## ***Introduction***

* To enhance the efficacy of a learned Ridge Regression model, prior experience in model training suggests using training data from a large and diverse set. A simple way to obtain such a training dataset is to merge data contained in “data silos” collected by different entities.
* In many applications, the data points in these data silos encode sensitive information and are possibly collected by multiple distrustful entities. These entities often cannot (due to legal obligations) share the private data contained in their silos, hence making collaborative analysis on joint data impossible.
* The challenge the paper focuses on: Training a linear regression model on joint data that must be kept *confidential* and/or is owned by multiple distrustful parties.
* The paper proposes an efficient solution in the *two-server model* commonly used by previous works on privacy-preserving machine learning, where no party needs to be trusted to handle the data in the clear. In this model, the computation of the model is outsourced to two non-colluding third parties.
* First phase: collecting private data in *encrypted form* from possible many data-owners
* Second phase: the two third parties engage in the computation of the model itself
* The system described is based only on a simple cryptographic primitive that can be implemented via efficient constructions – a *linearly-homomorphic (partial) encryption* *scheme* (LHE). Such an encryption scheme enables computing the sum of encrypted messages.
* System achieves practical performances when implemented using a standard encryption scheme such as *Pailler’s Cipher*
* Experiments described in the paper demonstrate that for many real scenarios, LHE is all that is needed to privately, yet efficiently, train a ridge regression model on distributed data
* System guarantees confidentiality for the data silos of each research group involved
* In the first phase, the system extends the approach used by Nikolaenko [4] to datasets that are arbitrarily partitioned using the techniques of labeled-homomorphic encryption. This is done to support the multiplication of pairs of ciphertexts encrypted via an LHE scheme. Thus, showing that a solution based only on LHE can handle scenarios more complicated than the horizontally partitioned case.
* For the second phase, to compute the ridge regression model which is a solution of a linear system of the form (Matrix A and vector b are encrypted and must be kept private and the solution is the model) the system designed an ad-hoc two-party protocol that solves this equation using only the linear homomorphic property of the underlying encryption scheme.
* This allows to boost the overall performance, and in particular, to considerably reduce the communication overhead.
* Gradient based solutions for solving the model use iterative algorithms and present the problem of estimating the number of iterations t which is either fixed or chosen adaptively based on the dataset. This can be infeasible in the privacy-preserving setting and the solution that the system designed for solving does not present this issue.

## ***Background***

### *Linear Regression*

* Linear regression learning algorithm- a procedure that on inputs (where ), outputs a vector such that for all .
* In linear regression, the model is computed by minimizing where A is a matrix which contains and b is an -sized vector containing
* In our case, the CEM-UPM linear regression algorithm, = , =, =
* In contrast with other linear regression algorithms, the values of and also change with each iteration while computing the model:

Graphical user interface, text, application

Description automatically generated

### *Cryptographic Tools*

* The privacy-preserving system utilizes homomorphic encryption
* is a finite group. A linearly homomorphic encryption (LHE) scheme for messages in is defined the following three algorithms:

1. **Key-generation algorithm** () takes an input of the security parameter and outputs a matching pair of secret and public keys
2. **Encryption algorithm** () is a randomized algorithm that uses the public key to transform a message from into ciphertext
3. **Decryption algorithm** () is a deterministic algorithm which uses the secret key to recover the original plaintext from the ciphertext

* Security property- it is infeasible for any computationally bounded algorithm to gain extra information about a plaintext when given only its ciphertext and the public key
* Homomorphic property- an operation exists such that for any ciphertexts   
  (­) , it holds that
* This implies that for any ,
* LHE scheme supports multiplication of ciphertexts and plaintexts however **NOT** two or more ciphertexts
* In some cases, being able to perform only linear operations on encrypted data is not sufficient
* An LHE scheme cannot directly handle such an operation
* A general solution to the problem of computing on encrypted data can be obtained via the use of fully homomorphic encryption
* However, systems constructed using fully homomorphic encryptions are still inefficient
* More efficient solutions have been designed for evaluating low-degree polynomials over encrypted data functionalities such as the concept of *labeled homomorphic encryption* [13]

### *Data Representation*

* It is necessary to represent the real values that form the input datasets as elements in the finite set (the message space)
* We assume that for some big integer
* If the linear system is now solved over then the entries of the solution are given as modular residues of and may be different than the entries of the desired model of
* In order to recover the model in form from the model computed over it is possible to apply the rational reconstruction technique [27, 11]

## ***Threat Model and System Overview***

* Consider the setting where the training dataset is not available in the clear to the entity that wants to train the ridge regression model
* The entity needs the help of the party handling the cryptographic keys in order to learn the desired model
* Protocols described in the paper are designed for the following parties:
* **The Data-Owners:** data owners where each data owner has dataset and is willing to share it only if it is encrypted. In our case, these might be different medical centers each having a dataset containing information about patients’ ages and methylation rates.
* **Machine-Learning Engine (MLE):** The party that wants to run a linear regression algorithm on the dataset which is obtained by merging however is only given access to encrypted copies of the datasets. In our case, this would be the Epigenetic Pacemaker using the CEM-UPM linear regression algorithm.
* **Crypto-Service Provider (CSP):** Takes care of initializing the encryption scheme used in the system and interacts with the MLE. CSP manages the cryptographic keys and is the *only* entity capable of decrypting them.
* We assume that MLE and CSP do not collude
* For each pair of parties involved in the protocol there exists a private and authenticated peer-to-peer channel in which communication amongst any two players cannot be eavesdropped
* The goal is to ensure that MLE obtains the desired model without MLE and CSP learning any information about the private datasets

This system’s protocol follows a two-phase architecture:

* **In phase 1**:

1. CSP generates the key pair , stores and makes public
2. Each data owner sends MLE ciphertexts computed using and the values in
3. MLE uses these ciphertexts and the homomorphic property of the encryption scheme in order to obtain encryptions of and (coefficient matrix and vector)

* **In phase 2**:

1. MLE uses the ciphertexts and and private random values in order to obtain encryptions of new values called “masked data”
2. These encryptions are then sent to the CSP, which decrypts and runs a given algorithm on the masked data
3. This computations output is the “masked model”, a vector , which is then sent back from the CSP to the MLE
4. MLE computes the output from

* We say the system is *correct* if the model computed by the MLE is equal to the model computed by the learning algorithm in the clear using the unencrypted data as training data
* We say the system is *private* if the distribution of the masked data sent by MLE to the CSP is independent of the distribution of the masked data sent by the MLE to the CSP is independent of the distribution of the local inputs
* In the protocols description we will see that Phase 1 depends on whether the dataset is horizontally or arbitrarily partitioned
* However, in both cases, MLE gets the encryptions of and and the CSP takes care of initializing the cryptographic primitive and generates the relative keys
* Phase 2 is realized by an interactive protocol between the MLE and the CSP. CSP takes the encryptions of and from the MLE and returns the solution of the system following a pattern referred to as the “masking trick”:

1. The MLE samples a random invertible matrix (size , containing values from the message space) and a random vector and uses the linear homomorphic property of the encryption scheme to compute and . The values are the “masked data”
2. The CSP decrypts and and computes . The vector is the “masked model” which is sent back to the MLE.
3. The MLE computes the desired model as

* If and are sampled uniformly at random, then the distribution of the masked data is independent of and .
* The CEM algorithm requires several iterations in order to optimize the values according to RSS. This will require several iterations of encryption and decryption between MLE and CSE

## ***Protocols Description***

### *Phase 1: Merging the dataset*

### *Phase 2: Computing the model*