, it is important to select an appropriate dimension of the dataset.

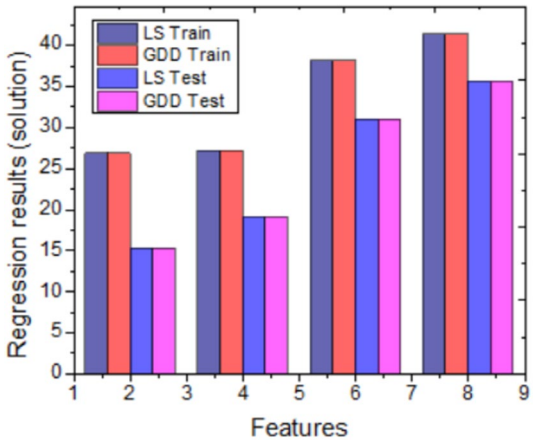


Fig. 4 Multivariate linear regression over plaintext using least square and gradient descent with division

Comparing Exact Solution with Gradient Descent Without Division Over Encrypted Data

In this section, we conducted a simulation to compare and provide the performance of the division-free gradient descent using iteration steps and the least squares approach (exact solution). The division-free gradient descent for minimizing the cost function gives the approximate solution, whereas the exact solution for the cost function gives the exact solution. These experiments were conducted on encrypted data with a learning rate of 0.01. From the result obtained in Table 3, it is obvious that 4 steps were sufficient in the gradient descent method for this size of data, so we used 3 or 4 steps in all the experiments conducted to provide timing for unencrypted and encrypted timing for the multivariate linear regression.

Timing on Unencrypted Data and Encrypted Data

Two sets of experiments are considered here. We simulated multivariate regression computational analysis based on gradient descent with division. This was conducted on encrypted and unencrypted data with different data sizes and features. The dataset was pre-processed by setting the mean

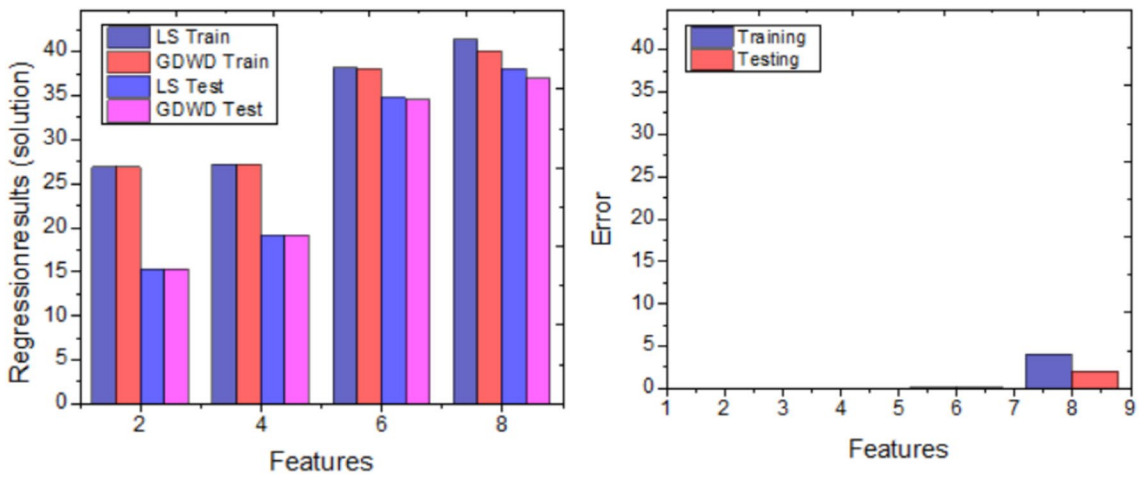


Fig. 5 Multivariate linear regression over plaintext using least square and gradient descent without division and the error difference

Table 3 Comparing the exact solution and approximate solution

Number (features)	Number (training)	Least square (exact solution)	Gradient descent (approx. solution)						
			Number of steps						
			1	2	3	4	5	6	7
2	70	26	18	24	26	26	26	26	26
4	70	27	22	25	27	27	27	27	27
6	70	38	30	34	37	38	37	38	38
8	70	41	33	35	38	39	41	41	41

Table 4 Timing in second for multivariate regression on encrypted data

Number (features)	Number (training)	Number (test)	Train (s)	Regress (s)
2	50	20	0.289000	0.012000
4	50	20	0.323000	0.200000
6	50	20	0.258000	0.220000
8	50	20	0.287000	0.242000
2	70	30	0.247000	0.201000
4	70	30	0.257000	0.212000
6	70	30	0.311000	0.238000
8	70	30	0.332000	0.250000
2	80	20	0.276000	0.101000
4	80	20	0.280000	0.127000
6	80	20	0.304000	0.191000
8	80	20	0.313000	0.208000

Table 5 Timing in second for multivariate regression on encrypted data

Number (features)	Number (training)	Number (test)	Train (s)	Regress (s)
2	50	20	39.832	2.721
4	50	20	14.088	9.132
6	50	20	23.216	15.87
8	50	20	54.617	20.53
2	70	30	75.462	5.674
4	70	30	86.716	7.785
6	70	30	99.243	19.34
8	70	30	121.41	22.67
2	80	40	97.22	5.867
4	80	40	121.46	9.758
6	80	40	132.34	21.85
8	80	40	143.62	23.87

to zero and scaling by standard deviation. The accuracy was set to two (2) decimal places. Real values are multiplied by 100 and rounded to integers. Four (4) iteration steps and learning rate of 0.01 were used.

Multivariate Linear Regression on Unencrypted Data: Table 4 shows the variation of computational time with respect to the different numbers of data features and data dimension sizes. The system runtime was measured as the number of features and the size of the dataset increased. The results demonstrate that as the size of the dataset increases, the computational time also increases. For example, while 2 features and 50 dimensions give a computational cost of 0.289000, 8 features and 100 dimensions of data give a computational cost of 0.313000.

Multivariate Linear Regression on Encrypted Data: Table 5 presents the differences in computational time with respect to the data dimension in encrypted data. In this experiment, training and regression times were measured on encrypted data. As anticipated, as the size and the dimension of the data increase, the training and the testing results also increase. However, the overall time obtained shows that our system is practical and can be used in real-life machine learning over encrypted data applications to ensure confidentiality over datasets with a minimal cost.

Overall Timing on Algorithms and Operations of CE-VHE Scheme

The CE-VHE scheme described in "System model and problem formulation" was implemented using various learning features. To encrypt the confidential version of the multivariate regression, the parameters were set as described above. A summary of the timings for homomorphic key generation, encryption, decryption, and homomorphic operations (addition, linear transformation, and weighted inner product) is presented in Table 6. The result indicates that as the dimension vector increases, the computation time also increases. It is interesting to know that the performance of the operations is better. In particular, the execution time of our linear transformation excels in performance compared to the normal operations in the original VHE [22]. Our system requires less computational cost, and we can see that the overall result shows that our scheme is practical.

Evaluation and Performance Analysis

Mean Absolute Error and Root Mean Squared Error Analysis

Mean Absolute Error (*MAE*)*:* The mean absolute error is a metric used to assess how close forecasts are to the actual outcome. It is determined as follows:

$$\mathrm{MAE} = 1/n \sum_{i=1}^{i=n} |e_i| \tag{12}$$

Root Mean Squared Error (*RMSE*)*:* The difference between the values predicted by a predictor and the values actually observed is measured by the root mean squared error. It computes the standard deviation of the residuals (i.e., prediction errors) as:

$$\mathrm{RMSE} = \sqrt{\sum_{i=1}^{i=n} \frac{e_i^2}{n}}. \tag{13}$$

The error analysis between cipher and plaintext calculations is shown in Fig. 6. In the instance of ciphertext, the MAR is 0.249 and the RMSE is 0.249. The MSE and RMSE values for plaintext are 0.317 and 0.329, respectively. As a result, we can observe that our encrypted text has less error than the plaintext in both circumstances.

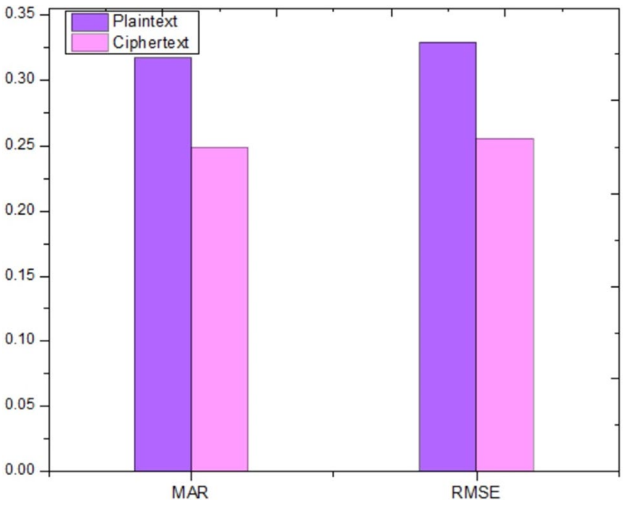


Fig. 6 Analysis of error between ciphertext and plaintext

Security Analysis

Our system safeguards the privacy of the user's sensitive data set and the corresponding regression result. Before submitting the training data set and all sensitive data to the cloud server, they are encrypted using CE-VHE, which is IND-CPA secure and can be reduced to the LWE problem. Without S_0, no malicious user can learn anything about R and a. The query vector w is transformed to produce M, which can also be thought of as the ciphertext's encryption using the secret keys S_0 and S_1. No knowledge of R can be derived from M using the transformation process. As per the CE-EVHE's operations, the regression output is also encrypted by the EVHE, and only the User can recover w using the new secret key S_1. Because the CE-VHE is IND-CPA secure, the secrecy of the regression output is based on the LWE problem. Hence, if LWE problem is hard, computing $S_0 M = S_1 + E$ is also hard. Therefore, given the LWE problem is hard to conjecture, it is infeasible to output the new private key S_0 with the key-switch matrix M. As a result, it is clear that homomorphic encryption systems are safe and may be used to protect the private information of users. Furthermore, by including additional redundant processes like linear transformation and weighted inner product, we can ensure that the server is nothing more than ciphertext. In recap, as long as the users keep S_0 and S_1 secret, the LWE

Table 6 Timing in second for key generation, encryption, decryption and operations

Features	Keygen	Encryption	Decryption	Addition	Linear transformation	Weighted inner product
10	0.130	0.0203	0.058	0.002	0.006	0.012
20	0.138	0.0446	0.071	0.005	0.08	0.010
30	0.148	0.0752	0.211	0.010	0.024	0.014
40	0.156	0.1791	0.230	0.019	0.031	0.017
50	0.157	0.1811	0.310	0.020	0.040	0.019
60	0.171	0.1981	0.370	0.034	0.049	0.022
70	0.182	0.2333	0.421	0.037	0.051	0.025
80	0.199	0.3034	0.500	0.040	0.071	0.028
90	0.209	0.3809	0.560	0.062	0.086	0.060
100	0.222	0.4933	0.723	0.078	0.113	0.080

problem is insurmountable. Using the CE-VHE scheme, no malicious user can learn anything about R, a and w

Correctness of Our System

The findings of the dataset feature vectors in the plaintext and ciphertext domains are compared. Considering v and v' as feature vectors from the plaintext and ciphertext domain features in our approach, the error rate in our system is computed as:

$$Error_{v'} = \| \frac{v - v'}{v'} \| \times 100\% \quad (14)$$

We calculated the error rate for various characteristics and dimensions using simulation. Obviously, it's near to 0 when parameters $w \leq 10, l = 16$ are set correctly. Correctness and security are diametrically opposed. It necessitates a lower limit for the modulus q with a size n and a standard deviation σ. The complexity of the proposed computation and the extent of the standard deviation have an impact on the error rate throughout the computation. The error's relative size in relation to the modulus q should be minimal enough. That is, q should be large enough to account for any errors that may occur throughout homomorphic operations. This will allow us to decrypt the data with the greatest accuracy feasible.

Effectiveness Analysis

We will now look at our system's efficacy to see if it has any practical utility. Encryption and regression are used in this system. On the one hand, we have demonstrated that encryption successfully protects privacy. The multivariate regression phase, on the other hand, is likewise effective in conducting predictions in the experiment, indicating that multivariate learning over encrypted data is possible.

Performance Analysis of the Multivariate Regression Over Encrypted Data

Phase One

The proposed multivariate regression over encrypted data is based on gradient descent and gives an approximate solution. So, we compared it with the exact solution and realized that it introduced some minimal errors. Assuming that w is the solution to the proposed system, the objective function is minimized by defining the error as:

$$Error_w = \left| \frac{f(w) - f(w^*)}{f(w^*)} \right| \times 100\% \quad (15)$$

The accuracy of this model depends on the size of the dataset and the number of iterations required to obtain the solution. This is obvious in Fig. 5, Two to six features with a size of 40 and 60 data dimension gave no error. However, 8 features with data size of 40 and 60 dimension gave errors with each less than 4.5.

Phase Two

It is obvious in Tables 4 and 5 that the operations on ciphertext are a little larger than those on plaintext, as was expected. Operations in the encrypted text domain take nearly twice as long as operations in the plaintext domain. The key switching needed to be computed needed a lot of operations on the number of vectors. We realized that the overhead introduced by the addition operation was very small. This is followed by a linear transformation that is equivalent to a weighted dot product. Comparing the overall result of this work to the reference [10] proposed by M. Nearing, it took almost a day to compute the date size of dimension 100 by 32. In Table 3, we discovered that the implementation would take roughly 8 min to perform a similar-sized task. On the whole, this timing result clearly shows that our approach can perform better than one or two degrees.

Conclusion

In this work, we developed a cloud service that performs private predictive analysis on encrypted health data. We used multivariate regression and homomorphic encryption to ensure security. The data owner shares encrypted data with the cloud server, which then makes predictions without accessing the owner's private data. Our proposed encryption scheme supports three fundamental operations on integer vectors, enabling the efficient and secure computation of integer polynomials up to a certain degree for various applications, including feature extraction, recognition, regression, and data aggregation. Also, while universal homomorphic encryption schemes for general computations remain challenging, we highlight the potential for simpler, application-specific homomorphic encryption schemes with reasonable communication and computation costs. These demonstrate the feasibility and security of performing machine learning tasks on encrypted data.

This study raises questions about optimizing computation, reducing public key size, and using polynomials for tasks, offering promising avenues for future research. Also, centralization introduces a potential single point of failure and may lead to operational delays when the key generation authority is unavailable. On the security front, VHE relies on mathematical algorithms, and any compromise can threaten its security. It's also susceptible to statistical property-based attacks. VHE remains an emerging technology with limited adoption, resulting in a shortage of developer resources,

emphasizing the need for further research to enhance performance and security.

The future work will concentrate on enhancing the execution and expanding the versatility of frameworks to work with encrypted health data. This involves enhancing the execution of practical homomorphic encryption schemes at scale and augmenting the class of capacities in which practical homomorphic encryption schemes can be effectively evaluated. Homomorphic encryption is a rapidly developing field, and thus, we expect that more unpredictable machine learning algorithms connected to large datasets requiring fewer computational datasets may soon be conceivable.

Data Availability All breast cancer datasets used during this study are openly available from the UCI machine learning repository at http://archive.ics.uci.edu/ml as cited in A. F. and A. Asuncion. UCI machine learning repository, 2022.

Declarations

Conflict of interest The authors wish to confirm that there are no known conflicts of interest associated with this publication.

References

1. Gentry C. Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on theory of computing STOC 09, 2009. 19 September. (7)9.
2. Wang C, Cao N, Li J, et al. Secure ranked keyword search over encrypted cloud data. In: Proceedings-30th IEEE International Conference on Distributed Computing Systems. Genova, Itay, 2010. p. 253–262.
3. Rivest RL, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM. 1978;21(2):120–6.
4. Brakerski Z, Vaikuntanathan V. Fully homomorphic encryption from ring-LWE and security for key dependent messages, vol. 6841. LNCS. Lecture notes in computer science. 2011. pp. 505–524.
5. Gentry C, Halevi S, Smart NP. Homomorphic evaluation of the AES circuit, vol. 7417. LNCS. Lecture notes in computer science. 2012. pp.850–867.
6. Fontaine C, Galand F. A survey of homomorphic encryption for nonspecialists. Eurasip J Inf Secur. 2007. https://doi.org/10.1155/2007/13801.
7. Khamitkar S. A survey on fully homomorphic encryption. IOSR J Comput Eng Ver III. 2015;17(6):2278–661.
8. Bogos S, Gaspoz J, Vaudenay S. Cryptanalysis of a homomorphic encryption scheme. Cryptogr Commun. 2018;10(1):27–39.
9. Basilakis J, Javadi B, Maeder A. The potential for machine learning analysis over encrypted data in cloud - based clinical decision support—background and review. In: Australasian workshop on health informatics and knowledge management (HIKM), January 2015. pp. 27–30.
10. Naehrig M, Lauter K, Graepel T. ML confidential: Machine learning on encrypted data. In: In: International conference on information security and cryptology Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. p. 1–21.
11. Bos JW, Lauter K, Naehrig M. Private predictive analysis on encrypted medical data. J Biomed Inform. 2014;50:234–43.
12. Barni M, Failla P, Kolesnikov V, et al. Secure evaluation of private linear branching programs with medical applications, vol. 5789. LNCS. Lecture notes in computer science. 2009. pp. 424–439.
13. Wu D, Haven J. Using homomorphic encryption for large scale statistical analysis. FHE-SI-Report, University Stanford. Technical Report TR-dwu4 (2012).
14. Dwork C. Proceedings of the 5th international conference on theory and applications of models of computation. 2008. pp. 1–19.
15. Jianan Z, Huang R, Yang B. Efficient GSW-style fully homomorphic encryption over the integers. Secur Commun Netw. 2021;2021:1–13.
16. Paillier P, et al. Public-key cryptosystems based on composite degree residuosity classes. In: Eurocrypt, vol. 99. Berlin: Springer; 1999. p. 223–38.
17. Gentry C, et al. Fully homomorphic encryption using ideal lattices. STOC. 2009;9(2009):169–78.
18. Regev O. On lattices, learning with errors, random linear codes, and cryptography. J ACM (JACM). 2009;56(6):34.
19. Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings. In: Annual international conference on the theory and applications of cryptographic techniques. Springer; 2010. pp. 1–23.
20. Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE. SIAM J Comput. 2014;43(2):831–71.
21. Chillotti I, Gama N, Georgieva M, Izabachene M. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In: Advances in cryptology—ASIACRYPT 2016: 22nd international conference on the theory and application of cryptology and information security, Hanoi, Vietnam, December 4–8, 2016, proceedings, part I 22. Springer; 2016. pp. 3–33.
22. Zhou H, Wornell G. Efficient homomorphic encryption on integer vectors and its applications. In: 2014 Information Theory and applications workshop, ITA 2014—conference proceedings, 2014.
23. Haomiao Y, et al. Secure and efficient knn classification for industrial internet of things. IEEE Internet Things J. 2020;7(11):10945–54.
24. Hall R, Fienberg SE, Nardi Y. Secure multiple linear regression based on homomorphic encryption. J Off Stat. 2011;27(4):669.
25. Yao C-C. How to generate and exchange secrets. In: 27th annual symposium on foundations of computer science, 1986. IEEE; 1986. pp. 162–167.
26. Bost R, Popa RA, Tu S, Goldwasser S. Machine learning classification over encrypted data. In: *NDSS*. 2015.
27. Gilad-Bachrach R, Dowlin N, Laine K, Lauter K, Naehrig M, Wernsing J. Cryptonets: applying neural networks to encrypted data with high throughput and accuracy. In: International conference on machine learning, 2016. pp. 201–210.
28. Sadat MN, Jiang X, Al Aziz MM, Wang S, Mohammed N. Secure and efficient regression analysis using a hybrid cryptographic framework: development and evaluation. JMIR Med Inform. 2018;6(1):e14.
29. Jiang Y, Hamer J, Wang C, Jiang X, Kim M, Song Y, Xia Y, Mohammed N, Sadat MN, Wang S. Securelr: secure logistic regression model via a hybrid cryptographic protocol. IEEE/ACM Trans Comput Biol Bioinform. 2018;16:113–23.
30. Hoekstra M, Lal R, Pappachan P, Phegade V, Del Cuvillo J. Using innovative instructions to create trustworthy software solutions. HASP@ ISCA, vol. 11, 2013.
31. Morshed T, Alhadidi D, Mohammed N. Parallel linear regression on encrypted data. In: 2018 16th Annual conference on privacy, security and trust (PST). IEEE; 2018.
32. Ludwig F, Tutz G. Multivariate statistical modelling based on generalized linear models. New York: Springer; 1994.

33. Nirmala MB, Raj P, Johnston L, et al. Handbook of research on cloud infrastructures for big data analytics. Hershey: IGI Global; 2014.
34. Gentry C, Halevi S, Smart NP. Better bootstrapping in fully homomorphic encryption, Vol. 7293 LNCS. Lecture notes in computer science. 2012. pp. 1.
35. Peikert C, Vaikuntanathan V, Waters B. A framework for efficient and composable oblivious transfer, vol. 5157. LNCS Lecture notes in computer science, 2008. pp. 554–571.
36. Naehrig M, Lauter K, Vaikuntanathan V. Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on cloud computing security workshop—CCSW'11. 2011. pp. 113–124.
37. A F., Asuncion A. UCI machine learning repository. 2022. http://archive.ics.uci.edu/ml.
38. Emad EA, Kanaan FA, Helmy T, Azzedin F, Al-Suhaim A. Evaluation of breast cancer tumor classification with unconstrained functional networks classifier. In: Computer systems and applications, IEEE international conference, 2006. pp. 281–287.
39. Sahu Y, Tripathi A, Gupta RK, Gautam P, Pateriya RK, Gupta A. A CNN-SVM based computer aided diagnosis of breast cancer using histogram K-means segmentation technique. Multimedia Tools Appl. 2023;82(9):14055–75.
40. Hu Y. Improving the efficiency of homomorphic encryption schemes. 2013, 103.
41. Barni M, Failla P, Lazzeretti R, et al. Efficient privacy-preserving classification of ECG signals. In: Proceedings of the 2009 1st IEEE international workshop on information forensics and security, WIFS 2009. 2009. pp. 91–95.