Doing Data Science Blindfolded

Louis J. M. Aslett (louis.aslett@durham.ac.uk)

Department of Mathematical Sciences
Durham University

Data Science North East Meetup
31st May 2018

Introduction


Cambridge
Analytica

CC0

Motivation

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

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

Motivation

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

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

Motivation

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

- 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

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

- 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);
- data sharing for inference (e.g. research institutes/corporate partners);

Motivation

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

- 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);
- data sharing for inference (e.g. research institutes/corporate partners);
- selling models without selling data (e.g. financial instruments);

Motivation

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

- 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);
- data sharing for inference (e.g. research institutes/corporate partners);
- selling models without selling data (e.g. financial instruments);
- massive pooling of sensitive data (e.g. phone app data);

Motivation

Privacy and security of commercially valuable or personally sensitive information is a growing concern in data science applications:

- 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);
- data sharing for inference (e.g. research institutes/corporate partners);
- selling models without selling data (e.g. financial instruments);
- massive pooling of sensitive data (e.g. phone app data);
- running confidential algorithms on confidential data (e.g. engineering reliability).

Security and privacy in data science : the challenges

Aguably, we can categorise into three intersecting challenges:

1. Keep my own data private
   - have access to all the data(?)
   - fitting to happen in hostile environment

Security and privacy in data science : the challenges

Aguably, we can categorise into three intersecting challenges:

1. Keep my own data private
    - have access to all the data(?)
    - fitting to happen in hostile environment

2. Pool my data with others
    - only release final model
    - keep raw data private

Security and privacy in data science : the challenges

Aguably, we can categorise into three intersecting challenges:

1. Keep my own data private
    - have access to all the data(?)
    - fitting to happen in hostile environment
2. Pool my data with others
    - only release final model
    - keep raw data private
3. Privacy of fitted models/predictions
    - avoid information leakage
    - side channel attacks

Security and privacy in data science : the tools

1. Differential privacy
   - well known and researched in statistics community
   - gives privacy guarantees on randomised algorithms

Security and privacy in data science : the tools

1. Differential privacy
   - well known and researched in statistics community
   - gives privacy guarantees on randomised algorithms

2. Homomorphic encryption
   - very little awareness or research in statistics community
   - allows computation on encrypted data

Security and privacy in data science : the tools

1. Differential privacy
    - well known and researched in statistics community
    - gives privacy guarantees on randomised algorithms

2. Homomorphic encryption
    - very little awareness or research in statistics community
    - allows computation on encrypted data

3. Homomorphic secret sharing
    - very little awareness or research in statistics community
    - allows multi-party computation on data with information theoretic security

Security and privacy in data science : the tools

1. Differential privacy
   - well known and researched in statistics community
   - gives privacy guarantees on randomised algorithms

2. Homomorphic encryption
   - very little awareness or research in statistics community
   - allows computation on encrypted data

3. Homomorphic secret sharing
   - very little awareness or research in statistics community
   - allows multi-party computation on data with information theoretic security

My take: we need all three working together

Encryption

Encryption basics (I)

Broadly speaking, an encryption scheme consists of:

- Unencrypted object, m, referred to as a *message*.
- Encrypted version, c, referred to as a *cipher text*.
- Single $(k_s) \in K_s$, or pair $(k_s, k_p) \in K_s \times K_p$, of 'keys'.
  - Single key means secret key scheme;
  - Pair of keys means public key scheme.
- Two algorithms
  - `Enc`, taking k_p and m to produce c.
  - `Dec`, taking k_s and c to produce m.
- `Enc` and `Dec` satisfy:

$$m = \texttt{Dec}(k_s, \texttt{Enc}(k_p, m)) \quad \forall\, m \in M$$

Encryption basics (II)

Fundamental idea ...

Easy

$$\texttt{Enc}(k_p, m) \rightleftharpoons c$$

Hard without k_s

$$\texttt{Dec}(k_s, c) = m$$

The *security level* of an encryption scheme is the order of the number of operations required to crack it (decrypt without k_s).

Clearly, an upper bound on the security of an encryption scheme is $O(|K_s|)$, since a brute force attack which tries every possible secret key will succeed.

Concepts: Public key -vs- private key

Presumably public key schemes are always better: can just choose not to distribute k_p?

Not really. Public key schemes tend to:

- have much larger cipher texts than messages, so are space inefficient.
- have greater computational cost, so are compute inefficient.
- rely on complex mathematical constructions rather than bit-level operations, so are hard to design custom hardware for.

Hence, private key schemes still involved in almost all cryptography, perhaps wrapped in a public key scheme. More anon ...

Concept: Semantic security

Definition (Semantic security)

An encryption scheme is said to be *semantically secure* if knowledge of the cipher text for some message has vanishingly small probability of revealing further information about any other encrypted message.

Informally: repeated encryption of same message renders different and seemingly unrelated cipher texts with high probability.

Why do we care? For private key scheme you don't. However, in a public key scheme where $|M| \ll |K_s|$ or probable messages are known, an attacker can perform a 'chosen plaintext attack' if not semantically secure — simply encrypt using the public key and compare.

Homomorphic Encryption

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 ...

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.

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

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

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.

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

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.

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

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.

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

Limitations of homomorphic encryption

1. Message space (what we can encrypt)
    - Commonly only easy to encrypt binary/integers/polynomials

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)

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

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!

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!

See accessible intro in L. Aslett et al. (2015a).

We really are doing data science blindfolded …

Existing implementations

- `libfhe` (Minar 2010) compact single C file library implementing Gentry (2010)
- 'Scarab' (Perl et al. 2011) low level C library implementing Smart & Vercauteren (2010)
- 'HELib' (Halevi & Shoup 2014) most impressive library, in C++ implementing Brakerski et al. (2012) and lots beyond the bare bones cryptography (i.e. Polynomial Chinese Remainder Theorem + automorphisms)
- more besides …

However, these all tend to be very low-level libraries.

HomomorphicEncryption R package (Aslett 2014)

All core code in high-performance multi-threaded C++, but accessible via simple R functions and overloaded operators:

```
library("HomomorphicEncryption")

p <- pars("FandV")
k <- keygen(p)
c1 <- enc(k$pk, c(42,34))
c2 <- enc(k$pk, c(7,5))
cres1 <- c1 + c2
cres2 <- c1 * c2
cres3 <- c1 %*% c2
dec(k$sk, cres1)
dec(k$sk, cres2)
dec(k$sk, cres3)
```

Demo

Encrypted Machine Learning

Machine Learning Encrypted?

Lots of constraints! Are traditional data science techniques out of reach to run on encrypted data?

Here we'll cover the basics of a novel variant of random forests (see L. Aslett et al. (2015b) for full mathematical details).

Machine Learning Encrypted?

Lots of constraints! Are traditional data science techniques out of reach to run on encrypted data?

Here we'll cover the basics of a novel variant of random forests (see L. Aslett et al. (2015b) for full mathematical details).

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

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

Completely Random Forests (CRFs) — Data encoding

1

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

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

Completely Random Forests (CRFs) — Data encoding

1

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

$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \begin{array}{|c|c|c|c|c|}\hline 0 & 0 & \overbrace{0}^{b_2 < x_{ij} \leq b_3} & \underbrace{1}_{b_3 < x_{ij} \leq b_4} & 0 \\ \hline \end{array} = \{x_{ijk} : k = 1, \ldots, B\}$$

Completely Random Forests (CRFs) — Data encoding

1.

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

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

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.

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

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

2. Then,

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

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

Completely Random Forests (CRFs) — Data encoding

1.

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

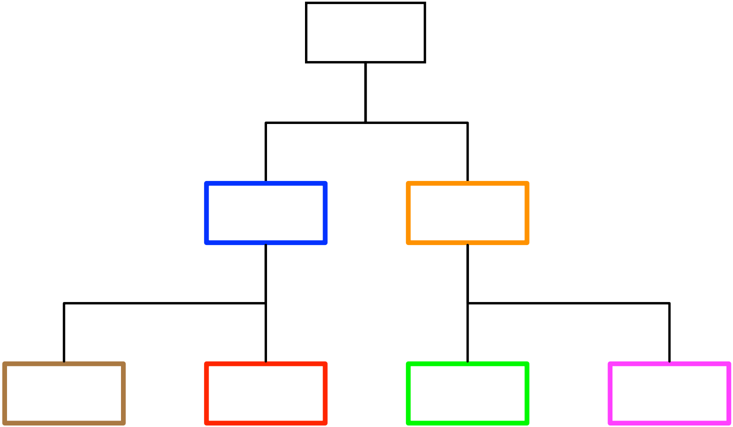
$$x_{ij} \in \mathbb{R} \xrightarrow{B \text{ quantiles}} \boxed{0}\boxed{0}\overbrace{\boxed{0}}^{b_2 < x_{ij} \leq b_3}\underbrace{\boxed{1}}_{b_3 < x_{ij} \leq b_4}\boxed{0} = \{x_{ijk} : k = 1, \dots, B\}$$

2. Then,

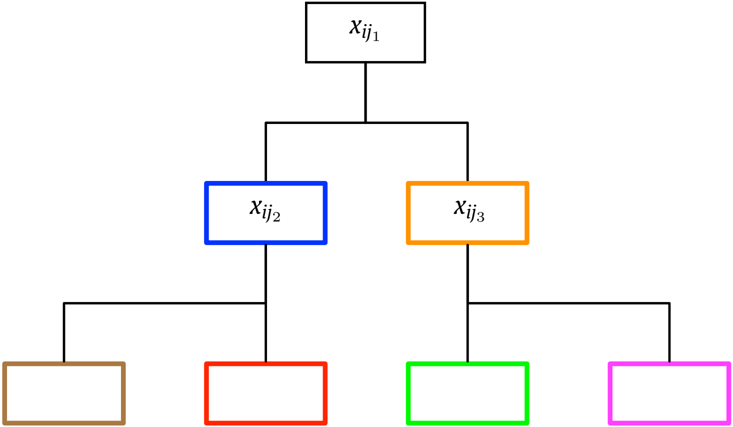
$$\mathbb{I}(x_{ij} \leq b_l) = \sum_{k=1}^{l} x_{ijk} \quad \text{and} \quad \mathbb{I}(x_{ij} > b_l) = \sum_{k=l+1}^{B} x_{ijk}$$

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.

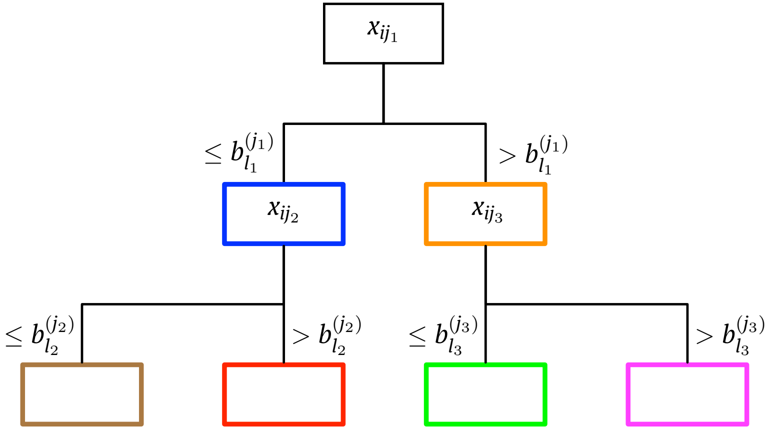
Completely Random Forests (CRFs) — Tree ‘fitting’, I



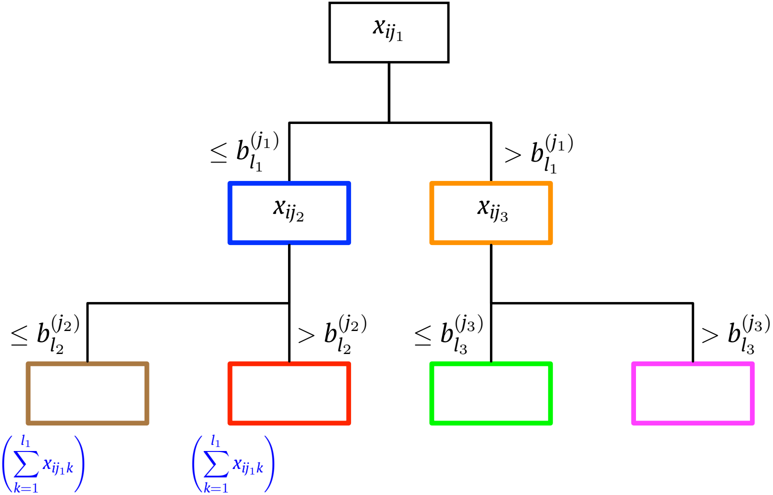
Completely Random Forests (CRFs) — Tree ‘fitting’, I



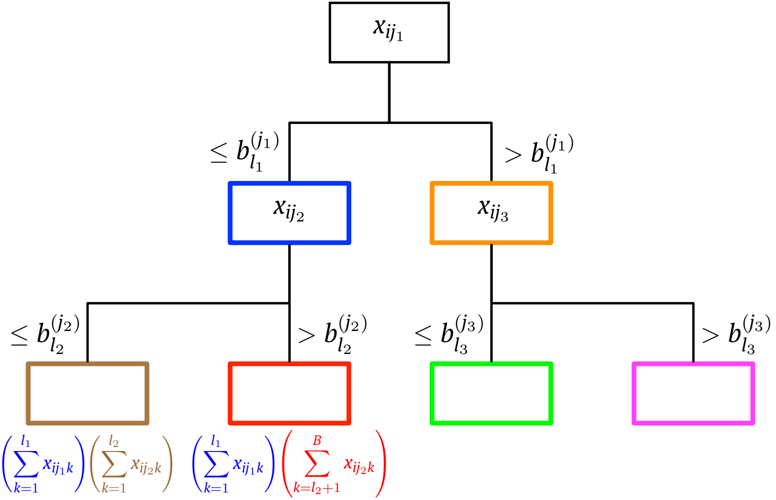
Completely Random Forests (CRFs) — Tree ‘fitting’, I



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

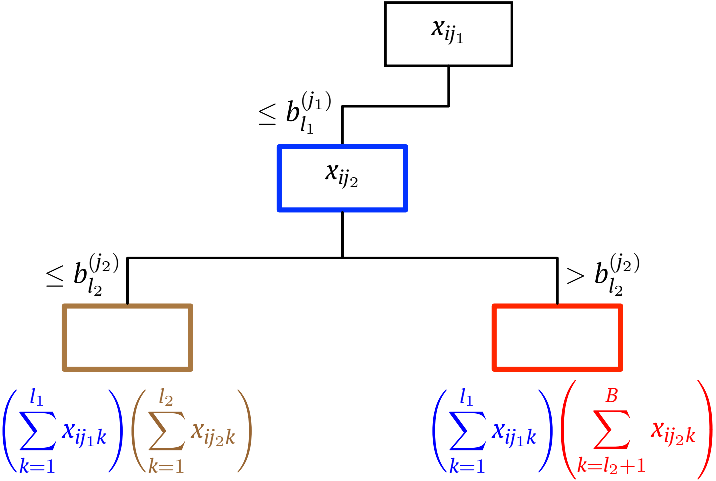


Completely Random Forests (CRFs) — Tree 'fitting', I

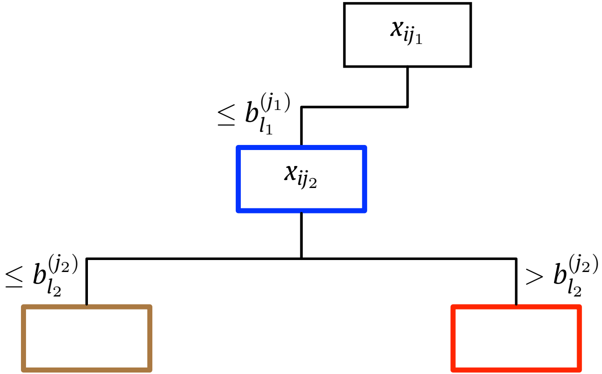


Exactly one terminal leaf indicator evaluates to 1, encrypted.

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



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:

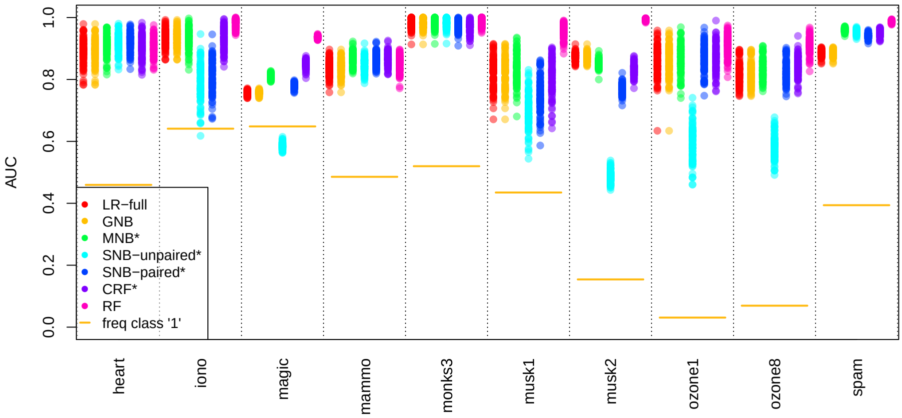
- evaluating a new observation through all branches;
- taking product with corresponding vote totals for each class;
- summing across trees and across leaves to get total votes for each class.

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 approximation to 2. See paper for details.

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

Computational considerations

Note that CRFs are parallelisable right down to the individual observation, which helps with ameliorating the cost of encrypted computation.

Computational considerations

Note that CRFs are parallelisable right down to the individual observation, which helps with ameliorating the cost of encrypted computation.

Wisconsin data ($N = 547$)

- Launched
  2×18 servers $\times 32$ cores $= 1,152$ CPU core cluster on Amazon EC2
  $\Rightarrow$ 576 Dublin & 576 São Paulo
- Fit 50 trees in Dublin, 50 in São Paulo
  - unique `set.seed()` for each region
- Data split into 17 shards of 32 obs $+ 1$ shard 3 obs $\Rightarrow 1$ datum per core!
- Reduction sum of votes in each region and combine regions $\Rightarrow 100$ tree forest

Computational considerations

Note that CRFs are parallelisable right down to the individual observation, which helps with ameliorating the cost of encrypted computation.

Wisconsin data ($N = 547$)

- Launched 2×18 servers $\times\ 32$ cores $= 1,152$ CPU core cluster on Amazon EC2 $\Rightarrow$ 576 Dublin & 576 São Paulo
- Fit 50 trees in Dublin, 50 in São Paulo
  - unique `set.seed()` for each region
- Data split into 17 shards of 32 obs $+$ 1 shard 3 obs $\Rightarrow$ 1 datum per core!
- Reduction sum of votes in each region and combine regions $\Rightarrow$ 100 tree forest

1h 36m

US$ 23.86

More Private Data Science

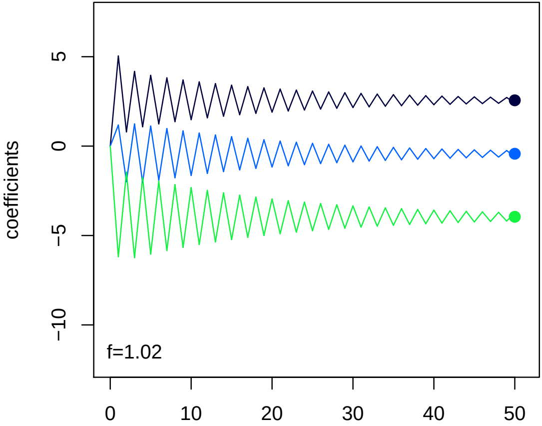
Encrypted naïve Bayes (L. Aslett et al. 2015b)

- Naïve Bayes classifier usually solves classification using a generative approach, i.e. by modelling the distribution of the predictors (Ng & Jordan 2002).
- We show possible to model decision boundary between response classes explicitly (without parametric model) while still remaining in the naïve Bayes framework.
- Involves a simple approximation to iteratively reweighted least squares for logistic regression.
- Typically underperforms completely random forest method, but faster to fit.

Encrypted linear modelling (Esperança et al. 2017)

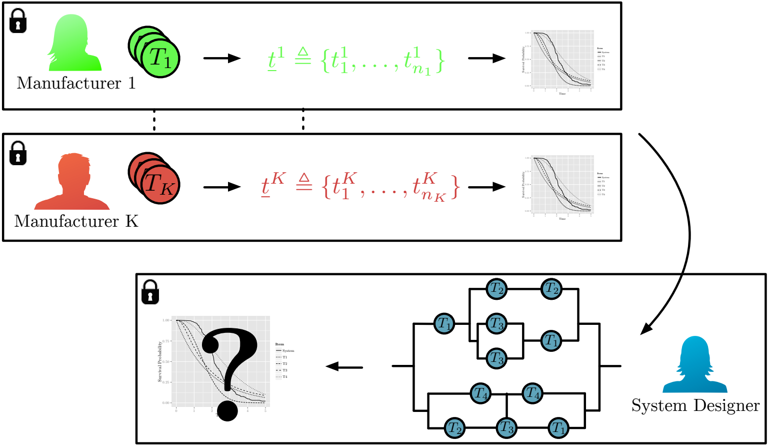
$$\underline{y} = X\underline{\beta} + \underline{\varepsilon}, \quad \varepsilon_i \sim N(0, \sigma^2)$$

Using coordinate descent accelerated by van Wijngaarden transformation.



Secure multi-party system reliability (Aslett 2016)

Inference on system/network reliability whilst *maintaining privacy requirements* of all parties.



Approximate Bayesian Computation Done Encrypted (ABCDE) — pending preprint

- **Eve** has a private model, including prior information which may itself be private.
- **Cain** and **Abel** have private data which is relevant to the fitting of Eve's model.

Can Eve fit a model, pooling data from Cain and Abel without observing their raw data and without revealing her model and prior information? Abel also doesn't trust Cain ...

$\pi(\cdot \mid \psi)$
$\pi(\psi)$

$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=1}^{n_1}$

$\{\mathbf{x}_i = (x_{i1}, \dots, x_{id})\}_{i=n_1+1}^{N}$

Confidential MCMC — pending preprint

If the model and prior are not private (only the data), we can perform *exact* pooled Bayesian inference using MCMC, with the Metropolis-Hastings acceptance decision for a proposal $\theta_i \to \theta'$ made privately:

With probability $\alpha(\theta_i, \theta')$ set $\theta_{i+1} = \theta'$, else set $\theta_{i+1} = \theta_i$ where $\alpha(\theta_i, \theta') = \min\{1, r(\theta_i, \theta')\}$ with

$$r(\theta_i, \theta') := \frac{\pi(\theta')q(\theta_i \mid \theta')}{\pi(\theta_i)q(\theta' \mid \theta_i)}$$

Homomorphic Secret Sharing + Differential Privacy

Shameless plug! Knowledge Transfer Partnership

Forthcoming KTP associate job, based at Atom Bank working with me and Camila Caiado at Durham University.

Jointly working with Computer Science KTP associate based at Atom and working with Newcastle University.

Statistical modelling and encrypted statistics for mortgage books.

Expected to advertise for a September – October 2018 start.

Atom bank

Durham University

References I

Aslett, L. J. M. (2014). HomomorphicEncryption: Fully homomorphic encryption. http://www.louisaslett.com/HomomorphicEncryption/.

Aslett, L. J. M. (2016). Cryptographically secure multiparty evaluation of system reliability.

Aslett, L., Esperança, P., & Holmes, C. (2015a). *A review of homomorphic encryption and software tools for encrypted statistical machine learning*. University of Oxford.

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

Brakerski, Z., Gentry, C., & Vaikuntanathan, V. (2012). (Leveled) fully homomorphic encryption without bootstrapping. *Proceedings of the 3rd innovations in theoretical computer science conference*, pp. 309–25. ACM.

Esperança, P. M., Aslett, L. J. M., & Holmes, C. C. (2017). Encrypted accelerated least squares regression. Singh A. & Zhu J. (eds) *Proceedings of the 20th international conference on artificial intelligence and statistics*, Proceedings of machine learning research, Vol. 54, pp. 334–43. Fort

References II

Lauderdale, FL, USA: PMLR.

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

Gentry, C. (2010). Computing arbitrary functions of encrypted data. *Communications of the ACM*, 53/3: 97–105. ACM.

Halevi, S., & Shoup, V. (2014). HElib. https://github.com/shaih/HElib.

Minar, J. (2010). Libfhe. https://github.com/rdancer/fhe/tree/master/libfhe.

Ng, A. Y., & Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, pp. 841–8.

Perl, H., Brenner, M., & Smith, M. (2011). Scarab library. https://hcrypt.com/scarab-library/.

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

Smart, N. P., & Vercauteren, F. (2010). Fully homomorphic encryption with relatively small key and ciphertext sizes. *Public key cryptography–PKC 2010*,

References III

pp. 420–43. Springer.