SecureBoost: A Lossless Federated Learning Framework

Kewei Cheng[1], **Tao Fan[2]**, Yilun Jin[3], Yang Liu[2], Tianjian Chen[2] and Qiang Yang[4]

[1]University of California, Los Angeles

[2]Webank

[3]Peking University

[4]Hong Kong University of Science and Technology

PART 01

BackGround of SecureBoost

Challenges for AI Industry: Data Privacy and Confidentiality

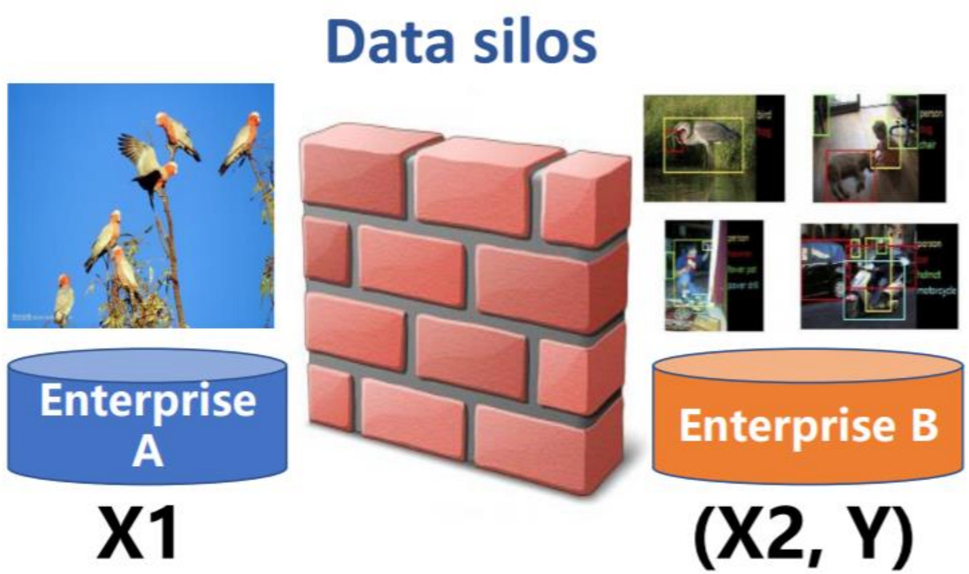
French regulator fines Google $57 million for GDPR violations

Share on Facebook | Share on Twitter | +

Google hasn't transparently implemented GDPR rules, French regulator claims.

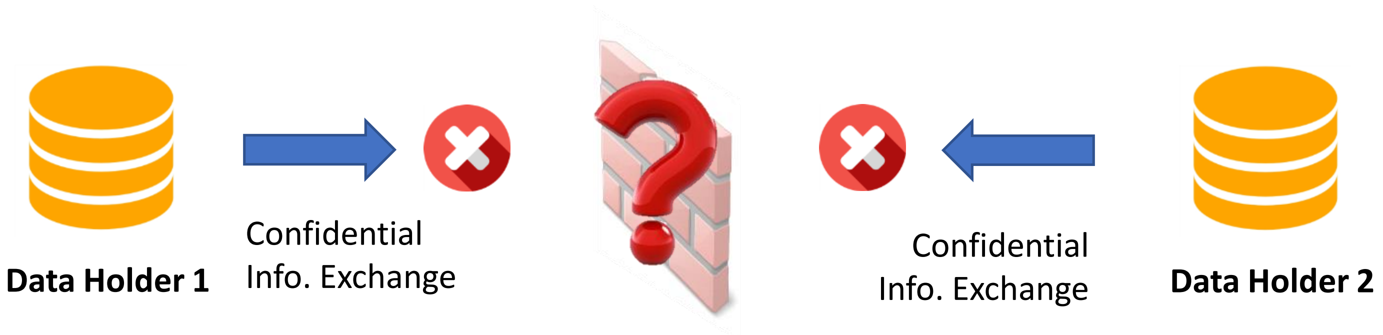
Society is increasingly concerned with the unlawful use and exploitation of personal data

Challenges for AI Industry: Data Privacy and Confidentiality



- Many data owners do not have a sufficient amount of data to build high-quality models
- Different organizations have to collaborate

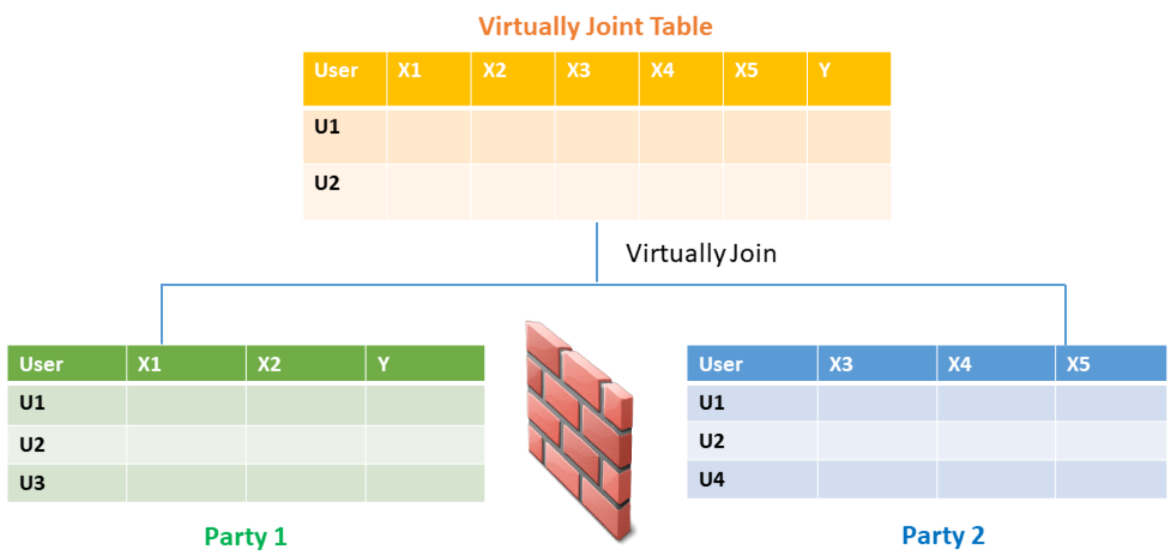
Challenges for AI Industry: Data Privacy and Confidentiality



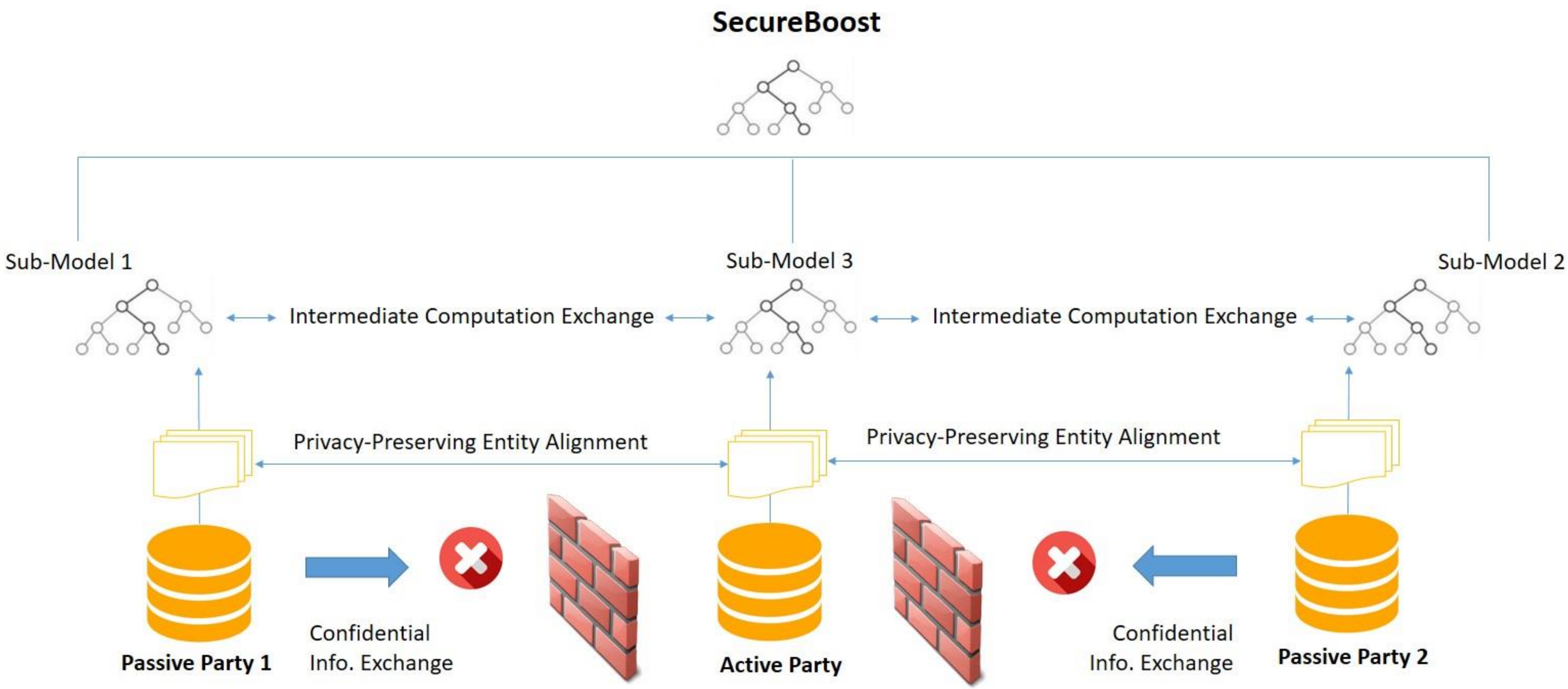
Problem Statement

Given: (1) vertically partitioned data; (2) only one data owner holds the label

Goal: Learn a shared model without leaking any information



Framework


SecureBoost
Sub-Model 1
Sub-Model 3
Sub-Model 2
Intermediate Computation Exchange
Intermediate Computation Exchange
Privacy-Preserving Entity Alignment
Privacy-Preserving Entity Alignment
Passive Party 1
Confidential
Info. Exchange
Active Party
Confidential
Info. Exchange
Passive Party 2

PART 02

What’s SecureBoost

Review of XGBoost

- Objective function

$$\sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right] + \Omega(f_t)$$

  - where $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}), \quad h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$
- Define the instance set in leaf j as $I_j = \{i | q(x_i) = j\}$
  - Regroup the objective by leaf

$$\begin{aligned} Obj^{(t)} &\simeq \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right] + \Omega(f_t) \\ &= \sum_{i=1}^{n} \left[g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2\right] + \gamma T + \lambda \frac{1}{2} \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T} \left[(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i + \lambda) w_j^2\right] + \gamma T \end{aligned}$$

Review of XGBoost

- Greedy learning of the tree
  - Start from tree with depth 0
  - For each leaf node of the tree, try to add a split. The change of objective after adding the split is

$$Gain = \frac{1}{2}\Big[\underbrace{\frac{G_L^2}{H_L+\lambda}}_{\text{the score of left child}} + \underbrace{\frac{G_R^2}{H_R+\lambda}}_{\text{the score of right child}} - \underbrace{\frac{(G_L+G_R)^2}{H_L+H_R+\lambda}}_{\text{the score of if we do not split}}\Big] - \underbrace{\gamma}_{\text{The complexity cost by introducing additional leaf}}$$

  - where $G_j = \sum_{i \in I_j} g_i \quad H_j = \sum_{i \in I_j} h_i$

Review of XGBoost

- An Algorithm for Split Finding

Algorithm 1 Greedy Split-find Algorithm

M: # features, N: # instances, K: # split candidates

for $m = 1$ to M **do**
  generate K split candidates $S_m = \{s_{m1}, s_{m2}, ..., s_{mk}\}$
end for
for $m = 1$ to M **do**
  loop N instances to generate gradient histogram with K bins
  $G_{mk} = \sum g_i$ where $s_{mk-1} < x_{im} < s_{mk}$
  $H_{mk} = \sum h_i$ where $s_{mk-1} < x_{im} < s_{mk}$
end for
$gain_{max} = 0$, $G = \sum_{i=1}^{N} g_i$, $H = \sum_{i=1}^{N} h_i$
for $m = 1$ to M **do**
  $G_L = 0$, $H_L = 0$
  for $k = 1$ to K **do**
    $G_L = G_L + G_{mk}$, $H_L = H_L + H_{mk}$
    $G_R = G - G_L$, $H_R = H - H_L$
    $gain_{max} = max(gain_{max}, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda})$
  end for
end for
Output the split with max gain

Recap: XGBoost Algorithm

- Add a new tree in each iteration
- Beginning of each iteration, calculate

$$g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}), \quad h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$$

- Use the statistics to greedily grow a tree $f_t(x)$

$$Obj = -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T$$

- Add $f_t(x)$ to the model $\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i)$
  - Usually, instead we do $y^{(t)} = y^{(t-1)} + \epsilon f_t(x_i)$
  - ϵ is called step-size or shrinkage, usually set around 0.1
  - This means we do not do full optimization in each step and reserve chance for future rounds, it helps prevent overfitting

Federated Learning for XGBoost

Party 1 (Passive Party)

Example	Bill Payment	Education
X1	3102	2
X2	17250	3
X3	14027	2
X4	6787	1
X5	280	1

Party 2 (Active Party)

Example	Age	Gender	Marriage	Label
X1	20	1	0	0
X2	30	1	1	1
X3	35	0	1	1
X4	48	0	1	2
X5	10	1	0	3

Party 3 (Passive Party)

Example	Amount of given credit
X1	5000
X2	300000
X3	250000
X4	300000
X5	200

- Gain only depend on the g_i and h_i

$$Gain = \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}$$

Federated Learning for XGBoost

Party 1 (Passive Party)

Example	Bill Payment	Education
X1	3102	2
X2	17250	3
X3	14027	2
X4	6787	1
X5	280	1

Party 2 (Active Party)

Example	Age	Gender	Marriage	Label
X1	20	1	0	0
X2	30	1	1	1
X3	35	0	1	1
X4	48	0	1	2
X5	10	1	0	3

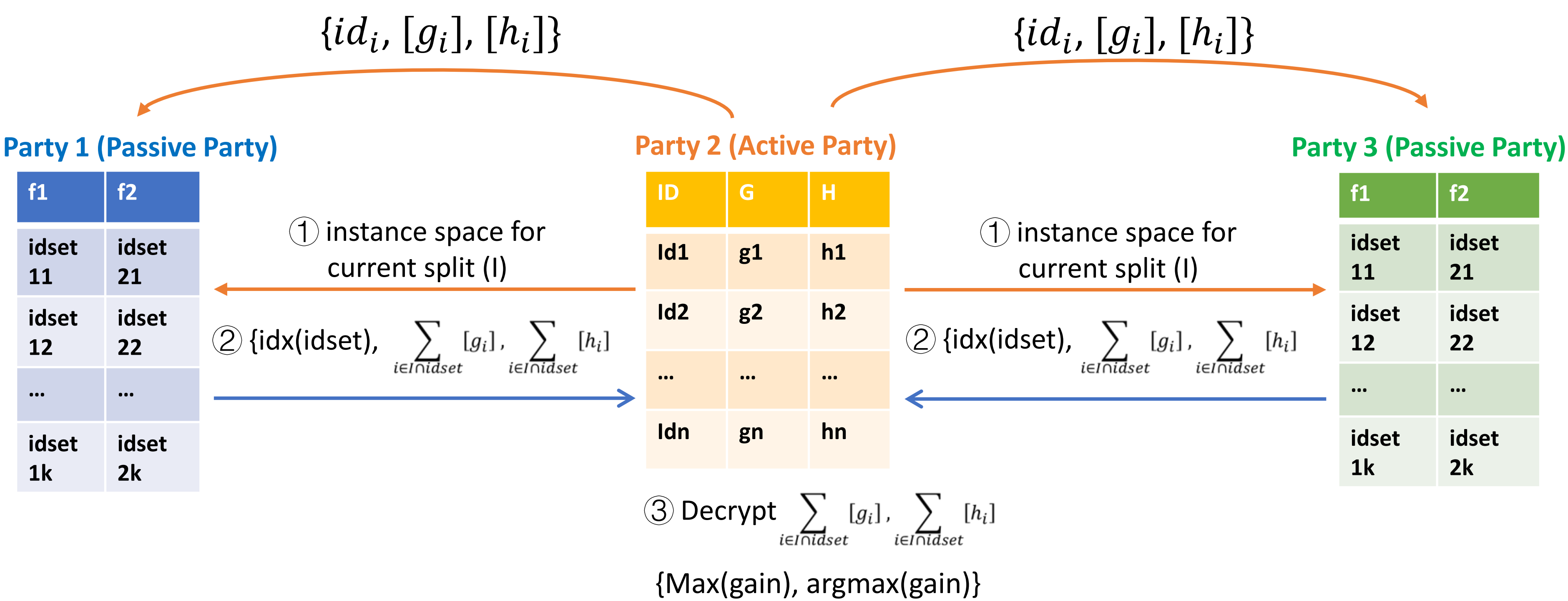
Party 3 (Passive Party)

Example	Amount of given credit
X1	5000
X2	300000
X3	250000
X4	300000
X5	200

- The class label is needed for the calculation of g_i and h_i
- Only active party holds label
- How to calculate Gain?

Federated Algorithm for Split Finding

$[g_i]$: hormomorphic encrypted g_i
$[h_i]$: hormomorphic encrypted h_i



Learned SecureBoost

Party 1 (Passive Party)

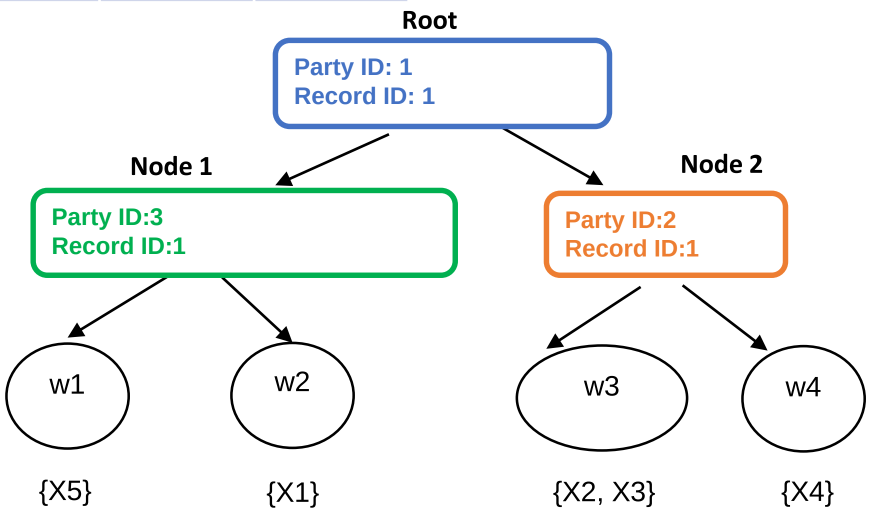
Example	BIll Payment	Education
X1	3102	2
X2	17250	3
X3	14027	2
X4	6787	1
X5	280	1

Party 2 (Active Party)

Example	Age	Gender	Marriage	Label
X1	20	1	0	0
X2	30	1	1	1
X3	35	0	1	1
X4	48	0	1	2
X5	10	1	0	3

Party 3 (Passive Party)

Example	Amount of given credit
X1	5000
X2	300000
X3	250000
X4	300000
X5	200



Lookup table

Party 1:

Record ID	Feature	threshold value
1	Bill Payment	5000

Party 2:

Record ID	Feature	threshold value
1	Age	40

Party 3:

Record ID	Feature	threshold value
1	Amount of given credit	800

Federated Inference

Training Set

Party 1 (Passive Party)

Example	Bill Payment	Education
X1	3102	2
X2	17250	3
X3	14027	2
X4	6787	1
X5	280	1

Party 2 (Active Party)

Example	Age	Gender	Marriage	Label
X1	20	1	0	0
X2	30	1	1	1
X3	35	0	1	1
X4	48	0	1	2
X5	10	1	0	3

Party 3 (Passive Party)

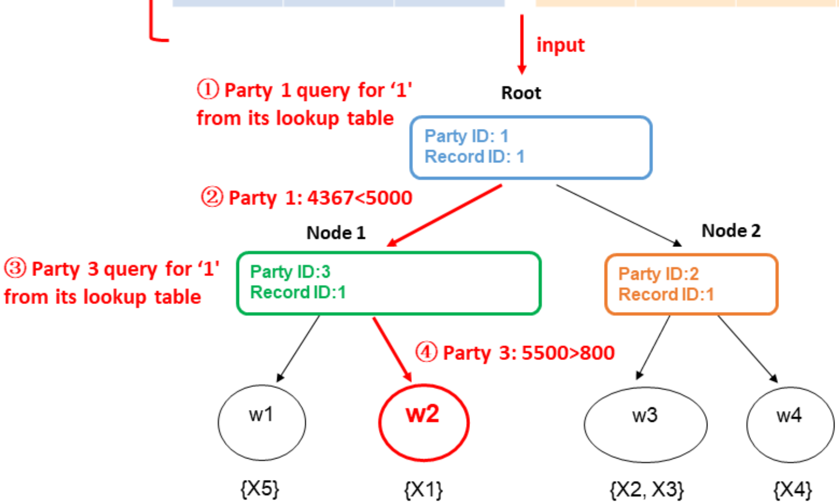
Example	Amount of given credit
X1	5000
X2	300000
X3	250000
X4	300000
X5	200

Predict

Example	Bill Payment	Education
X6	4367	2

Example	Age	Gender	Marriage	Label
X6	28	1	0	0

Example	Amount of given credit
X6	5500



Lookup table

Party 1:

Record ID	Feature	threshold value
1	Bill Payment	5000

Party 2:

Record ID	Feature	threshold value
1	Age	40

Party 3:

Record ID	Feature	threshold value
1	Amount of given credit	800

Advantages

- No exposure of raw data
- Property of Lossless

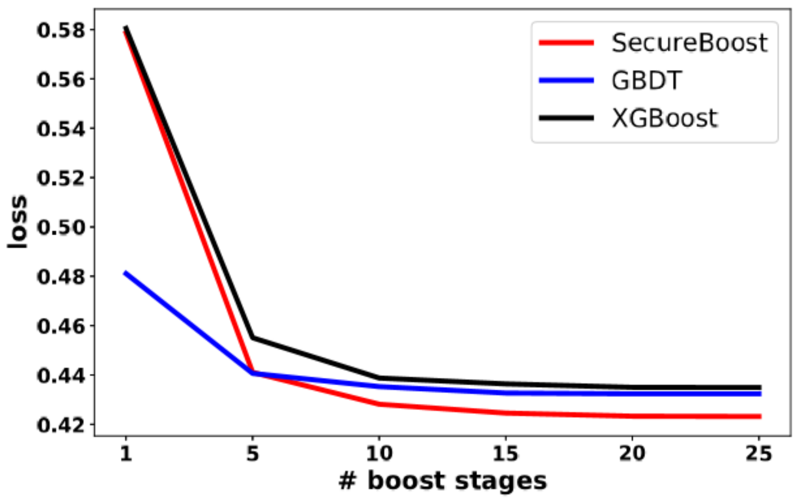
The source code of SecureBoost can be seen in FATE (An Industrial Level Federated Learning Framework: https://github.com/WeBankFinTech/FATE)

PART

03

Experiment

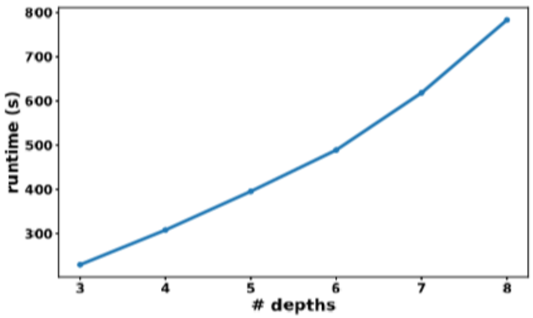
Property of Lossless



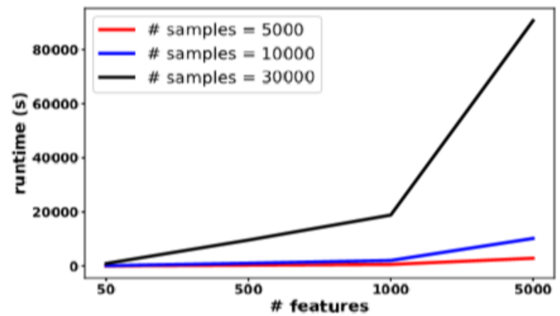
(b) Test Error

- Our proposed SecureBoost framework perform equally well as baseline methods.
- We also give a theoretical analysis for lossless property.

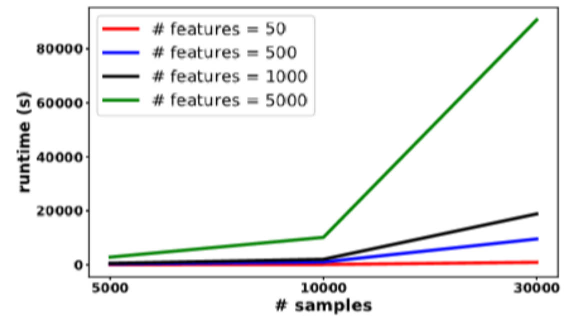
Scalability



(a)Runtime w.r.t. maximum depth of individual tree



(b)Runtime w.r.t. feature size



(c)Runtime w.r.t. sample size

- With the increase of the maximum depth of each individual tree, the runtime increases almost linearly.
- Sample and feature numbers contribute equally to running time.

Thanks