Privacy Preserving Distributed Maximum Likelihood Estimation

Xiaodong Lin

University of Cincinnati

Joint work with Alan F. Karr

Outline

- Privacy preserving MLE for horizontally partitioned data
- Private preserving MLE for vertically partitioned data
- Secure multi-party protocol for function evaluation
- “Opt out" strategy
- Future work

Distributed Maximum Likelihood Estimation

- Data $\mathbf{x^n} = \{x_1, \cdots, x_n\}$ generated from $f(x;\theta)$
- The data are distributed across different agencies
  1. Horizontally partitioned
  2. Vertically partitioned
- The MLE is

$$\hat{\theta} = arg \max_{\theta} l(\theta|\mathbf{x^n})$$

- Goal: compute $\hat{\theta}$ without sharing data between agencies

Horizontally partitioned, exponential family

- Exponential family $f(x) = b(x)exp\{a(\theta)^T t(x) - c(\theta)\}$
- Log likelihood

$$l(\theta; \mathbf{x^n}) = \sum_{i=1}^{n} \log b(x_i) + \sum_{i=1}^{n} \{a(\theta)^T t(x_i) - c(\theta)\}$$

- The MLE is

$$\hat{\theta} = \arg\max_{\theta} \;\; a(\theta)^T \sum_{i=1}^{n} t(x_i) - nc(\theta)$$

- Secure summation of $\sum_{i=1}^{n} t(x_i)$

Horizontally partitioned, Newton Raphson

- Given the estimates $\theta^{(s-1)}$ from the previous step, new estimate is

$$\theta^{(s)} = \theta^{(s-1)} - (D^2 l(\mathbf{x}^{\mathbf{n}}; \theta^{(s-1)}))^{-1} \nabla l(\mathbf{x}^{\mathbf{n}}; \theta^{(s-1)}),$$

where $D^2 l()$ is the Hassian and $\nabla l()$ is the gradient

- Assume $\theta = \{\theta_1, \cdots, \theta_k\}$,

$$\nabla l(\mathbf{x}^{\mathbf{n}}; \theta^{(s-1)}) = \left(\sum_{i=1}^{n} \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_1}}{f(x_i;\theta)}, \cdots, \sum_{i=1}^{n} \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_k}}{f(x_i;\theta)} \right)'_{\theta^{(s-1)}}$$

Horizontally partitioned, Newton Raphson

- Locally, we can compute L_j, $1 \leq j \leq m$, where

$$L_j = \left(\sum_{i=1}^{m_j} \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_1}}{f(x_i;\theta)}, \cdots, \sum_{i=1}^{m_j} \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_k}}{f(x_i;\theta)} \right)'_{\theta^{(s-1)}}$$

- Similarly we can compute

$$H_j(h,l) = \sum_{i=1}^{m_j} \left(\frac{\frac{\partial^2 f(x_i;\theta)}{\partial \theta_h \partial \theta_l}}{f(x_i;\theta)} - \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_h} \frac{\partial f(x_i;\theta)}{\partial \theta_l}}{f^2(x_i;\theta)} \right)_{\theta^{(s-1)}}$$

- The iteration step becomes

$$\theta^{(s)} = \theta^{(s-1)} - (\sum_{j=1}^{m} H_j)^{-1} (\sum_{j=1}^{m} L_j)$$

Horizontally partitioned, Newton Raphson

- H_j and L_j can be computed locally at each agency
- If $m > 2$, use secure summation to compute and share $\sum_{j=1}^{m} H_j$ and $\sum_{j=1}^{m} L_j$
- Potential drawbacks
  1. m has to be greater than 2
  2. Share more than necessary
- Compute $(\sum_{j=1}^{m} H_j)^{-1}(\sum_{j=1}^{m} L_j)$ directly

Horizontally partitioned, direct computation

- Without loss of generaility, assume $m = 2$
- Note that when $m = 2$, secure summation can't be applied
- Our goal: Compute $(H_1 + H_2)^{-1}(L_1 + L_2)$ securely
- Approach: Solving linear equation system
- Denote $X = (H_1 + H_2)^{-1}(L_1 + L_2)$, the problem is equivalent to solve

$$(H_1 + H_2)X = (L_1 + L_2)$$

Horizontally partitioned, direct computation

- Assume two agencies A and B
- A and B generate $k \times k$ matrix M_1 and M_2 respectively, both with rank $k/2$
- A sends M_1 to B. B computes $M_1 H_2$ and $M_1 L_2$, sends them to A
- A can produce the linear equation system

$$M_1(H_1 + H_2)X = M_1(L_1 + L_2)$$

- Symmetrically, B can produce

$$M_2(H_1 + H_2)X = M_2(L_1 + L_2)$$

Horizontally partitioned, direct computation

- Sharing the two linear equation systems directly will reveal L_1 and L_2
- Solution: A and B generate full rank matrices T_1 and T_2 respectively
- Combine the following two linear equation systems to solve for X

$$T_1 M_1 (H_1 + H_2) X = T_1 M_1 (L_1 + L_2)$$

$$T_2 M_2 (H_1 + H_2) X = T_2 M_2 (L_1 + L_2)$$

Security analysis and discussion

- Agency A sent to B: $M_2H_1, M_2L_1, T_1M_1(H_1+H_2)$ and $T_1M_1(L_1+L_2)$
- A can check the rank of M_2. When $K>2$, H_1 and L_1 are not revealed
- Sharing of $T_1M_1(H_1+H_2)$ reveals T_1H_1 to B, but not H_1
- Protocol is symmertric
- Protocol works for $m=2$

Vertically partitioned, independent variable

- Assume $\mathbf{x^n} = \{x_1, \cdots, x_n\}$, where $x_i = (x_i^1, \cdots, x_i^p)$. Each agency owns portion of the variables for all x_i
- Assume $f(x_i, \theta) = \Pi_{s=1}^{p} f_s(x_i^s; \theta)$
- Log likelihood

$$l = \sum_{s=1}^{p} \left[\sum_{i=1}^{n} \log f_s(x_i^s; \theta) \right]$$

- Compute locally at each agency and use secure summation or the direct computation protocol

Vertically partitioned, exponential family

- Exponential family $f(x) = b(x) exp\{a(\theta)^T t(x) - c(\theta)\}$
- The MLE is

$$\hat{\theta} = \arg\max_{\theta} \quad a(\theta)^T \sum_{i=1}^{n} t(x_i) - nc(\theta)$$

- Two agencies, A and B. A holds $(x_{1,i}, \cdots, x_{k,i})$, and B holds $(x_{k+1,i}, \cdots, x_{p,i})$, $1 \leq i \leq n$
- Need a protocol to compute $\sum_{i=1}^{n} t(x_{1,i}, \cdots, x_{k,i}; x_{k+1,i}, \cdots, x_{p,i})$ securely

Vertically partitioned, secure two party computation

- Protocol to compute $\sum_{i=1}^{n} t(x_{1,i}, x_{2,i})$ securely
- Step one. Agency A generate a vector of length s, among which the kth item $x_{1,i}^{k} = x_{1,i}$. The other $s-1$ items are random numbers
- Step two. A sends this vector to B, B computes $t^1 = t(x_{1,i}^1, x_{2,i}), \cdots, t^s = t(x_{1,i}^s, x_{2,i})$. B generates a random number ϵ_i and computes $g_i^1 = t^1 - \epsilon_i, \cdots, g_i^s = t^2 - \epsilon_i$

Vertically partitioned, secure two party computation

- Step three. Agency A obtains g_i^k using 1 out of s oblivious transfer

- Step four. Agency A has $\sum_{i=1}^{n} g_i^k$ and Agency B has $\sum_{i=1}^{n} \epsilon_i$. Their sum gives $\sum_{i=1}^{n} t(x_{1,i}, x_{2,i})$

Vertically partitioned, secure two party computation

- Agency A obtains g_i^k. Since Agency does not know ϵ_i, value of $x_{2,i}$ is not revealed
- The quantities $\sum_{i=1}^{n} g_i^k$ and $\sum_{i=1}^{n} \epsilon_i$ are shared, but not the individual values
- Non symmetric due to 1 out of N oblivious transfer
- Communication cost $n * s + n * L(s)$. $L(s)$ is the communication cost for 1 out of s oblivious transfer

Vertically partitioned, Newton Raphson

- The gradient vector and Hessian matrix are

$$L = \left(\sum_{i=1}^{n} \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_1}}{f(x_i;\theta)}, \ldots, \sum_{i=1}^{n} \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_k}}{f(x_i;\theta)} \right)_{\theta^{(s-1)}}$$

and

$$H = \sum_{i=1}^{n} \left(\frac{\frac{\partial^2 f(x_i;\theta)}{\partial \theta_h \partial \theta_l}}{f(x_i;\theta)} - \frac{\frac{\partial f(x_i;\theta)}{\partial \theta_h} \frac{\partial f(x_i;\theta)}{\partial \theta_l}}{f^2(x_i;\theta)} \right)_{\theta^{(s-1)}}$$

- Assume the functional form of H and L are shared, parameters can be updated using the last protocol

“Opt out" strategies

- Utility and security considerations will cause agencies to opt out
- Size of dataset, numbers of variables
- Observed Fisher Information matrix

$$(\mathbf{J}(\theta))_{qh} = -\sum_{i=1}^{n} \frac{\partial^2}{\partial\theta_q \partial\theta_h} \log f(x_i; \theta).$$

  Compare local J with the global J
- Other utility and risk measures

Conclusion

- Privacy Preserving MLE for horizontally partitioned data using secure summation
- Privacy Preserving MLE for horizontally partitioned data using direct computation
- Privacy Preserving MLE for vertically partitioned data using secure function evalution
- Opt out strategies

Future work

- Private information propagation through iterations
- Constrained MLE

$$\hat{\theta} = \arg\max l(\theta; \mathbf{x}^{\mathbf{n}}) \ \ s.t. \ \ C_j(\theta) \ 1 \leq j \leq m,$$

  where $C_j(\theta)$ are the parameter constraints each agency follows and can not be shared
- General constrained optimization problems with privacy assurance
- Connection between privacy preserving distributed computing and SDL