Privacy-Preserving Ridge Regression with only Linearly-Homomorphic

Fidaa Shkeer 318483971
Lydia Abu Saleh 318484912
Hind Abu Saleh 318697596
Janan Ghanadri 207570508

University of Haifa -CS 203.4860
Secure Multi-Party Computation
Lecturer: Dr. Adi Akavia

January 2022

1 Introduction

Linear regression is an important statistical tool that models the relationship between some explanatory values (features) and an outcome value using a linear function, and *ridge regression* is one of the most widely-used forms of regression.
In many applications (e.g., personalized medicine) the data points encode sensitive information and are collected by possibly mutually distrustful entities. Often, these entities will not (or cannot) share the private data contained in their silos, making collaborative analysis on joint data impossible.

Our goal : training a linear regression model on joint data that must be kept confidential and/or are owned by multiple parties.

In this work, we present the first approach to privacy-preserving ridge regression that uses only linearly-homomorphic encryption(LHE).
LHE: that is, an encryption schemethat enables computing the sum of encrypted messages

2 System Overview:

- $TheData-Owners$: There are m data-owners DO1, . . . , DOm; each data-owner DOi has a private dataset Di and is willing to share it only if encrypted.

- $TheMachine-LearningEngine(MLE)$: This is the party that wants to run a linear regression algorithm on the dataset D obtained by merging the local datasets D1, . . . , Dm, but has access only to the encrypted copies of them. For this reason, MLE needs the help of the Crypto Service Provider.

- $TheCryptoServiceProvider(CSP)$: takes care of initializing the encryption scheme used in the system and interacts with MLE to help it in achieving its task (computing the linear regression model). CSP manages the cryptographic keys and is the only entity capable of decrypting.

3 Protocols Description:

Phase 1 (merging the local datasets): CSP generates the key pair (sk, pk), stores sk and makes pk public; each DOi sends to MLE specific ciphertexts computed using pk and the values in Di. MLE uses the ciphertexts received and the homomorphic property of the underling encryption scheme in order to obtain encryptions of A and b (the matrix A and the victor b in Aw = b)
A and b are the training data points that form the training set,The solution w is the model.

Phase 2 (computing the model): MLE uses the ciphertexts Encpk (A) and Encpk (b) and private random values in order to obtain encryptions of new values that we call "masked data"; these encryptions are sent to the CSP; the latter decrypts and runs a given algorithm on the masked data. The output of this computation ("masked model") is a vector w˜ that is sent back from the CSP to the MLE. The latter computes the output w from w˜ .

4 Our implementation:

class CSP: takes care of initializing the encryption scheme,and makes pk public, while it keeps sk secret

class MLE: MLE computes encryptions of the entries of A and b,and returns the solution of the system Aw = b following the pattern "masking trick"

startProtocol function: in this function we implement the protocol 1 Phase 1 in the horizontally-partitioned setting (Merging the dataset)

startProtocol2 function: Protocol 3 Phase 2 (Computing the model).

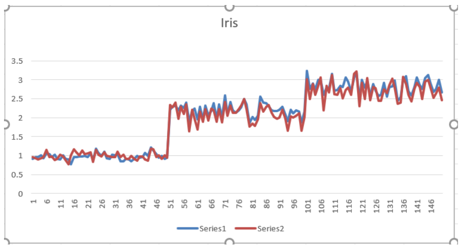
5 without enc file:

The implementation of the protocol without encryption,using the sklearn library

6 Results:

comparing results with encryptin and without

0.917341728	0.964051697
0.961410244	0.931347281
0.951810309	0.896850205
1.012608776	0.922104318
0.923891829	0.956284877
1.056802348	1.152913759
1.037625916	0.956250047
0.955440057	0.970342376
1.02070502	0.877795371
0.918696925	0.907013424
0.898271337	1.018580433
1.000088488	0.968866234
0.911394978	0.875377233
0.898165295	0.767387985
1.013360622	0.96150733
1.033522295	1.10272205
0.841534039	0.962202467
0.842476829	1.022101626
0.918696925	0.907013424
0.896187731	0.908546136 ....



as we can see the results are similar.

Communication complexity: The messages sent during Protocol 1,hor and Protocol 2 contain $\theta(d^2)$ elements from Z_N, This implies a communication cost of $O(d^3 log(nd))$ bits for 1,hor and 2. The attached tables below are showing the analysis according to different terms.
We use the following public parameters: n (number of instances), d (total number of features) and d_k (in the arbitrarily-partitioned setting, the data-owner DO_k holds d_k entries of X, y. That is d_k is the size of D_k). Notice that $\sum_{k=1}^{m} d_k = d(n+1)$.

The running time (secs), for Iris UCI dataset is 4.8264 seconds.

List of Tables

1 Communication complexity in terms of number of plaintexts and ciphertexts sent at each step. 5
2 Summarizes the computational complexity in terms of number of elementary operations for protocol 1. 5
3 Summarizes the computational complexity in terms of number of elementary operations for protocol 3. 6
4 Runtime for each step in each protocol. 6

Communication complexity	
Phase 1 in the horizontally-partitioned setting Protocol 1:	Phase 2 Protocol 3:
-CSP sends pk to each party. -DO_k sends $\frac{d*(d+1)}{2}+d$ ciphertexts to MLE	-MLE sends (d^2+d) ciphertexts to CSP. -CSP sends d plaintexts to MLE

Table 1: Communication complexity in terms of number of plaintexts and ciphertexts sent at each step.

Protocol 1		
Step 1	Step 2	Step 3
-CSP- 1 execution of Gen.	-DO_k- $(n_k - n_{k-1})(\frac{d(d+1)}{2}+d)mult.$ $(n_k - n_{k-1})(\frac{d(d+1)}{2}+d)add.$ $(\frac{d(d+1)}{2}+d)enc.$	-MLE- $m(\frac{d(d+1)}{2}+d)enc-add.$

Table 2: Summarizes the computational complexity in terms of number of elementary operations for protocol 1.

Protocol 3		
Step 1	Step 2	Step 3
-MLE- $(d^3+d^2+d) enc{-}add.$ $(d^3+d^2) cMult.$	-CSP- $(d^2+d) dec.$ 1 solution of d × d linear system	-MLE- $(d^2+d) add.$ $(d^2) mult.$ d rational reconstruct.

Table 3: Summarizes the computational complexity in terms of number of elementary operations for protocol 3.

Protocol.	Step1.	Step2.	Step3.
1	0.033 sec	4.364 sec	0.062 sec
3	0.015 sec	0.046 sec	0.021 sec

Table 4: Runtime for each step in each protocol.

7 articles:

`https://marcjoye.github.io/papers/GJJPY18privateml.pdf`