Secure Linear Regression on Vertically Partitioned Datasets

Adria Gascon    Phillipp Schoopmann    Borja Balle
Mariana Raykova    Samee Zahur    Jack Doerner
David Evans
Predictive Model

- Given samples \((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\)
  - \(x_i \in \mathbb{R}^d, y_i \in \mathbb{R}\)
- Learn a function \(f\) such that \(f(x_i) = y_i\)

<table>
<thead>
<tr>
<th>Patient</th>
<th>Blood Count</th>
<th>Heart Conditions</th>
<th>Digestive Track</th>
<th>Medicine Effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RBC</td>
<td>WBC</td>
<td>...</td>
<td>Murmur</td>
</tr>
<tr>
<td>A</td>
<td>3.9</td>
<td>10.0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>4.5</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2.5</td>
<td>11</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>4.3</td>
<td>5.3</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>:</td>
<td></td>
<td>:</td>
</tr>
</tbody>
</table>
## Linear Regression

- Given samples \((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\)
  - \(x_i \in \mathbb{R}^d, y_i \in \mathbb{R}\)
- Learn a function \(f\) such that \(f(x_i) = y_i\)

<table>
<thead>
<tr>
<th>Patient</th>
<th>Blood Count</th>
<th>Heart Conditions</th>
<th>Digestive Track</th>
<th>…</th>
<th>Medicine Effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RBC</td>
<td>WBC</td>
<td>…</td>
<td>Murmur</td>
<td>Arrhythmia</td>
</tr>
<tr>
<td>A</td>
<td>3.9</td>
<td>10.0</td>
<td>…</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>4.5</td>
<td>…</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>2.5</td>
<td>11</td>
<td>…</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>4.3</td>
<td>5.3</td>
<td>…</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

\(f\) is well approximated by a linear map \(y_i \approx \theta^T x_i\)
Secure Computation

- **Shared database** - \((x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\) do not belong to the same party
- **Compute** \(\theta\) **securely** \(y_i \approx \theta^T x_i\)

<table>
<thead>
<tr>
<th>Patient</th>
<th>Blood Count</th>
<th>Heart Conditions</th>
<th>Digestive Track</th>
<th>Medicine Effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RBC</td>
<td>WBC</td>
<td>Murmur</td>
<td>Arrhythmia</td>
</tr>
<tr>
<td>A</td>
<td>3.9</td>
<td>10.0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>2.5</td>
<td>11</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>4.3</td>
<td>5.3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Patient Blood Count

Heart Conditions

Digestive Track

Medicine Effectiveness

<table>
<thead>
<tr>
<th>Patient</th>
<th>Blood Count</th>
<th>Heart Conditions</th>
<th>Digestive Track</th>
<th>Medicine Effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3.9</td>
<td>10.0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>2.5</td>
<td>11</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>4.3</td>
<td>5.3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Cryptography in the RAM Computation Model
## Horizontally Partitioned Database

- Different rows belong to different parties
  - E.g., each patient has their own information

<table>
<thead>
<tr>
<th>Patient</th>
<th>Blood Count</th>
<th>Heart Conditions</th>
<th>Digestive Track</th>
<th>Medicine Effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RBC</td>
<td>WBC</td>
<td>Murmur</td>
<td>Arrhythmia</td>
</tr>
<tr>
<td>A</td>
<td>3.9</td>
<td>10.0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>2.5</td>
<td>11</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>4.3</td>
<td>5.3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

- Cryptography in the RAM Computation Model

6/18/16 • 5
Vertically Partitioned Database

<table>
<thead>
<tr>
<th>Patient</th>
<th>Blood Count</th>
<th>Heart Conditions</th>
<th>Digestive Track</th>
<th>Medicine Effectiveness</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RBC</td>
<td>WBC</td>
<td>Murmur</td>
<td>Arrhythmia</td>
</tr>
<tr>
<td>A</td>
<td>3.9</td>
<td>10.0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>5.0</td>
<td>4.5</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>2.5</td>
<td>11</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>4.3</td>
<td>5.3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

- Different columns belong to different parties
  - E.g., different specialized hospitals have different parts of the information for all patients

Cryptography in the RAM Computation Model
Ridge Regression

- Computing linear model on inputs \((x_1, y_1), \ldots, (x_n, y_n)\)
  \(x_i \in \mathbb{R}^d, y_i \in \mathbb{R}\)

- Optimization formulation

\[
\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (y^i - \langle \theta, x^i \rangle)^2 + \lambda \|\theta\|^2
\]

- Linear System Formulation

\[
x = \begin{bmatrix}
  \vdots & x^1 & \vdots \\
  \vdots & \vdots & \vdots \\
  \vdots & x^n & \vdots \\
\end{bmatrix}
\quad y = \begin{bmatrix}
  y^1 \\
  \vdots \\
  y^n \\
\end{bmatrix}
\]

\[
\left(\frac{1}{n}X^TX + \lambda I\right)\theta = X^T y
\]

Positive definite
Contributions

- Secure computation for ridge regression for vertically partitioned database
  - Two phase protocol:
    - **Phase1** – compute \( A = \frac{1}{n} X^T X + \lambda I \quad b = X^T Y \)
      - Output is additively shared between two parties
    - **Phase2** – solve \( A\theta = b \) where \( A \) and \( b \) are shared between two parties
  - Two party and multiparty protocol for Phase1
    - Two party inner product computation
  - Three algorithms for Phase2:
    - Cholesky, LDLT, Conjugate Gradient Descent (CGD)
- Implementation and evaluation
Phase 1

- Compute $A = \frac{1}{n}X^TX + \lambda I \quad b = X^TY$
  - The output is additively shared between two parties

- Each entry of $A$ is a dot product of the vectors held by two different parties
  - In the multi-party case too
  - Two party computation of dot product
Phase 1

- Architecture – inspired by [NWIJBT13]
  - Two additional semi-honest, non-colluding parties:
    1. Crypto Service Provider (CSP) – generates parameters
    2. Evaluator – helps for the evaluation of the protocols, has no inputs
- Our setting
Phase 1

Two Parties

Many Parties

Cryptography in the RAM Computation Model

Phase 1

Two Parties

Many Parties

Cryptography in the RAM Computation Model

Phase 1

Two Parties

Many Parties

Cryptography in the RAM Computation Model
Phase 2

• Two party protocol
  o **Inputs:** additive shares of matrix $A$ and vector $b$
  o **Outputs:** additive shares of $\theta$ such that $A\theta = b$

• Gabled circuits computation

• Solutions algorithms
  o Two **exact** algorithms: Cholesky, LDLT
  o One **approximation** algorithm: Conjugate Gradient Descent (CGD)

• [NWIJBT13] implements Cholesky
Cholesky decomposition for positive definite matrices
- $A = LL^T$
- $L$: $d \times d$ lower triangular matrix

Idea: solve $LL^T \theta = b$
- $L\theta' = b$
- $L^T \theta = \theta'$

Complexity: $O(d^3)$ floating point operations

Two properties:
- Data-agnostic – no pivoting
- Numerically robust – suitable for finite precision implementations
LDLT

- Variant of Cholesky decomposition
  - $A = LDL^T$
  - $L$ – lower triangular
  - $D$ – diagonal, non-negative entries
- Idea: solve $LDL^T \theta = b$
  - $L\theta'' = b$
  - $D\theta' = \theta''$
  - $L^T\theta = \theta'$
- Complexity: $O(d^3)$
  - No square root
  - Additional substitution phase
- Same properties

---

Algorithm 2: LDLT

Input: $A$, $b$
Output: Solution $\theta$ to $A\theta = b$

for $j = 1 \ldots d$ do
  $D_j = A_{jj} - \sum_{k=1}^{j-1} L_{jk}^2 D_k$
  for $i = j + 1 \ldots d$ do
    $L_{ij} = (A_{ij} - \sum_{k=1}^{i-1} L_{ik} L_{jk} D_k)/D_j$
  end
end

$\theta'_1 = b_1$

for $i = 2 \ldots d$ do
  $\theta'_i = b_i - \sum_{j=1}^{i-1} L_{ij} \theta'_j$
end

$\theta'_d = \theta'_d/D_d$

for $i = d - 1 \ldots 1$ do
  $\theta_i = \theta'_i/D_i - \sum_{j=i+1}^d L_{ji} \theta_j$
end
Approximate solution

Solving $A\theta = b$ by solving the optimization

$$\text{argmin}_\theta \|A\theta - b\|^2$$

Iterative solutions approach based on conjugate gradients

Complexity
- Until convergence $O(d^3)$
- Early termination $O(d^2)$ per iteration

Error: $\varepsilon$ after $O(\sqrt{\kappa \log 1/\varepsilon})$ iterations
- $\kappa$ - condition number

Algorithm 3: Conjugate Gradient Descent

<table>
<thead>
<tr>
<th>Input</th>
<th>$A$, $b$, number of iterations $k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output:</td>
<td>Approximate solution $\theta_k$ to $A\theta = b$</td>
</tr>
</tbody>
</table>

Let $\theta_0 = 0$ and $p_0 = r_0 = A\theta_0 - b$

for $t = 0 \ldots k$ do

\[\theta_{t+1} = \theta_t - \frac{(r_t, r_t)}{p_t^T A p_t} p_t\]

\[r_{t+1} = A\theta_{t+1} - b\]

\[p_{t+1} = r_{t+1} + \frac{(r_{t+1}, r_{t+1})}{(r_t, r_t)} p_t\]

end
Fixed-Point Arithmetic

\[
\mathbb{R} \xleftrightarrow{\phi_\delta} \mathbb{Z} \xleftrightarrow{\varphi_q} \mathbb{Z}_q
\]

- \( \phi_\delta(r) = [r/\delta] ; \ \tilde{\phi}_\delta(z) = z\delta, |r - \tilde{\phi}_\delta(\phi_\delta(r))| \leq \delta \)
- \( \varphi(z) = z \) if \( z \geq 0 \); \( \varphi(z) = z + q \) if \( z < 0 \)
- \( \tilde{\varphi}(u) = u \) if \( 0 \leq u \leq q/2 \); \( \tilde{\varphi}(u) = u - q \) if \( q/2 < u \leq q - 1 \)

- **Phase 1**: n-dim vectors with entries of size R
  - Error: \( n(2R\delta + \delta^2) \)
  - Normalize \( R \leq 1/\sqrt{n} \Rightarrow \) error \( \epsilon \) with \( \delta = \epsilon / 2\sqrt{n} \) and \( q = 8n/\epsilon^2 \)
    - \( O(\log(n/\epsilon)) \) bit representation

- **Phase 2 – experiments**
  - \( q = 2^{32} \) (4 bits integer part, 1 bit sign) \( \Rightarrow \) \( \delta = 2^{-27} \)
  - \( q = 2^{64} \) (4 bits integer part, 1 bit sign) \( \Rightarrow \) \( \delta = 2^{-59} \)
Implementation and Evaluation

- Obliv-C
  - Most recent optimizations: Free XOR, Garbled Row Reduction, Fixed Key Block Ciphers, Half Gates

- Fixed point arithmetic on top of Obliv-C
  - Algorithms: multiplication (Karatsuba-Comba), division (Knuth’s algorithm D), square root (Newton’s method)
  - 32 bits: 4 bits (integral part) + 28 bit (fractional part)

- Synthetic datasets (vs real datasets)
  - Generated with correct $\lambda$ parameter – sample from $d$-dimensional Gaussian distribution
  - Tuning $\lambda$ privately is hard question – incorrect $\lambda$ makes the optimization too easy or too difficult

- Amazon EC2 C4 (15GB RAM, 8 CPU cores)
Phase 1

The first phase (Figure 3) was executed with different input dimensions \( d \). It shows that for all values of \( d \), with increasing number of parties, the amount of work taken by the parties decreases. The absolute times are for the TI increases, while the average computation time for the parties decreases.

![Bar chart showing normalized computation time for different numbers of parties and input dimensions.](chart.png)

### Table 1: Normalized Computation Time

<table>
<thead>
<tr>
<th>d</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.17</td>
<td>0.033</td>
<td>0.22</td>
</tr>
<tr>
<td>100</td>
<td>19</td>
<td>1.7</td>
<td>26</td>
</tr>
<tr>
<td>500</td>
<td>109</td>
<td>146</td>
<td>149</td>
</tr>
</tbody>
</table>

Database partitioned equally among parties:

- Column1: \((2000, 20)\)
- Column2: \((100000, 100)\)
- Column3: \((500000, 500)\)

Cryptography in the RAM Computation Model
Phase 2

![Graph showing circuit size vs. size d](image)

<table>
<thead>
<tr>
<th>CGD-15</th>
<th>10</th>
<th>20</th>
<th>50</th>
<th>100</th>
<th>500</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.47668</td>
<td>7.40373</td>
<td>37.8064</td>
<td>140.652</td>
<td>4876.59</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cholesky</th>
<th>10</th>
<th>20</th>
<th>50</th>
<th>100</th>
<th>200</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.403127</td>
<td>1.69357</td>
<td>20.4527</td>
<td>162.049</td>
<td>1157.95</td>
<td></td>
</tr>
</tbody>
</table>
Phase 2

Convergence of CGD

Fixed vs Floating Point
Conclusions

• Machine learning algorithms – target for MPC
• Ridge regression
  o Vertically partitioned datasets
• Tailored protocol for Phase1
• Two party computation for solving systems of linear equations for Phase2
  o Exact (Cholesky, LDLT) and approximation (CGD) algorithms
  o Approximation: more efficient with sufficient precision
• Next steps – classification (logistic regression)
Thank You!