Optim Lett (2012) 6:431–436
DOI 10.1007/s11590-010-0268-9

ORIGINAL PAPER

Privacy-preserving horizontally partitioned linear programs

Olvi L. Mangasarian

Received: 21 June 2010 / Accepted: 17 November 2010 / Published online: 1 December 2010

Abstract We propose a simple privacy-preserving reformulation of a linear program whose equality constraint matrix is partitioned into groups of rows. Each group of matrix rows and its corresponding right hand side vector are owned by a distinct private entity that is unwilling to share or make public its row group or right hand side vector. By multiplying each privately held constraint group by an appropriately generated and privately held random matrix, the original linear program is transformed into an equivalent one that does not reveal any of the privately held data or make it public. The solution vector of the transformed secure linear program is publicly generated and is available to all entities.

Keywords Security · Privacy-preserving · Linear programming · Horizontally partitioned data

1 Introduction

Recent interest in privacy-preserving classification and data mining [2,7,9–11,13–16], wherein the data to be classified or mined is owned by different entities that are unwilling to reveal the data they hold or make it public, has spread to the field of optimization and in particular linear programming [1,3,4,8,13]. In [1] a number of shortcomings in the privacy-preserving linear programming literature are pointed out. In [8] a method for handling privately held vertical partitions of a linear programming constraint matrix and cost vector is proposed that is based on private random transformations

O. L. Mangasarian (✉)
Computer Sciences Department, University of Wisconsin, Madison, WI 53706, USA
e-mail: olvi@cs.wisc.edu

O. L. Mangasarian
Department of Mathematics, University of California at San Diego, La Jolla, CA 92093, USA

of the corresponding problem variables. In this work we deal with a *horizontal* partition of the equality constraint matrix of a linear program and the corresponding right hand side vector into groups. Each group is owned by a distinct entity wishing to keep its data private, but at the same time wanting a solution to the overall problem based on all the constraints. We address this problem by multiplying each privately held equality constraint by an appropriate and privately generated and held random matrix. This process transforms the original linear program into an equivalent secure linear program that does not reveal any of the privately held data or make it public. The solution vector is then generated publicly and is made available to all entities.

We briefly describe the contents of the paper. In Sect. 2 we give the theory and in Sect. 3 an implementation of our method for a privacy-preserving linear programming formulation using a random transformation of the problem constraints. In Sect. 4 we give numerical examples demonstrating our approach. Section 5 concludes the paper with an open problem.

We describe our notation now. All vectors will be column vectors unless transposed to a row vector by a prime $'$. For a vector $x \in R^n$ the notation x_j will signify either the jth component or jth block of components. The scalar (inner) product of two vectors x and y in the n-dimensional real space R^n will be denoted by $x'y$. The notation $A \in R^{m\times n}$ will signify a real $m \times n$ matrix. For such a matrix, A' will denote the transpose of A, A_i will denote the ith row or ith block of rows of A and $A_{\cdot j}$ the jth column or the jth block of columns of A. A vector of zeros in a real space of arbitrary dimension will be denoted by 0.

2 Privacy-preserving linear programming for horizontally partitioned data

We consider the linear program:

$$\min_{x\in X} \; c'x \quad \text{where } X = \{x \mid Ax = b,\; x \geq 0\}. \tag{1}$$

Here, the matrix $A \in R^{m\times n}$ together with the right hand side vector $b \in R^m$, that is $[A \quad b]$, are divided into p horizontal blocks of $m_1, m_2, \ldots$ and m_p, $(n+1)$-dimensional rows with $m_1 + m_2 + \cdots + m_p = m$. Each block of rows of $[A \quad b]$ corresponding to the index sets $I_1, I_2, \ldots, I_p$, $\bigcup_{i=1}^{p} I_i = \{1, 2, \ldots, m\}$, is "owned" by a distinct entity that is unwilling to make its block of data public or share it with the other entities. We wish to solve this linear program without revealing any privately held data. We shall achieve this by proceeding as follows.

Each of entity $i, i = 1, \ldots, p$, chooses its own privately held random matrix $B_{\cdot I_i} \in R^{k\times m_i}$ with $k \geq m_i$, corresponding to the index set I_i. We thus have the following decompositions:

$$A = \begin{bmatrix} A_{I_1\cdot} \\ A_{I_2\cdot} \\ \vdots \\ A_{I_p\cdot} \end{bmatrix}, \quad b = \begin{bmatrix} b_{I_1} \\ b_{I_2} \\ \vdots \\ b_{I_p} \end{bmatrix}, \quad B = [B_{\cdot I_1}\; B_{\cdot I_2}, \ldots, B_{\cdot I_p}] \in R^{k\times m}. \tag{2}$$

We note immediately that the rank of the randomly generated matrix $B \in R^{k\times m}$ with $k \geq m$ is m [5], which is the reason for choosing $k \geq m$. Utilizing this fact we define the following transformations:

$$\begin{aligned} BA &= [B_{\cdot I_1}\ B_{\cdot I_2}, \ldots, B_{\cdot I_p}] \begin{bmatrix} A_{I_1\cdot} \\ A_{I_2\cdot} \\ \vdots \\ A_{I_p\cdot} \end{bmatrix} \\ &= B_{\cdot I_1} A_{I_1\cdot} + B_{\cdot I_2} A_{I_2\cdot} + \cdots + B_{\cdot I_p} A_{I_p\cdot} \in R^{k\times n}, \end{aligned} \tag{3}$$

and

$$\begin{aligned} Bb &= [B_{\cdot I_1}\ B_{\cdot I_2}, \ldots, B_{\cdot I_p}] \begin{bmatrix} b_{I_1} \\ b_{I_2} \\ \vdots \\ b_{I_p} \end{bmatrix} \\ &= B_{\cdot I_1} b_{I_1} + B_{\cdot I_2} b_{I_2} + \cdots + B_{\cdot I_p} b_{I_p} \in R^{k}. \end{aligned} \tag{4}$$

With these transformations our original linear program (1) turns into the following "secure" linear program:

$$\min_{y\in Y} c'y \quad \text{where} \quad Y = \{y \mid BAy = Bb,\ y \geq 0\}. \tag{5}$$

We use the term "secure" to describe the transformed linear program (5) because it does not reveal any of the privately held data $\begin{bmatrix} A_{I_i\cdot} & b_{I_i} \end{bmatrix}$, $i = 1, \ldots, p$. This is so because for each entity different from entity i, it is impossible to compute either $A_{I_i\cdot}$ from the revealed product $B_{\cdot I_i} A_{I_i\cdot}$, or b_{I_i} from the revealed product $B_{\cdot I_i} b_{I_i}$ *without* knowing the random matrix $B_{\cdot I_i}$ chosen by entity i and known to itself only. See also Remark 3.2 below.

We now relate our original linear program (1) to the transformed linear program (5) as follows.

Proposition 2.1 *Let $k \geq m$ for the random matrix $B \in R^{k\times m}$ of (2). The secure linear program (5) is solvable if and only if the linear program (1) is solvable in which case the solution sets of the two linear programs are identical.*

Proof Because the rank of the random matrix B of (2) is m the following equivalence is obvious:

$$Ax = b \Longleftrightarrow BAx = Bb. \tag{6}$$

Consequently the feasible region X of the original linear program (1) is equivalent to the feasible region Y of the secure linear program (5). Since both objective functions are the same, it follows immediately that both problems have the same solution set.

We turn now to an explicit implementation of the secure linear programming formulation (5).

3 Privacy-preserving for horizontally partitioned linear program (PPHPLP)

Starting with the linear program (1) that is partitioned among p entities as described in Sect. 2, we propose the following algorithm for generating a solution to the linear program without disclosing any of the privately held data.

Alogrithm 3.1 PPHPLP Algorithm

(I) All p entities agree on a value for $k \geq m$, where k is the number of rows of the random matrix $B \in R^{k \times m}$ as defined in (2).

(II) Each entity generates its own privately held random matrix $B_{\cdot I_i} \in R^{k \times m_i}$, $i = 1, \ldots, p$, where m_i is the number of rows held by entity i which results in:

$$B = [B_{\cdot I_1} \;\; B_{\cdot I_2} \ldots B_{\cdot I_p}] \in R^{k \times m}. \tag{7}$$

(III) Each entity i makes public only its matrix product $B_{\cdot I_i} A_{I_i \cdot}$ as well as its right hand side product $B_{\cdot I_i} b_{I_i}$. These products do not reveal either $A_{I_i \cdot}$ or b_{I_i} but allow the public computation of the full constraint matrix needed for the secure linear program (5):

$$BA = [B_{\cdot I_1} A_{I_1 \cdot} + B_{\cdot I_2} A_{I_2 \cdot} + \cdots + B_{\cdot I_p} A_{I_p \cdot}] \in R^{k \times n}, \tag{8}$$

as well as the right hand side for (5):

$$Bb = [B_{\cdot I_1} b_{I_1} + B_{\cdot I_2} b_{I_2} + \cdots + B_{\cdot I_p} b_{I_p}] \in R^{k}. \tag{9}$$

(IV) A public optimal solution vector y to the secure linear program (5) is obtained which, by Proposition 2.1, also solves the original linear program (1).

Remark 3.2 Note that in the above algorithm no entity i reveals its data matrix $A_{I_i \cdot}$ or its right hand side vector b_{I_i}. However, the solution vector y obtained is publicly available. Note further that it is impossible to compute the $m_i n$ numbers constituting $A_{I_i \cdot} \in R^{m_i \times n}$, given only the kn numbers constituting the revealed matrix product $B_{\cdot I_i} A_{I_i \cdot}$, and not knowing $B_{\cdot I_i} \in R^{k \times m_i}$. Similarly it is impossible to compute $b_{I_i} \in R^{m_i}$ from $B_{\cdot I_i} b_{I_i} \in R^k$. Hence, all entities share the publicly computed optimal solution, but without revealing their privately held data.

We turn now to some computational results.

4 Computational results

We demonstrate our results by solving two examples as follows on a 4 Gigabyte machine running $i386\_rhe15$ Linux. We utilize the CPLEX linear programming code [6] within MATLAB [12] to solve our linear programs. The first example has 600 constraints and 1,000 variables, while the second example has 1,000 variables and 1,000 constraints. For both examples, the components of the matrix A were uniformly distributed in the interval $[-50, 50]$, and approximately half of the components of the primal solution x of (1) were uniformly distributed in the interval [0, 10] while the other half was zero. Similarly, approximately half of the constraints of the dual problem were active, that is satisfied as equalities, while the other half were inactive. We used $k = 1{,}000$ in both examples.

Example 4.1 For our first example we generated a random solvable linear program (1) with $m = 600$ and $n = 1{,}000$. We partitioned the rows of A as well as the right hand side vector b into three groups with $m_1 = 100$, $m_2 = 200$ and $m_3 = 300$. We generated three random matrices, with coefficients uniformly distributed in the interval [0, 1] with $B_{\cdot I_1} \in R^{k\times m_1}$, $B_{\cdot I_2} \in R^{k\times m_2}$ and $B_{\cdot I_3} \in R^{k\times m_3}$. We solved the secure linear program (5) and compared its optimal solution with that of (1). The two optimal solutions were identical. Computation time was 11.817 s for the secure linear program (5).

Example 4.2 For our second example we generated a random solvable linear program (1) with $m = 1{,}000$ and $n = 1{,}000$ with approximately half of the primal constraints being redundant. We partitioned the rows of A as well as the right hand side vector b into three groups with $m_1 = 200, m_2 = 300$ and $m_3 = 500$. We generated three random matrices, with coefficients uniformly distributed in the interval [0, 1] with $B_{\cdot I_1} \in R^{k\times m_1}$, $B_{\cdot I_2} \in R^{k\times m_2}$ and $B_{\cdot I_3} \in R^{k\times m_3}$. We solved the secure linear program (5) and compared its optimal solution with that of (1). The two optimal solutions were identical. Computation time was 14.184 s (5).

5 Conclusion and outlook

We have shown how to securely solve a linear program when its equality constraint matrix and its right hand side data are partitioned among entities unwilling to share their data or make it public. Another interesting problem in this realm occurs when the equality constraints of the linear program (1) are inequality constraints instead. The approach proposed here does not work because we cannot multiply these inequality constraints by a random matrix $B \in R^{k\times m}$, even if $B \geq 0$, and preserve the original feasible region of the problem. Furthermore, if we convert the inequality constraints to equality constraints by adding slack variables, multiplying the ith identity matrix coefficient matrix of the slack variables of the ith entity by its privately held random matrix $B_{\cdot i}$ would reveal $B_{\cdot i}$. Hence, treating inequality constraints remains an open problem for future research.

Acknowledgments The research described in this Data Mining Institute Report 10-02, April 2010, was supported by the Microsoft Corporation and ExxonMobil.

References

1. Bednarz, A., Bean, N., Roughan, M.: Hiccups on the road to privacy-preserving linear programming. In: Proceedings of the 8th ACM Workshopon Privacy in the Electronic Society, pp. 117–120 (2009)
2. Chen, K., Liu, L.: Privacy preserving data classification with rotation perturbation. In: Proceedings of the Fifth International Conference of Data Mining (ICDM'05), pp. 589–592. IEEE (2005)
3. Du, W.: A study of several specific secure two-party computation problems. Technical report, Purdue University, West Lafayette, IN (2001). PhD Dissertation. http://www.cis.syr.edu/~wedu/Research/paper/duthesis.ps
4. Du, W., Zahn, Z.: A practical approach to solve secure multi-party computation problems. In: Proceedings of the 2002 Workshop on New Security Paradigms, pp. 127–135 (2002)
5. Feng, X., Zhang, Z.: The rank of a random matrix. Appl. Math. Comput. **185**, 689–694 (2007)
6. ILOG, Incline Village, Nevada: ILOG CPLEX 9.0 User's Manual (2003) http://www.ilog.com/products/cplex/
7. Laur, S., Lipmaa, H., Mielikäinen, T.: Cryptographically private support vector machines. In: KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 618–624. ACM, New York, NY, USA (2006)
8. Mangasarian, O.L.: Privacy-preserving linear programming. Optim. Lett. (2010, to appear). Technical Report 10-01. Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, March ftp://ftp.cs.wisc.edu/pub/dmi/tech-reports/10-01.pdf
9. Mangasarian, O.L., Wild, E.W.: Privacy-preserving classification of horizontally partitioned data via random kernels. In: Stahlbock, R., Crone S.V., Lessman, S. (eds.) Proceedings of the 2008 International Conference on Data Mining, DMIN08, vol. II, pp. 473–479, Las Vegas July (2008). Technical Report 07-03, Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, November (2007)
10. Mangasarian, O.L., Wild, E.W.: Privacy-preserving random kernel classification of checkerboard partitioned data. Ann. Info. Syst. **XIII**, 375–387 (2010). Technical Report 08-02. Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, September 2008
11. Mangasarian, O.L., Wild, E.W., Fung, G.M.: Privacy-preserving classification of vertically partitioned data via random kernels. ACM Trans. Knowl. Discov. Data (TKDD) **2**(3) (2008). Technical Report 07-02, Data Mining Institute, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin, September 2007.
12. MATLAB: User's Guide. The MathWorks, Inc., Natick, MA 01760 (1994–2006). http://www.mathworks.com
13. Vaidya, J., Clifton, C., Zu, M.: Privacy Preserving Data Mining. Springer, New York (2009)
14. Xiao, M.-J., Huang, L.-S., Shen, H. Luo, Y.-L.: Privacy preserving id3 algorithm over horizontally partitioned data. In: Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05), pp. 239–243. IEEE Computer Society (2005)
15. Yu, H., Jiang, X., Vaidya, J.: Privacy-preserving SVM using nonlinear kernels on horizontally partitioned data. In: SAC '06: Proceedings of the 2006 ACM symposium on Applied computing. pp. 603–610. ACM Press, New York, NY, USA (2006)
16. Yu, H., Vaidya, J., Jiang, X.: Privacy-preserving svm classification on vertically partitioned data. In: Proceedings of PAKDD '06. LNCS: Lecture Notes in Computer Science, vol. 3918, pp. 647–656. Springer, Berlin (2006)