Scalability Issues and the Potential for Encrypted Machine Learning

Louis J. M. Aslett, Pedro M. Esperança and Chris C. Holmes

Department of Statistics, University of Oxford

2nd UCL Workshop on the Theory of Big Data
6th January 2016

i-like.org.uk

Motivation

Security in big data applications is a growing concern:

- computing in a ‘hostile’ environment (e.g. cloud computing);

Motivation

Security in big data applications is a growing concern:

- computing in a ‘hostile’ environment (e.g. cloud computing);
- donation of sensitive/personal data (e.g. medical/genetic studies);

Motivation

Security in big data applications is a growing concern:

- computing in a 'hostile' environment (e.g. cloud computing);
- donation of sensitive/personal data (e.g. medical/genetic studies);
- complex models on constrained devices (e.g. smart watches)

Motivation

Security in big data applications is a growing concern:

- computing in a ‘hostile’ environment (e.g. cloud computing);
- donation of sensitive/personal data (e.g. medical/genetic studies);
- complex models on constrained devices (e.g. smart watches)
- running confidential algorithms on confidential data (e.g. engineering reliability)

Encryption the solution?

Encryption can provide security guarantees …

Easy

$$\texttt{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \texttt{Dec}(k_s, c) = m$$

Hard without k_s

… but is typically 'brittle'.

Encryption the solution?

Encryption can provide security guarantees …

$$\mathtt{Enc}(k_p, m) \overset{\text{Easy}}{\underset{\text{Hard without } k_s}{\rightleftharpoons}} c \qquad\qquad \mathtt{Dec}(k_s, c) = m$$

… but is typically 'brittle'.

Rivest et al. (1978) proposed encryption schemes capable of arbitrary addition and multiplication may be possible. Gentry (2009) showed first **fully homomorphic encryption** scheme.

Encryption the solution?

Encryption can provide security guarantees …

Easy

$$\texttt{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \texttt{Dec}(k_s, c) = m$$

Hard without k_s

… but is typically 'brittle'.

Rivest et al. (1978) proposed encryption schemes capable of arbitrary addition and multiplication may be possible. Gentry (2009) showed first **fully homomorphic encryption** scheme.

$$m_1 \qquad m_2 \xrightarrow{+} m_1 + m_2$$

Encryption the solution?

Encryption can provide security guarantees …

Easy

$$\mathsf{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \mathtt{Dec}(k_s, c) = m$$

Hard without k_s

… but is typically ‘brittle’.

Rivest et al. (1978) proposed encryption schemes capable of arbitrary addition and multiplication may be possible. Gentry (2009) showed first **fully homomorphic encryption** scheme.

$$\begin{tikzcd}
m_1 \arrow[d, "\mathsf{Enc}(k_p, \cdot)"] & m_2 \arrow[r, "+"] \arrow[d] & m_1 + m_2 \\
c_1 & c_2 &
\end{tikzcd}$$

Encryption the solution?

Encryption can provide security guarantees ...

Easy

$$\mathtt{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \mathtt{Dec}(k_s, c) = m$$

Hard without k_s

... but is typically 'brittle'.

Rivest et al. (1978) proposed encryption schemes capable of arbitrary addition and multiplication may be possible. Gentry (2009) showed first **fully homomorphic encryption** scheme.

$$\begin{array}{ccccc}
m_1 & & m_2 & \xrightarrow{+} & m_1 + m_2 \\
\downarrow & \mathtt{Enc}(k_p, \cdot) & \downarrow & & \uparrow \mathtt{Dec}(k_s, \cdot) \\
c_1 & & c_2 & \xrightarrow{\oplus} & c_1 \oplus c_2
\end{array}$$

Encryption the solution?

Encryption can provide security guarantees …

Easy

$$\mathtt{Enc}(k_p, m) \rightleftharpoons c \qquad\qquad \mathtt{Dec}(k_s, c) = m$$

Hard without k_s

… but is typically 'brittle'.

Rivest et al. (1978) proposed encryption schemes capable of arbitrary addition and multiplication may be possible. Gentry (2009) showed first **fully homomorphic encryption** scheme.

$$\begin{tikzcd}
m_1 \arrow[d, "\mathtt{Enc}(k_p, \cdot)"] & m_2 \arrow[d] & & m_1 + m_2 \\
c_1 & c_2 \arrow[rr, "\oplus"] & & c_1 \oplus c_2 \arrow[u, "\mathtt{Dec}(k_s, \cdot)"']
\end{tikzcd}$$

Limitations of homomorphic encryption

1. Message space (what we can encrypt)
   - Commonly only easy to encrypt binary/integers/polynomials
2. Cipher text size (the result of encryption)
   - Present schemes all inflate the size of data substantially (e.g. 1MB $\rightarrow$ 16.4GB)
3. Computational cost (computing without decrypting)
   - 1000's additions per sec
   - $\approx$ 50 multiplications per sec
4. Division and comparison operations (equality/inequality checks)
   - Not possible in current schemes!
5. Depth of operations
   - After a certain depth of multiplications, need to 'refresh' cipher text: hugely time consuming, so avoid!

Statistics & Machine Learning Encrypted?

Lots of constraints! Are traditional statistics and machine learning techniques out of reach to run on encrypted data? We've looked at a semi-parametric naïve Bayes and a variant of random forests.

Statistics & Machine Learning Encrypted?

Lots of constraints! Are traditional statistics and machine learning techniques out of reach to run on encrypted data? We've looked at a semi-parametric naïve Bayes and a variant of random forests.

So, want to build a random forest on encrypted data … but,

- No comparisons possible to evaluate splits
- No max possible to find highest class vote
- No division possible to do average votes
- …

Thus random forests (and other methods) need to be tailored for encrypted computation. This is where statistics and machine learning community can get involved!

Completely Random Forests (CRFs) — Data encoding

1

$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0 \mid 0 \mid 0 \mid 1 \mid 0}$$

Completely Random Forests (CRFs) — Data encoding

1.

$$b_0 := -\infty \qquad\qquad b_B := \infty$$

$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0}\boxed{0}\overbrace{\boxed{0}}^{b_2 < x_{ij} \leq b_3}\underbrace{\boxed{1}}_{b_3 < x_{ij} \leq b_4}\boxed{0}$$

Completely Random Forests (CRFs) — Data encoding

1.

$$b_0 := -\infty \qquad\qquad b_B := \infty$$

$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0}\,\boxed{0}\,\overbrace{\boxed{0}}^{b_2 < x_{ij} \le b_3}\,\underbrace{\boxed{1}}_{b_3 < x_{ij} \le b_4}\,\boxed{0} = \{x_{ijk} : k = 1, \ldots, B\}$$

Completely Random Forests (CRFs) — Data encoding

1.

$$b_0 := -\infty \qquad\qquad b_B := \infty$$

$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0}\,\boxed{0}\,\overbrace{\boxed{0}}^{b_2 < x_{ij} \leq b_3}\,\underbrace{\boxed{1}}_{b_3 < x_{ij} \leq b_4}\,\boxed{0} = \{x_{ijk} : k = 1, \ldots, B\}$$

2. Then,

$$\mathbb{I}(x_{ij} \leq b_l) = \sum_{k=1}^{l} x_{ijk} \quad \text{and} \quad \mathbb{I}(x_{ij} > b_l) = \sum_{k=l+1}^{B} x_{ijk}$$

Completely Random Forests (CRFs) — Data encoding

1.

$$b_0 := -\infty \qquad\qquad b_B := \infty$$

$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0}\boxed{0}\overbrace{\boxed{0}}^{b_2 < x_{ij} \leq b_3}\underbrace{\boxed{1}}_{b_3 < x_{ij} \leq b_4}\boxed{0} = \{x_{ijk} : k = 1, \ldots, B\}$$

2. Then,

$$\mathbb{I}(x_{ij} \leq b_l) = \sum_{k=1}^{l} x_{ijk} \quad \text{and} \quad \mathbb{I}(x_{ij} > b_l) = \sum_{k=l+1}^{B} x_{ijk}$$

3. Similarly encode response category c, $y_i \to y_{ic} \in \{0, 1\}$.

Completely Random Forests (CRFs) — Data encoding

1. $$b_0 := -\infty \qquad\qquad b_B := \infty$$

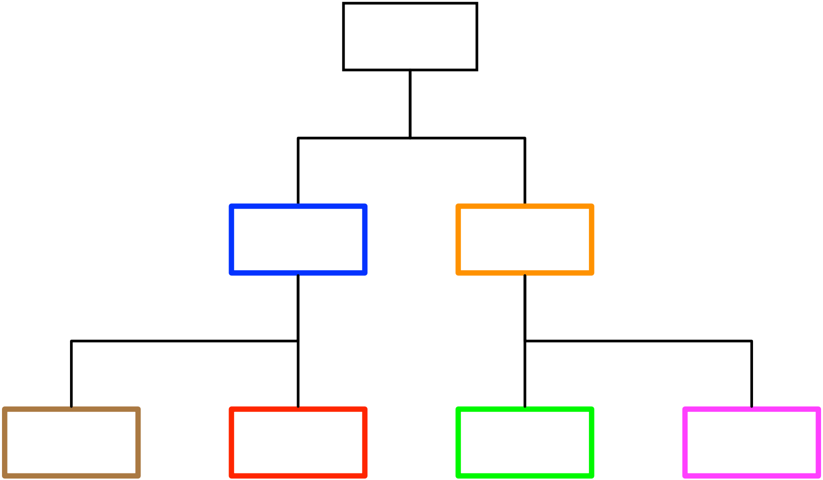
$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0}\,\boxed{0}\,\overbrace{\boxed{0}}^{b_2 < x_{ij} \leq b_3}\,\underbrace{\boxed{1}}_{b_3 < x_{ij} \leq b_4}\,\boxed{0} = \{x_{ijk} : k = 1, \ldots, B\}$$

2. Then,

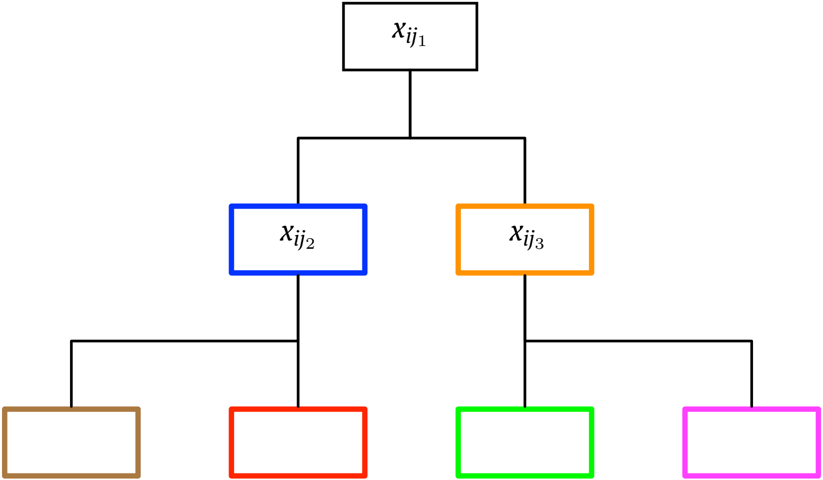
$$\mathbb{I}(x_{ij} \leq b_l) = \sum_{k=1}^{l} x_{ijk} \quad \text{and} \quad \mathbb{I}(x_{ij} > b_l) = \sum_{k=l+1}^{B} x_{ijk}$$

3. Similarly encode response category c, $y_i \to y_{ic} \in \{0, 1\}$.
4. Build a decision tree selecting variable j and split point b_l *completely* at random to a fixed depth.

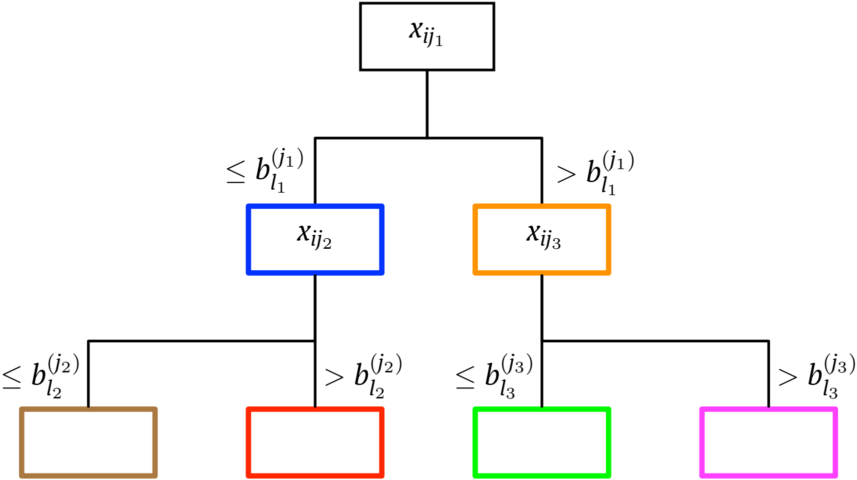
Completely Random Forests (CRFs) — Tree ‘fitting’, I



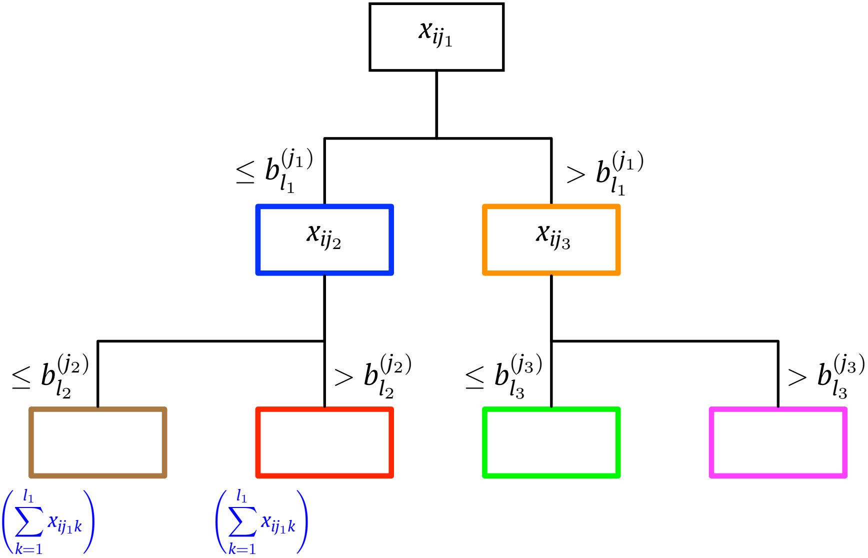
Completely Random Forests (CRFs) — Tree 'fitting', I



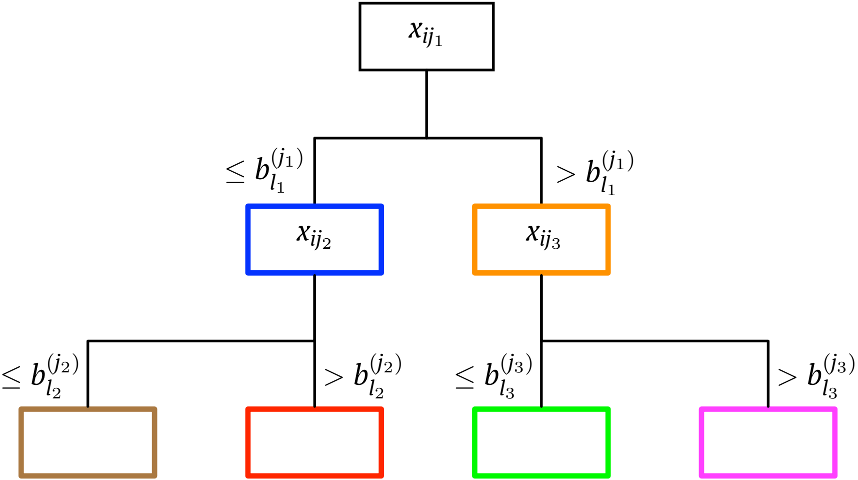
Completely Random Forests (CRFs) — Tree ‘fitting’, I



Completely Random Forests (CRFs) — Tree ‘fitting’, I



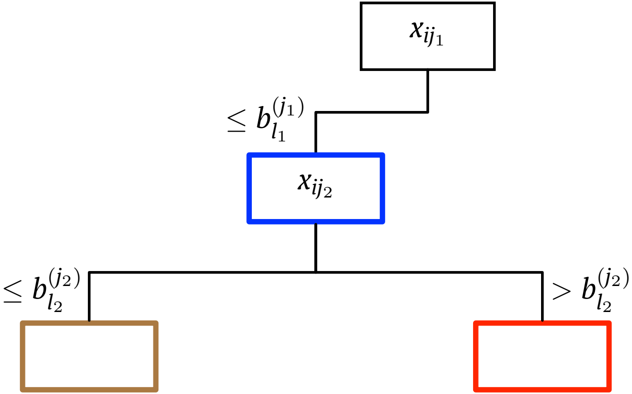
Completely Random Forests (CRFs) — Tree ‘fitting’, I



$$\left(\sum_{k=1}^{l_1} x_{ij_1k}\right)\left(\sum_{k=1}^{l_2} x_{ij_2k}\right) \quad \left(\sum_{k=1}^{l_1} x_{ij_1k}\right)\left(\sum_{k=l_2+1}^{B} x_{ij_2k}\right)$$

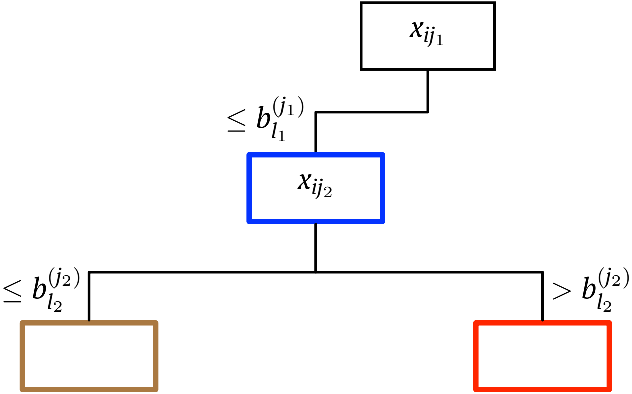
Exactly one terminal leaf indicator evaluates to 1, encrypted.

Completely Random Forests (CRFs) — Tree ‘fitting’, II



$$\left(\sum_{k=1}^{l_1} x_{ij_1k}\right)\left(\sum_{k=1}^{l_2} x_{ij_2k}\right) \qquad \left(\sum_{k=1}^{l_1} x_{ij_1k}\right)\left(\sum_{k=l_2+1}^{B} x_{ij_2k}\right)$$

Completely Random Forests (CRFs) — Tree ‘fitting’, II



$$\nu_{ic} = y_{ic} \left(\sum_{k=1}^{l_1} x_{ij_1k} \right) \left(\sum_{k=1}^{l_2} x_{ij_2k} \right) \qquad \nu_{ic} = y_{ic} \left(\sum_{k=1}^{l_1} x_{ij_1k} \right) \left(\sum_{k=l_2+1}^{B} x_{ij_2k} \right)$$

NB Must evaluate *all* branches and categories as blindfold.

Completely Random Forests (CRFs) — Prediction

Prediction involves:

- evaluating a new observation through all branches;
- taking product with corresponding vote totals for each class;
- summing across trees and across leaves to get total votes for each class.

Completely Random Forests (CRFs) — Prediction

Prediction involves:

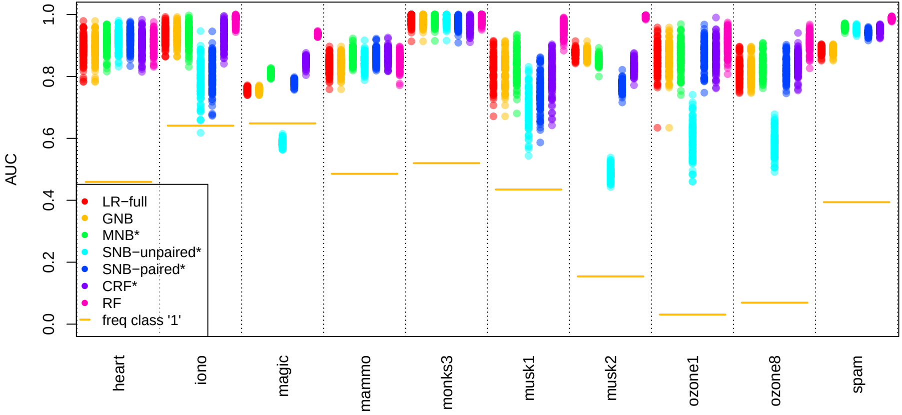
- evaluating a new observation through all branches;
- taking product with corresponding vote totals for each class;
- summing across trees and across leaves to get total votes for each class.

But, confused leaves with many votes can overwhealm certain ones with few. Random Forests usually use:

1. single vote per tree (requires comparison to find max)
2. relative class frequencies (requires division)

… developed novel 'stochastic fraction estimate', an unbiased approximation to 2.

Results


AUC
1.0
0.8
0.6
0.4
0.2
0.0
LR−full
GNB
MNB*
SNB−unpaired*
SNB−paired*
CRF*
RF
freq class '1'
heart
iono
magic
mammo
monks3
musk1
musk2
ozone1
ozone8
spam

HomomorphicEncryption R package

```
library("HomomorphicEncryption")
p <- parsHelp("FandV", lambda=128, L=5)
k <- keygen(p)
c1 <- enc(k$pk, 2); c2 <- enc(k$pk, 3)
cres <- c1 + c2 * c1
dec(k$sk, cres)
```

```
[1] 8
```

```
cmat <- enc(k$pk, matrix(1:9, nrow=3))
cmat2 <- cmat %*% cmat
dec(k$sk, cmat2)
```

```
     [,1] [,2] [,3]
[1,]   30   66  102
[2,]   36   81  126
[3,]   42   96  150
```

References I

Aslett, L. J. M. (2016). HomomorphicEncryption: Fully homomorphic encryption. `http://www.louisaslett.com/HomomorphicEncryption/`.

Aslett, L. J. M., Esperança, P. M., & Holmes, C. C. (2015a). A review of homomorphic encryption and software tools for encrypted statistical machine learning. *arXiv:1508.06574 [stat.ML]*.

Aslett, L. J. M., Esperança, P. M., & Holmes, C. C. (2015b). Encrypted statistical machine learning: New privacy preserving methods. *arXiv:1508.06845 [stat.ML]*.

Gentry, C. (2009). *A fully homomorphic encryption scheme* (PhD thesis). Stanford University. Retrieved from `<crypto.stanford.edu/craig>`

Rivest, R. L., Adleman, L., & Dertouzos, M. L. (1978). On data banks and privacy homomorphisms. *Foundations of Secure Computation*, 4/11: 169–80.