UNECE

CONFERENCE OF EUROPEAN STATISTICIANS

Expert Meeting on Statistical Data Confidentiality

1-3 December 2021, Poland

Private linear regression: Can we scale up with Big Data?

Giuseppe Bruno (Bank of Italy)
giuseppe.bruno@bancaditalia.it

Abstract

A protocol for secure computation allows two parties with respective private inputs x and y, to collaboratively perform some computation f(x,y) = (f1(x,y), f2(x,y)) such that the first party receives only f1(x,y) and the second party receives only f2(x,y). In this work, we explore the scalability in both number of observations and number of explanatory variables for vertically distributed datasets. The empirical application takes simulated data and carries out benchmark measurement of a suitable extension of the Obliv-c protocol for performing a multivariate regression in an oblivious fashion by harnessing open source libraries. Preliminary results are in single precisions and seem to scale reasonably well with size. The introduction of double precision in the Obliv-c library could be a very boosting feature.

Secure Multiparty Regression: Are we ready to keep data privacy promise?

Giuseppe Bruno[*]

[*] Bank of Italy, Economics and Statistics Directorate.

Abstract

Quite often in our societies data on citizens are split up and held by different institutions. For examples we have Social security, Internal Revenues Services, Central Banks and real estate registry. Assuming it is possible to build a single combined databaes collecting all of the information from the individuals present in the different databases coming from different institutions. Many techniques for secure or private computation relies on executing programs in a data-oblivious way, where the same instructions execute independent of the private inputs which are kept in encrypted form throughout the computation. Designers of such computations today must either put substantial effort into constructing a circuit representation of their algorithm, or use a high-level language and lose the opportunity to make important optimizations or experiment with protocol variations. We gauge here the performance and software modification costs for implementing the data oblivious computation. We show a simple extension of the Obliv-C language which allows application developers to program secure computations without being experts in cryptography.

Here we focus on methods for computing multivariate linear regression as well as goodness of fit statistics index.

JEL classification: C83, D84, E32.
Keywords: Privacy preserving regression, Encryption, Homomorphic encryption.

1. Bank of Italy, Research Area.
[*] giuseppe.bruno@bancaditalia.it
This version June 2021
The views expressed are the authors' only and do not imply those of the Bank of Italy.

1 Introduction and motivation

In sinu urbis hostes sunt, tota civitas plena metus est ac nostri vario exitu sub moenibus urbis pugnant.

Protecting the privacy of individuals and firms is of paramount relevance in our societies. In the last thirty years different techniques taken from the cryptography literature have emerged as possible solution computing function of input variables which must remain private. Secure Multiparty computation (henceforth MPC) is a generic name collecting together different procedures for evaluating a function whose input variables are and should remain private. considering the high value given to personal and sensitive data MPS can be considered none of the gtreat achievements of modern cryptography.Multi-Party Computation. MPC [1, 3] enables two or more parties to collaboratively evaluate a func-tion that depends on private inputs from all parties, while revealingnothing aside from the result of the function. Generic approaches tomulti-party computation (MPC) can compute any function that canbe represented as a Boolean-circuit. Our experiments use Yaosgarbled circuit protocol [33, 58], although our general design is compatible with any Boolean-circuit based MPC protocol.

Multi-Party Computation (MPC) is an area of cryptography concerned with enabling multiple parties to jointly evaluate a public function on their private inputs, without leaking any information about their respective inputs (other than their sizes and whatever can be inferred from the result of the computation).

The paper is arranged in the following way. After this introduction in section 2 we present the original Obliv-c setup and our simple extension. Section 3 describes the multivariate model and the data generation process. Section 4 illustrates the results achieved with our software extension. Finally section 5 provides some concluding remarks.

2 Obliv-C

Obliv-C is an extension of the C programming language that provides data-oblivious programming constructs which allow to embed secure computation protocols [4]. The idea is the following: when performing a multi-party distributed computation with sensitive data we simply need to write the algorithm in the Obliv-C language and compile/link it. The upshot of this is a secure cryptographic protocol that performs this operation without revealing any of the inputs or intermediate values of the computation to any of the parties. Only the outputs are finally shared. The most relevant feature of Obliv-C is the capability to keep all the security features offered by the protocol exposing the data-oblivious methods to the programmer. As an example we report one of the most relevant language extension: the oblivious conditional. Assuming we have two secrete values: x and y and we need to compute the following conditional

$$\textbf{obliv} \quad \textbf{if}\ (x > y) \quad x\ =\ y \tag{1}$$

The oblivious nature of x and y prevent the knowledge of the truth value for the $x > y$ condition even at run time. Therefore the code has to be converted using assignment statements. For example the previous conditional statement could be translated in the following way to remove any control flow dependency:

```
cond = (x>y); // true or false
x = x + cond*(y-x)
```

Obliv-C supports most of the primitive data types available in C. Unfortunately extended precision data type such as **double** or **long** are unavailable. It seems this integration would greatly increase its adoption among practioners. The reference example available in Obliv-C consider a single variable regression where there are two parties. The first one plays the role of the server which owns some data. The second part is the client which has its own data and it is assumed to be interested in knowing the slope of a regression line. The two players employs the Yao protocol to exchange the required data [3, 2]. This is a *constant-round* protocol for securely computing any functionality under the assumption of semi-honest adversaries. It can be shown that the complexity of the Yao protocol is $\mathcal{O}(N^2)$ where N is the number of observations. Therefore we can expect a lack of scalability when the number of observation overcome a given threshold. This is exactly the issue we found in our experiments described in the following sections.

3 The employed Econometric Model

In this work we have employed a linear multivariate econometric model. The data are generate with a random process by means of an R program. In the first model we have generated

according to the following code snippet:

```
set.seed(4321)
 x1 <- rexp(n)
 x2 <- rnorm(n,mean=0,sd=1)
 x3 <- rnorm(n,mean=3,sd=1.5)
```

The first variable follows the exponential distribution with density $f(x) = -\lambda \cdot exp^{-\lambda \cdot x}$ where we have set $\lambda = 1$ and the support is $x \geq 0$. The other two variables are gaussian random variables with mean and standard deviations shown in the above code. We have built a dataset with 10^6 observations and with that extracted four different sets with respectively $10^3,\ 10^4,\ 10^5$ and finally 10^6 observations. The dependent variable has been computed according to the following equation:

$$y_v = 1.5 + .5 \cdot x_1 + .2 \cdot x_2 + .1 \cdot x_3 + \varepsilon \tag{2}$$

This variable has been built with the same lengths as the independent variables datasets. The second model is a linear model with 10 independent variables computed with two exponential distribution, four gaussian , two chi-squared and the other two were rayleigh. These variables have been created as follows:

```
x1 <- rexp  (n)
x2 <- rexp  (n,rate=1.6)
x3 <- rnorm (n,mean=0,sd=1)
x4 <- rnorm (n,mean=3,sd=1.5)
x5 <- rnorm (n,mean=-3,sd=1.1)
x6 <- rchisq(n,df = 3, ncp=0)
x7 <- rchisq(n,df = 5,ncp=0.2)
x8 <- rrayleigh(n, scale = 1)
x9 <- rrayleigh(n, scale = 1.6)
```

The estimated model was represented by the following equation:

$$\begin{aligned} y_z = 0.7 + .5 \cdot x_1 + .2 \cdot x_2 + .1 \cdot x_3 + .9 \cdot x_4 + .1 \cdot x_5 - 1.2 \cdot x_6 + \\ - .25 \cdot x_7 + .8 \cdot x_8 + 1.5 \cdot x_9 + \varepsilon \end{aligned} \tag{3}$$

We ran this second model to check whether there was some nonlinear effect in increasing the number of regressors. All the numerical results and the performance figure are presented and discussed in the 4 section Different data distribution assumptions could be considered. In our preliminary application we have assumed that the server has all the independent variables while the client owns the dependent variables. Different distribution could be considered but they do not affect the overall performances of the privacy regression.

4 The Empirical Application and its results

The performance results for private estimation of model 2, which has been carried out for the four datasets with a growing number of observations, are shown in the table **??**:

Table 1: timinig results for the private regression model.

Value 1	**Value 2**	**Value 3**
α	β	γ
1	1110.1	a
2	10.1	b
3	23.113231	c

The whole set of regressions have been carried out by using synthetic data

5 Concluding Remarks

In this work we have shown a possible avenue for carrying out a linear multivariate regression by extending the Obliv-C framework

References

[1] Oded Goldreich, Micali Silvio, and Wigderson Avi. How to Play any Mental Game or a Complete theorem for protocols with honest majority. *ACM Symposium on the Theory of Computing*, 1987.

[2] Yehuda Lindell and Benny Pinkas. An Efficient Protocol for Secure Two-Party computation in the presence of Malicious Adversaries. *EUROCRYPT 2007, LNCS 4515*, 2007.

[3] Andrew Chi-Chih Yao. How to Generate and Exchange Secrets. *IEEE Symposium on Foundation of Computer Science*, 1986.

[4] Samee Zahur and David Evans. Obliv-C: A Language for Extensible Data-Oblivious Computation. *International Conference on Machine Learning*, 2005.

Appendices

This is the appendix