Chapter 12

Privacy-Preserving Techniques for Analysis of Medical Data: Secure Multi-Party Computation

By Gabriele Spini, Marie Beth van Egmond, Thijs Veugen and Alex Sangers

DOI: 10.1561/9781638282372.ch12

Published in *Technology in Healthcare: Introduction, Clinical Impacts, Workflow Improvement, Structuring and Assessment* by Brian Pickering, Roland Roller, Holmer Hemsen, Gerrit J. Noordergraaf, Igor Paulussen and Alyssa Venema (eds.). 2024. ISBN 978-1-63828-236-5. E-ISBN 978-1-63828-237-2.

Suggested citation: Gabriele Spini, Marie Beth van Egmond, Thijs Veugen and Alex Sangers. 2024. "Privacy-Preserving Techniques for Analysis of Medical Data: Secure Multi-Party Computation" in *Technology in Healthcare: Introduction, Clinical Impacts, Workflow Improvement, Structuring and Assessment*. Edited by Brian Pickering, Roland Roller, Holmer Hemsen, Gerrit J. Noordergraaf, Igor Paulussen and Alyssa Venema. pp. 154–171. Now Publishers. DOI: 10.1561/9781638282372.ch12.

now
the essence of knowledge

12.1 Introduction

Modern machine-learning techniques require large-scale and well-characterized datasets to achieve their full potential. In the medical domain, this requirement translates to a need to store medical patient data and to combine information from different institutions; the COVID-19 outbreak is an example of a situation where this is deemed crucial [1, 2].

However, the collection, processing, and exchange of personal data are a sensitive matter, and the risks of privacy violations are especially high for medical data. This has led to legal frameworks that regulate and restrict the usage of personal (medical) data, with the General Data Protection Regulation[i] (GDPR) and the Health Insurance Portability and Accountability Act[ii] (HIPAA) being two prominent examples. These regulations mandate informed consent from patients in order

i. https://gdpr-info.eu/.

ii. https://www.govinfo.gov/content/pkg/PLAW-104publ191/pdf/PLAW-104publ191.pdf.

to use the corresponding medical data; however, asking for consent for machine-learning purposes is often impractical, since it is a time-consuming process and since contact with patients may have been lost since the moment of data collection.

This conflict between, on the one hand, the need to gather, combine, and process large amounts of data for better machine-learning techniques and, on the other hand, the need to minimize personal data usage for privacy protection has led to the development of several solutions for privacy-preserving data analysis. In particular, a collection of cryptographic techniques known as Secure Multi-Party Computation, or MPC for short, is being applied more and more in the medical domain. Intuitively, the goal of MPC is to allow several parties to compute the output of a certain function or computation, depending on the private inputs of each party, without actually disclosing information on their inputs to each other.

Within the BigMedilytics project, The Netherlands Organization for Applied Scientific Research (TNO), together with university medical center Erasmus MC and health insurance company Achmea, developed a secure algorithm to predict the number of hospitalization days for heart failure patients. Although the project does not use real patient data in its current phase, the MPC solution that we present is based on the following real-life use case, which serves as a motivating example. In Rotterdam, a group of individuals took part in the 'Rotterdam study', [3] a program in the Epidemiology department of Erasmus MC. Erasmus MC collected data on the lifestyle of patients, for example, their exercising, smoking, and drinking behaviors. Achmea, on the other hand, has claims data from its customers (including several participants in the Rotterdam study), which encompass different aspects, such as hospitalization days and healthcare usage outside of the hospital. Recent work has shown that using machine-learning models on medical data has the potential to predict the survival of heart failure patients [4]. The datasets of Achmea and Erasmus MC, once intersected and combined, could be used to train a prediction model that identifies high-impact lifestyle factors for heart failure and thus, in turn, to recognize high-risk heart failure patients.

However, privacy concerns mean that Erasmus MC and Achmea cannot simply share their data with each other to allow for a straightforward analysis. TNO has therefore developed and implemented the MPC-based Proof of Concept described in this article, which allows Erasmus MC and Achmea to securely train a prediction model without disclosing any personal medical information.

Before we present the details of our solution, we offer an overview of the current landscape of privacy-preserving data analysis techniques, focusing on the medical domain and on solutions that bear resemblance to ours. We will then discuss how our solution compares to these existing techniques. The contents of this chapter are largely based on a scientific article and two blog posts describing the experiment and solution [5–7].

12.1.1 Previous and Related Work

A straightforward approach for privacy-preserving data analytics consists of data anonymization and pseudonymization. These methods ensure that only feature data are revealed, instead of identifiers. However, feature data can often uniquely identify an individual, especially if other related data are acquired through public sources, as shown in several studies [8, 9]. Thus, in practice, data anonymization and pseudonymization offer little guarantee for the protection of the identities of individuals involved in collaborative data analysis.

A more sophisticated and popular approach consists of federated learning, where algorithms are trained on decentralized devices or servers, each possessing its own data, by only exchanging intermediate model coefficients with each other. Federated learning shows great potential to facilitate big data usage for medical applications, in particular for international consortia [10]. Federated learning works fairly straightforward for horizontally partitioned data (where institutions hold the same type of data on different individuals), while vertically partitioned data remain a challenge to be tackled.

Cryptographic solutions such as MPC typically overcome these limitations, but with an inherent overhead in terms of computation time and communication volume compared to non-cryptographic solutions. Specific applications in the medical domain cover a wide range, including, for instance, disclosure of case counts, while preserving the confidentiality of healthcare providers [11]; sharing insights on the effectiveness of HIV treatments, while preserving both the privacy of involved patients and the confidentiality of practitioners' treatment choices [12, 13]; patient risk stratification [14]; privacy-preserving analysis of hospital workflows [15]; secure genome study and secure distributed logistic regression for medical data [16, 17]. Compliance of MPC techniques with the GDPR has been discussed in [18].

With regard to concerns related to working with MPC techniques, a challenge in secure distributed data analysis lies in the combination of different datasets: namely, different institutions hold, in general, data on different individuals, and a first challenge consists of determining which individuals are present in both datasets, and retrieving their relevant features. Various works have been done on 'secure set intersection' (also referred to as 'private set intersection'), [19–22] where the different parties learn which individuals are present in all datasets, but it is guaranteed that no information on individuals outside the intersection will be revealed. To the best of our knowledge, however, no previous work has been published that describes a secure inner join solution where individuals in the intersection are determined but not revealed and where the corresponding feature values are associated with each individual.

Concerning the secure training of a linear regression model on distributed data, a lot of work has been done on a variant of linear regression known as Ridge regression, e.g., [23–32]. Privacy is often preserved by using homomorphic encryption techniques, [24–27] yet there are also implementations that make use of secret sharing, [23] or garbled circuits [28].

The solution that we present here focuses on another linear regression method called Lasso, which has the advantage that once the model has been (securely) trained, less data are needed to evaluate the model. This is a very desirable property for a healthcare-prediction scenario, and in particular for the identification of high-impact factors for heart failure, as described at the beginning of this paragraph: gathering and using only the data that is strictly necessary to apply the model is important to comply with privacy regulations and their data-minimization requirements. In [4], it is even shown that for the prediction of the survival of heart failure patients, training a model on two features alone can yield more accurate predictions than those made using all available features.

12.1.2 The Contributions of the BigMedilytics Project

Within the BigMedilytics project, a solution was developed for (1) computing a secure inner join of two datasets and (2) securely training a Lasso regression model on the obtained (encrypted) data.

Our solution is tailored to the heart failure use-case described above and involves Achmea and Eramus MC as data parties and healthcare information intermediation company ZorgTTP as helper party. The solution has been installed on a test infrastructure by the three involved parties, generated artificial data, and been tested for performance in terms of the quality of the obtained model and efficiency. We elaborate on the benefits and lessons learned from this experiment at the end of this chapter.

12.2 Applications

12.2.1 Description of the Desired Functionality

We first discuss the details of the functionality that we aim to realize. Privacy and security aspects are not considered here and will instead be discussed in paragraph 2.2, following the same structure as the current paragraph.

Table 12.1. AC dataset.

Identifier	Feature $\alpha^{(1)}$	...	Feature $\alpha^{(\ell)}$
a_1	$\alpha_1^{(1)}$	...	$\alpha_1^{(\ell)}$
a_2	$\alpha_2^{(1)}$	...	$\alpha_2^{(\ell)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Table 12.2. EMC dataset.

Identifier	Feature $\beta^{(1)}$	...	Feature $\beta^{(m)}$
b_1	$\beta_1^{(1)}$	...	$\beta_1^{(m)}$
b_2	$\beta_2^{(1)}$	...	$\beta_2^{(m)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

12.2.1.1 Description of the setting and data formatting

We begin with the general setup and a description of the format of the input data. In our setting, two data-providing parties are involved: a healthcare insurance company, Achmea (AC), and a university hospital, Erasmus MC (EMC). We assume that each party owns a dataset where several features of various customers/patients are contained. Each row in the dataset corresponds to a customer or patient, and we refer to it as a record. Specifically, we denote the dataset of Achmea, and its element, as in Table 12.1, and we denote by A its set of identifiers $\{a_1, a_2, \dots\}$.

The dataset of Erasmus MC, on the other hand, is depicted in Table 12.2, and we denote by B the set of identifiers $\{b_1, b_2, \dots\}$.

Before discussing the properties of identifiers and features, we stress the fact that the research described in this article did not use any actual identifiers or features corresponding to existing individuals. For the running time, accuracy, and performance experiments, synthetic data were created or existing public data sets were used.

It was assumed that identifiers in A and B are of the same type; for simplicity, one may think of them as the social security number of a customer/patient. In particular, it is assumed that if a_i and b_j refer to the same person, then $a_i = b_j$. Notice that we are actually interested in the intersection of A and B, as we want to train a regression algorithm on all features.

For what concerns the features, both $\alpha^{(i)}$ and $\beta^{(j)}$ are assumed to be numerical or Boolean. One of the features serves as a target; intuitively, we aim to predict its value as a function of the other feature values. We formalize this intuitive goal in the following sub-paragraphs.

Table 12.3. AC and EMC example datasets, respectively.

Identifier	Hospitalization days
000000	10
111111	5
555555	8
777777	9

Identifier	Hours of exercise per week
000000	0
111111	2
777777	1
999999	3

Table 12.4. Inner join example.

Identifier	Hospitalization days	Hours of exercise per week
000000	10	0
111111	5	2
777777	9	1

Table 12.5. Inner join dataset.

Identifier	Feature $\alpha^{(1)}$	...	Feature $\alpha^{(\ell)}$	Feature $\beta^{(1)}$	...	Feature $\beta^{(m)}$
$a_{i_1} = b_{j_1}$	$\alpha_{i_1}^{(1)}$	...	$\alpha_{i_1}^{(\ell)}$	$\beta_{j_1}^{(1)}$	...	$\beta_{j_1}^{(m)}$
$a_{i_2} = b_{j_2}$	$\alpha_{i_2}^{(1)}$	...	$\alpha_{i_2}^{(\ell)}$	$\beta_{j_2}^{(1)}$	...	$\beta_{j_2}^{(m)}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

12.2.1.2 Inner join of the data

In order to find a correlation among different features, the first necessary step is to identify which features belong to the same customer/patient. Namely, not every person in Achmea is necessarily present in the database of Erasmus MC (as not all customers of AC took part in the social and behavioral study of EMC), and vice versa.

Therefore, the two parties need to (i) compute the intersection of A and B (i.e., identify which persons are represented in both databases) and (ii) ensure that the feature values $\alpha_i^{(\cdot)}$ and $\beta_j^{(\cdot)}$ are identified for all i and j such that the identifiers a_i and b_j coincide and belong to the intersection of A and B (i.e., assign to each identifier in the intersection the corresponding features). In Tables 12.3 and 12.4, an example of the aimed result of this intersection is shown, inspired by the heart failure use-case presented in the introduction paragraph.

More abstractly, Table 12.5 would therefore be obtained, using the notation of Tables 12.1 and 12.2.

This type of operation is commonly referred to as Inner join in the field of database management [33].

The next step is to train a regression algorithm on the data contained in Table 12.5. We remark that, at this point, the identifier column is no longer necessary and will indeed play no role in the regression step.

12.2.1.3 Lasso regression algorithm

Given Table 12.5, we are now interested in finding a way of expressing a given feature (the number of hospitalization days) as a linear combination of the other features or as an approximation of such a linear combination. This is accomplished by training a linear regression model on Table 12.5. In this sub-paragraph, we give some information on this process; for a more complete explanation, the reader can refer to the scientific article describing this experiment [5].

A linear regression problem can be informally expressed by the following question: for a known matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, where n is the number of records and m is the number of features, and a target vector $\mathbf{y} \in \mathbb{R}^{n \times 1}$, can we find a weight vector $\mathbf{w}$ such that the equality $\mathbf{Xw} = \mathbf{y}$ is satisfied? In general, the system is overdetermined, and there exists no solution. Instead, one aims to find $\mathbf{w}$ such that some function of the approximation error vector $\mathbf{Xw} - \mathbf{y}$ (and possibly some other arguments) is minimized.

We focus on a variant of this problem known as Least Absolute Shrinkage and Selection Operator (*Lasso*), [34, 35] which automatically discards features with little impact on the target vector.

12.2.1.4 Gradient descent approach

Gradient Descent (GD) is a general optimization algorithm that finds a local minimum of an objective function. The algorithm takes repeated steps in the opposite direction of the (approximate) gradient of the objective function at the current point. In that way, it moves to the direction of the steepest descent. GD is a building block for many different models, including Ridge regression and support vector machine.

12.2.2 Description of the Secure Solution

12.2.2.1 Aim and assumptions

The goal of this sub-paragraph is to show how the functionality described in paragraph 2.1 can be realized in a secure way. This means that while both parties will learn the output of the Lasso regression (i.e., the model coefficients) trained on the inner join of their datasets,[iii] no other information on the datasets of each party will be disclosed to any other party.

iii. To be completely precise, we also reveal the size of the intersection of the two datasets to the involved parties.

Our secure solution involves a third party, which does not supply any input and does not receive any output (except for the size of the intersection of the two datasets). For our Proof of Concept, this third-party role is taken by ZorgTTP, a company that offers consultancy and practical solutions on the topic of privacy-preserving data analysis in the healthcare sector. The addition of such a party has two benefits, relating to the two steps of our solution: secure inner join and secure Lasso regression. For the first step, the presence of a third party allows us to design a custom, highly efficient protocol; for the second step, we are able to use the MPyC library, [36] which provides useful building blocks but requires at least three parties to guarantee security.

Before discussing the details of our solution, we give a brief introduction to Secure Multi-Party Computation. Notice that we chose to present cryptographic concepts with a focus on intuition so as not to burden the reader with an unnecessary level of formalism. The reader can refer to [37, 38] for a more formal discussion of general cryptographic concepts (including cryptographic hash functions, homomorphic encryption, and secret sharing) and to [39, 40] for an in-depth discussion of MPC and secret sharing.

12.2.2.2 Secure inner join

As outlined in paragraph 2.1, in order to realize a protocol that securely implements our desired functionality, the first step to be performed is to compute the so-called inner join of the datasets of Achmea and Erasmus MC. Namely, we need to obtain a database with the identifiers that are present in both the datasets of Achmea and Erasmus MC, and with the corresponding features coming from both datasets. Notice that we do not wish to reveal the dataset obtained in this way to any party, as it would still contain highly sensitive personal data (in the case of an application involving real data). The inner-join database will thus remain secret, yet computing the coefficients of a Lasso regression model on this secret dataset will be possible.

Our solution makes use of three core components: (keyed) cryptographic hash functions, (additively) homomorphic encryption, and 2-out-of-2 secret sharing.

- Hash functions. A cryptographic hash function is a deterministic function H: $\mathcal{D} \to \mathcal{C}$, that maps any alphanumeric string $s \in \mathcal{D}$ to another alphanumeric string $H(s) = z \in \mathcal{C}$, called *digest*, of fixed length. Such a function enjoys the property that given a digest $z \in \mathcal{C}$, it is unfeasible to compute a string s such that $H(s) = z$.
- Homomorphic encryption. An (additively) homomorphic encryption scheme is a public-key encryption scheme such that there exists a special operations on ciphertexts $\boxplus$ with $\mathrm{Dec}(\mathrm{Enc}(m_1) \boxplus \mathrm{Enc}(m_2)) = m_1 + m_2$, and similarly for the "minus" operation.

Table 12.6. Encrypted data sent to ZorgTTP by AC and EMC, respectively.

Hashed identifier	Encrypted feature α
$H(a_1 \| r)$	$[\alpha_1]_{\text{AC}}$
$H(a_2 \| r)$	$[\alpha_2]_{\text{AC}}$
$\vdots$	$\vdots$

Hashed identifier	Encrypted feature β
$H(b_1 \| r)$	$[\beta_1]_{\text{AC}}$
$H(b_2 \| r)$	$[\beta_2]_{\text{AC}}$
$\vdots$	$\vdots$

Table 12.7. Encrypted data obtained and intersected by ZorgTTP.

Matching identifiers	Feature α	Feature β	Value AC	Value EMC
$H(a_i \| r) = H(b_j \| r)$	$[\alpha]_{\text{AC}}$	$[\beta]_{\text{EMC}}$	$[\alpha - z]_{\text{AC}}$	$[\beta - s]_{\text{EMC}}$

Table 12.8. Final tables of secret-shares obtained by AC and EMC, respectively.

α-share	β-share
$\alpha - z$	s

α-share	β-share
z	$\beta - s$

- 2-out-of-2 secret sharing. This building block can be seen as a form of keyless encryption, distributed among two parties, and works as follows: given a secret (numerical) value s, two elements s_1 and s_2 called shares are randomly sampled, but subject to the condition that $s_1 + s_2 = s$. Then, s_1 is assigned to a party and s_2 to another party; in this way, each party has individually no knowledge of s (since the share s_i that they have is a random number), but the original secret value s can be reconstructed when the two parties cooperate and communicate their shares to each other.

The presence of a third party (ZorgTTP) allows us to design a novel, highly efficient protocol for secure inner join, which we believe to be of independent interest. The goal is for AC and EMC to obtain a secret-shared version of the features from Table 12.5. Our secure inner join protocol between AC, EMC, and ZorgTTP uses cryptographic hash functions, and both AC and EMC have an (additively) homomorphic encryption key pair; we used SHA-256 as a hash function and the Paillier homomorphic encryption scheme in our implementation [41, 42].

We did not discuss how the secure inner join was realized; the reader can refer to the article [5] for a detailed discussion.

12.2.2.3 Secure lasso regression

Once the steps of paragraph 2.2 have been performed, we obtain a '2-out-of-2 secret-shared' version of Table 12.5: namely, Achmea and Erasmus MC each have a table filled with apparently random numbers, but if they were to add up the corresponding numbers, they would obtain exactly Table 12.5.

Recall that our purpose is to train a linear regression model – specifically Lasso – on this table. Now letting Achmea and Erasmus MC communicate their datasets to each other in order to reconstruct Table 12.5, and then train the regression model, is clearly not an option: the information that they would obtain consists of personal data, the exchange of which has to be prevented.

Instead, we present a solution that is able to compute the regression coefficients from the two datasets without leaking information on their content.

The fundamental building block that allows us to design and implement this solution is Shamir Secret Sharing, which can be seen as a form of key-less distributed encryption (just like 2-out-of-2 secret sharing), but with different privacy and reconstruction guarantees; in particular, Shamir Secret Sharing can ensure that shares are distributed among three parties instead of two. We make use of the software platform MPyC, [36] which implements this form of secret sharing and other useful communication and computation tools.

Such a secret-sharing scheme can be used to construct MPC protocols; assume that the three involved parties (Achmea, Erasmus MC, and ZorgTTP) have access to a Shamir Secret Sharing scheme. Let us assume that parties wish to perform some computation on a value α (held by Achmea) and β (held by Erasmus MC). The three parties can then proceed as follows: first, Achmea secretshares α, i.e., computes $(\alpha_1, \alpha_2, \alpha_3) = \mathit{Share}(\alpha)$, such that Achmea, Erasmus MC, and ZorgTTP will receive α_1, α_2, and α_3, respectively. Notice that by the privacy property of the scheme, no information on α is leaked at this point. Erasmus MC then similarly secret-shares β, i.e., computes and distributes $(\beta_1, \beta_2, \beta_3) = \mathit{Share}(\beta)$.

The key property now is that for any operation that the parties wish to perform on the values α and β, there exists a corresponding operation that can be performed *on the shares* α_i, β_i, resulting in some other sharing s_1, s_2, s_3, in such a way that no information at all is leaked on α, or β.

It then becomes possible to evaluate a complex algorithm such as Lasso regression on several features of Achmea and Erasmus MC: parties can secret-share their features, then decompose the Lasso regression into basic operations, and perform the corresponding operations on the shares. Eventually, they will obtain shares of the regression coefficients; Achmea and Erasmus MC at this point simply need to exchange their shares with each other in order to reconstruct the coefficients.

12.3 Results

In this paragraph, we first present the security results of our solution. We only give a brief overview and once again refer to [5] for a detailed discussion.

12.3.1 Security Results

The security of our solution is guaranteed under the following assumptions: First of all, we assume that any two parties are connected by secure channels. We assume that parties do follow the instructions of the protocol; in the cryptographic lexicon, they are thus assumed to be semi-honest. We assume that no party will collude with any other party and exchange information with them.

Under the above conditions, it is guaranteed that the only information that will be revealed are regression coefficients and the size of the intersection between the datasets of Achmea and Erasmus MC.

12.3.2 Running Time

We implemented our solution in Python. In order to test the efficiency of our implementation, we ran several experiments on three machines, under the control of Achmea, Erasmus MC, and ZorgTTP, respectively, and geographically separated.

The experiments include the secure inner-join computation and the protocol to securely train a Lasso regression algorithm as described in paragraph 2.2. For the same reason, no test data are extracted from these artificial datasets.

In order to test the efficiency of our solution, we sampled artificial datasets with an increasing number of records and features, and ran several instances of our solution. We vertically split the dataset into two datasets with an equal number of features (up to one difference) and a complete overlap in record IDs, i.e., the identifiers in the Achmea dataset were identical to those of the Erasmus MC dataset for each iteration.

The total running time (thus encompassing both secure inner join and secure Lasso regression) is shown in Figures 12.1 and 12.2. Our solution thus takes roughly 3500 seconds, slightly less than 1 hour, to process two datasets with 5000 records each and a total of 30 features. Moreover, the running time of our solution scales linearly with the number of records and features.

12.3.3 Performance and Accuracy Results

To test the performance and accuracy of our secure model, we use the 'Medical Costs' dataset by Brett Lantz [43]. This public dataset contains 1338 records of patients with 12 features each (including age, BMI, children, gender, and medical costs), of which four are numerical and eight are Boolean. We centered and scaled the data in advance, such that the feature values are between 0 and 1. We also split them into a train and a test set (10% of the data, randomly selected).

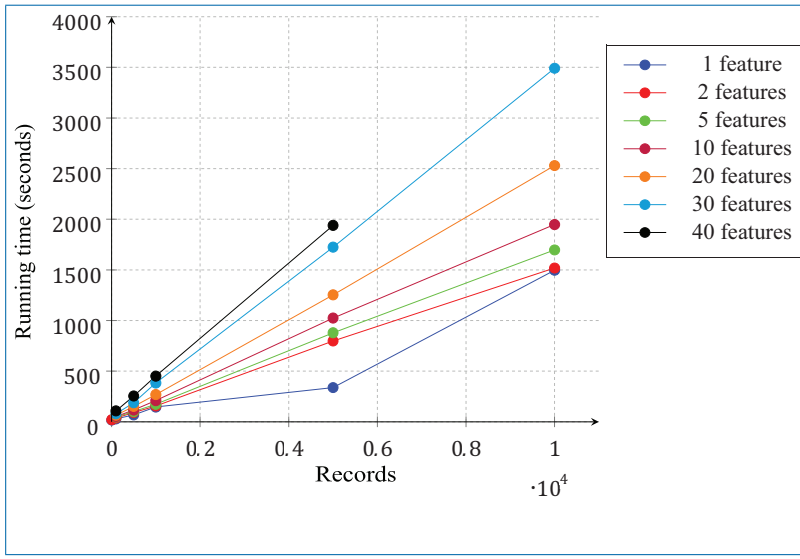

Figure 12.1. Total running time of the experiments as a function of the number of records (median values).

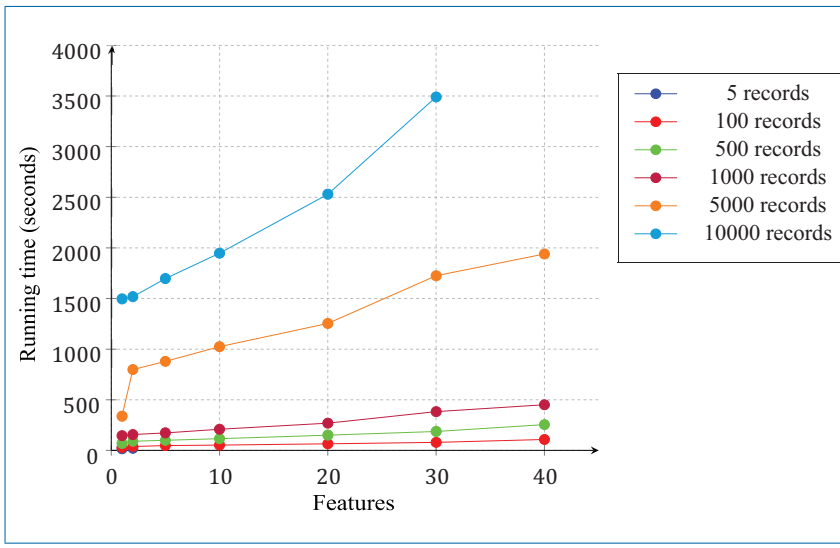

Figure 12.2. Total running time of the experiments as a function of the number of features (median values).

12.3.4 Performance of Lasso Regression

To test the performance of our solution, we compare the results of our secure model with the non-secure scikit-learn Lasso model [44]. Note that the secure inner join has no influence on the performance of the Lasso regression. Therefore,

Table 12.9. Comparison plaintext model and Sklearn Lasso: objective, R^2, mean squared error, and mean absolute error.

Model	Obj	R^2	MSE	MAE	Intercept
scikit-learn	0.009	0.66	0.012	0.082	0.39
our secure model	0.013	0.74	0.008	0.062	0.18
Abs. diff.	0.004	0.08	0.004	0.020	0.21

Table 12.10. Comparison plaintext model and Sklearn Lasso: coefficients.

Model	c1	c2	c3	c4	c5	c6	c7	c8	c9	c10	c11
scikit-learn	0.08	0.01	0	0	0	−0.03	0	0	0	0	0
our secure model	0.17	0.10	0.001	0	0	−0.19	0.18	0	0	0	0
Abs. diff.	0.09	0.09	0.001	0	0	0.11	0.18	0	0	0	0

as input into our secure model, the data is secretly shared between the three parties.

We trained our secure model on 11 features of the train set for predicting the (numerical) target feature of medical costs by varying λ and tolerance. We found the optimal choice, leading to a good fit (R^2, mean squared error) and enough coefficients set to zero to be $\lambda = 0.001$ and tolerance $= 0.0001$. Applying the trained model to our test set, we achieve an R^2 of 0.70, a mean squared error of 0.0086, a mean absolute error of 0.062, and an objective of 0.013. As a validation of the solving method that we used, we compare these results with the (highly optimized) Lasso model of scikit-learn, [44] using the same parameters. After the model was trained on the train set, on the test set, we found an R^2 value of 0.66, a mean squared error of 0.012, a mean absolute error of 0.082, and an objective of 0.0090. Although the goodness-of-fit measures of our secure model are better than those of the scikit-learn model, it has a larger objective value. In Tables 12.9 and 12.10, one can see that in the scikit-learn model, two more coefficients are set to zero, which is one of the aims of Lasso. Therefore, we can conclude that our secure model has good performance, although the (highly optimized) scikit-learn model performs slightly better.

12.4 Benefits

In light of the results shown in paragraph 3, we conclude that our solution does provide a viable way of securely training a Lasso regression model on distributed patient data in a privacy-preserving way. In particular, the good quality of the obtained

model, together with its satisfying efficiency in a fairly realistic setup, makes our solution a promising tool for privacy-preserving analysis of distributed patient data.

On a more general level, the use of MPC can help organizations to comply with privacy regulations when analyzing medical data. In particular, MPC can be seen as a strong form of pseudonymization, and the outcome itself (the only data being revealed) is typically anonymous. Moreover, since all parties are required to participate in the MPC process, they can ensure that the data are only being used for a purpose they have vetted; this is in contrast with more classical solutions, where data owners need to rely on the bona fide of data processors for what concerns the type of analysis that is run on the data.

12.5 Learnings

A number of lessons can be extracted from the process of devising and implementing the solution described in the previous paragraphs.

- MPC is a powerful tool that has reached a high level of maturity. In particular, even relatively complex functionalities like training machine-learning models can be realized with it.
- Using MPC does incur a large computational and communication overhead, like for many other privacy-enhancing technologies; in particular, this means that training a machine-learning model with MPC is significantly slower than on plaintext data. However, the efficiency guarantees for this type of computation are not very stringent, and a running time of around an hour is generally deemed acceptable.
- While conclusions from this and other projects indicate that MPC can help organizations in reconciling data analysis with privacy regulations, there are no explicit national or international regulations that concern MPC. One of the consequences of this is that the digital privacy impact assessment of MPC solutions may take more time to be reviewed by the relevant parties.
- According to Achmea and Erasmus MC, the process was technically and organizationally complex to implement in a corporate environment. MPC is a complex technology on both conceptual and technical levels, and more familiarity with it is needed in order to speed up its adoption in production environments.
- In more general terms, moving a to higher level of technology readiness requires a bigger focus on non-technical challenges, such as compliance and legal aspects, and to ensure that employees and management are properly involved in the process and get acquainted with the used techniques, which constitutes a time-consuming process.

Additional Funding

The authors of this Chapter were supported in part by the Future PET project, financed by the Ministry of Justice and Security of the Netherlands. Further funding was provided by TNO.

References

[1] Dwivedi YK, Hughes DL, Coombs C, Constantiou I, Duan Y, Edwards JS, Gupta B, Lal B, Misra S, Prashant P, *et al.* Impact of COVID-19 pandemic on information management research and practice: Transforming education, work and life. International Journal of Information Management. 2020. p. 102211.

[2] Raisaro JL, Marino F, Troncoso-Pastoriza J, Beau-Lejdstrom R, Bellazzi R, Murphy R, Bernstam EV, Wang H, Bucalo M, Chen Y, Gottlieb A, Harmanci A, Kim M, Kim Y, Klann J, Klersy C, Malin BA, Méan M, Prasser F, Scudeller L, Torkamani A, Vaucher J, Puppala M, Wong STC, Frenkel-Morgenstern M, Xu H, Musa BM, Habib AG, Cohen T, Wilcox A, Salihu HM, Sofia H, Jiang X, Hubaux JP. SCOR: A secure international informatics infrastructure to investigate COVID-19. Journal of the American Medical Informatics Association. July 2020. 27(11), 1721–1726. https://doi.org/10.1093/jamia/ocaa172.

[3] Ikram MA, Brusselle GG, Murad SD, van Duijn CM, Franco OH, Goedegebure A, Klaver CC, Nijsten TE, Peeters RP, Stricker BH, *et al.* The Rotterdam study: 2018 update on objectives, design and main results. European Journal of Epidemiology. 2017. 32(9), 807–850.

[4] Chicco D, Jurman G. Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. BMC Medical Informatics and Decision Making. 2020. 20(1), 16.

[5] van Egmond MB, Spini G, van der Galien O, IJpma A, Veugen T, Kraaij W, Sangers A, Rooijakkers T, Langenkamp P, Kamphorst B, *et al.* Privacypreserving dataset combination and lasso regression for healthcare predictions. BMC Medical Informatics and Decision Making. 2021. 21(1), 1–16.

[6] van Egmond MB. Identifying heart failure patients at high risk using MPC. https://medium.com/applied-mpc/identifying-heart-failure-patients-at-high-risk-using-mpc-ab8900e75295, accessed: 2022-11-25.

[7] van Egmond MB. Identifying high-risk factors for diseases while preserving privacy. https://www.tno.nl/en/newsroom/insights/2021/09/identifying-high-risk-factors-diseases/, accessed: 2022-11-25.

[8] Narayanan A, Shmatikov V. Robust de-anonymization of large sparse datasets. IEEE Symposium on Security and Privacy. 2008. pp. 111–125.
[9] Sweeney L. Weaving technology and policy together to maintain confidentiality. The Journal of Law, Medicine & Ethics. 1997. 25(2–3), 98–110.
[10] Zerka F, Barakat S, Walsh S, Bogowicz M, Leijenaar RT, Jochems A, Miraglio B, Townend D, Lambin P. Systematic review of privacy-preserving distributed machine learning from federated databases in health care. JCO Clinical Cancer Informatics. 2020. 4, 184–200.
[11] Emam KE, Hu J, Mercer J, Peyton L, Kantarcioglu M, Malin BA, Buckeridge DL, Samet S, Earle C. A secure protocol for protecting the identity of providers when disclosing data for disease surveillance. Journal of the American Medical Informatics Association. 2011. 18(3), 212–217. https://doi.org/10.1136/amiajnl-2011-000100.
[12] De optimale HIV behandeling vinden met MPC, https://www.tno.nl/nl/tno-insights/artikelen/de-optimale-hiv-behandeling-vinden-met-mpc/, accessed: 2020-10-26.
[13] Spini G, Mancini E, Attema T, Abspoel M, de Gier J, Fehr S, Veugen T, van Heesch M, Worm D, De Luca A, *et al.* New approach to privacypreserving clinical decision support systems for HIV treatment. Journal of Medical Systems. 2022. 46(12), 1–11.
[14] Dong X, Randolph DA, Weng C, Kho AN, Rogers JM, Wang X. Developing high performance secure multi-party computation protocols in healthcare: A case study of patient risk stratification. AMIA Summits on Translational Science Proceedings. 2021. 200.
[15] Spini G, van Heesch M, Veugen T, Chatterjea S. Private hospital workflow optimization via secure k-means clustering. Journal of Medical Systems. 2020. 44(1), 8:1–8:12. https://doi.org/10.1007/s10916-019-1473-4.
[16] Zhang Y, Dai W, Jiang X, Xiong H, Wang S. Foresee: Fully outsourced secure genome study based on homomorphic encryption. BMC Medical Informatics and Decision Making. 2015. Vol. 15, p. S5. Springer.
[17] Shi H, Jiang C, Dai W, Jiang X, Tang Y, Ohno-Machado L, Wang S. Secure multi-party computation grid logistic regression (SMAC-GLORE). BMC Medical Informatics and Decision Making. 2016. 16(S–3), 89. https://doi.org/10.1186/s12911-016-0316-1.
[18] van Haaften W, Sangers A, van Engers T, Djafari S. Coping with the general data protection regulation; anonymization through multi-party computation technology. 2020.
[19] Cristofaro ED, Tsudik G. Practical private set intersection protocols with linear complexity. In: Sion, R. (ed.) Financial Cryptography and Data Security, 14th International Conference, FC 2010, Tenerife, Canary Islands, Spain,

January 25–28, 2010, Revised Selected Papers. Lecture Notes in Computer Science. 2010. Vol. 6052, pp. 143–159. Springer. https://doi.org/10.1007/978-3-642-14577-3_13.

[20] Freedman MJ, Nissim K, Pinkas B. Efficient private matching and set intersection. Eurocrypt. Lecture Notes on Computer Science. 2004. Vol. 3027, pp. 1–19.

[21] Pinkas B, Rosulek M, Trieu N, Yanai A. Spot-light: Lightweight private set intersection from sparse OT extension. Cryptology ePrint Archive. 2019.

[22] Pinkas B, Schneider T, Tkachenko O, Yanai A. Efficient circuit-based PSI with linear communication. Advances in Cryptology – EUROCRYPT. 2019. pp. 122–153. https://doi.org/10.1007/978-3-030-17659-4_5.

[23] Bogdanov D, Kamm L, Laur S, Sokk V. Rmind: A tool for cryptographically secure statistical analysis. IEEE Transactions on Dependable and Secure Computing. 2008. 15(3), 481–495.

[24] Dankar FK, Brien R, Adams C, Matwin S. Secure multi-party linear regression. EDBT/ICDT Workshops. 2014. pp. 406–414. Citeseer.

[25] de Cock M. Dowsley R, Nascimento AC, Newman SC. Fast, privacy preserving linear regression over distributed datasets based on pre-distributed data. Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security. 2015. pp. 3–14. ACM.

[26] Hall R, Fienberg SE, Nardi Y. Secure multiple linear regression based on homomorphic encryption. Journal of Official Statistics. 2011. 27(4), 669.

[27] Hu S, Wang Q, Wang J, Chow SSM, Zou Q. Securing fast learning! ridge regression over encrypted big data. IEEE Trustcom/BigDataSE/ISPA. Augustus 2016. pp. 19–26. https://doi.org/10.1109/TrustCom.2016.0041.

[28] Nikolaenko V, Weinsberg U, Ioannidis S, Joye M, Boneh D, Taft N. Privacypreserving ridge regression on hundreds of millions of records. Security and Privacy, IEEE Symposium on. pp. 334–348. 2013.

[29] Chen YR, Rezapour A, Tzeng WG. Privacy-preserving ridge regression on distributed data. Information Sciences. 2018. 451–452, 34–49. https://doi.org/10.1016/j.ins.2018.03.061.

[30] Blom F, Bouman NJ, Schoenmakers B, de Vreede N. Efficient secure ridge regression from randomized Gaussian elimination. IACR Cryptol. ePrint Arch. 2019. 773.

[31] Gascón A, Schoppmann P, Balle B, Raykova M, Doerner J, Zahur S, Evans D. Privacy-preserving distributed linear regression on high-dimensional data. Proceedings on Privacy Enhancing Technologies. 2017. 4, 345–364.

[32] Giacomelli I, Jha S, Page CD, Yoon K. Privacy-preserving ridge regression on distributed data. IACR Cryptology ePrint Archive. 2017. 707.

[33] Join Clause (SQL). https://en.wikipedia.org/wiki/Join_(SQL)#Inner_join, accessed: 2020-10-06.
[34] Santosa F, Symes WW. Linear inversion of band-limited reflection seismograms. SIAM Journal on Scientific and Statistical Computing. 1986. 7(4), 1307–1330.
[35] Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological). 1996. 58(1), 267–288.
[36] Schoenmakers B. MPyC – secure multiparty computation in python, https://github.com/lschoe/mpyc.
[37] Katz J, Lindell Y. Introduction to Modern Cryptography, Second Edition. CRC Press. 2014. https://www.crcpress.com/Introduction-to-Modern-Cryptography-Second-Edition/Katz-Lindell/p/book/ 9781466570269.
[38] Menezes A, van Oorschot PC, Vanstone SA. Handbook of Applied Cryptography. CRC Press. 1996. https://doi.org/10.1201/9781439821916.
[39] Cramer R, Damgård I, Nielsen JB. Secure Multiparty Computation and Secret Sharing. Cambridge University Press. 2015. http://www.cambridge.org/de/academic/subjects/computer-science/cryptography-cryptology-and-coding/secure-multiparty-computation-and-secret-sharing?format=HB&isbn=9781107043053.
[40] Lindell Y. Secure multiparty computation. Communications of the ACM. 2021. 64(1), 86–96. https://doi.org/10.1145/3387108.
[41] FIPS 180-4. Secure Hash Standard. National Institute of Standards and Technology. 2005. p. 36.
[42] Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) Advances in Cryptology – EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2–6, 1999, Proceeding. Lecture Notes in Computer Science, vol. 1592, pp. 223–238. Springer (1999). https://doi.org/10.1007/3-540-48910-X_16.
[43] Medical Costs Dataset, Brett Lantz, https://www.kaggle.com/mirichoi0218/insurance. Accessed: 2020-10-26.
[44] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011).