Doing Machine Learning Blindfolded

Louis J. M. Aslett (`louis.aslett@durham.ac.uk`)
Department of Mathematical Sciences
Durham University

Joint work with Pedro M. Esperança (Imperial)
and Chris C. Holmes (Oxford)

RSS Privacy Meeting
2 May 2017

Durham University

Outline

1. Introduction
   - Motivation
   - High-level overview of homomorphic encryption
   - Discussion of constraints
2. Software
   - Discussion of implementation issues and `HomomorphicEncryption` R package.
3. Encrypted Machine Learning
   - Completely Random Forests (CRF)
   - Extreme variant of extremely random forests
   - Including ‘stochastic fraction estimator’
   - Embarrasingly parallel down to single datum
4. Other / Future Work
   - Brief discussion of other complete and in progress projects

Introduction

Motivation

Security in statistics and machine learning 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 …

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

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

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.

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

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

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

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.

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

Formal definition

Definition (Homomorphic encryption scheme)

An encryption scheme is said to be *homomorphic* if there is a set of operations $\circ \in \mathcal{F}_M$ acting in message space, M, that have corresponding operations $\diamond \in \mathcal{F}_C$ acting in cipher text space, C, satisfying the property:

$$\texttt{Dec}(k_s, \texttt{Enc}(k_p, m_1) \diamond \texttt{Enc}(k_p, m_2)) = m_1 \circ m_2 \quad \forall\, m_1, m_2 \in M$$

A scheme is *fully homomorphic* if $\mathcal{F}_M = \{+, \times\}$ and an arbitrary number of such operations are possible.

Formal definition

Definition (Homomorphic encryption scheme)

An encryption scheme is said to be *homomorphic* if there is a set of operations $\circ \in \mathcal{F}_M$ acting in message space, M, that have corresponding operations $\diamond \in \mathcal{F}_C$ acting in cipher text space, C, satisfying the property:

$$\mathtt{Dec}(k_s, \mathtt{Enc}(k_p, m_1) \diamond \mathtt{Enc}(k_p, m_2)) = m_1 \circ m_2 \quad \forall\, m_1, m_2 \in M$$

A scheme is *fully homomorphic* if $\mathcal{F}_M = \{+, \times\}$ and an arbitrary number of such operations are possible.

$\{+, \times\}$ pretty limiting? Note that if $M = \mathrm{GF}(2)$, then:

- $+ \equiv \veebar$, i.e. XOR, 'exclusive or'
- $\times \equiv \wedge$, i.e. AND, 'and'

Moreover, *any* electronic logic gate can be constructed using only XOR and AND gates.

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!

We really are doing statistics blindfolded ...

Software

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

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!

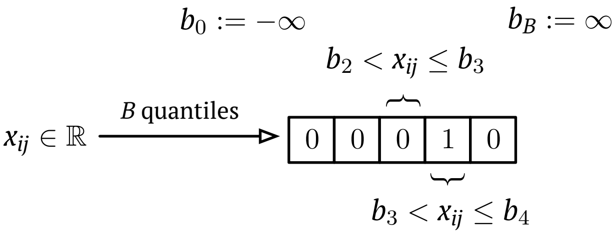
Completely Random Forests (CRFs) — Data encoding

1

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

Completely Random Forests (CRFs) — Data encoding

1



Completely Random Forests (CRFs) — Data encoding

1

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

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

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.

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

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

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

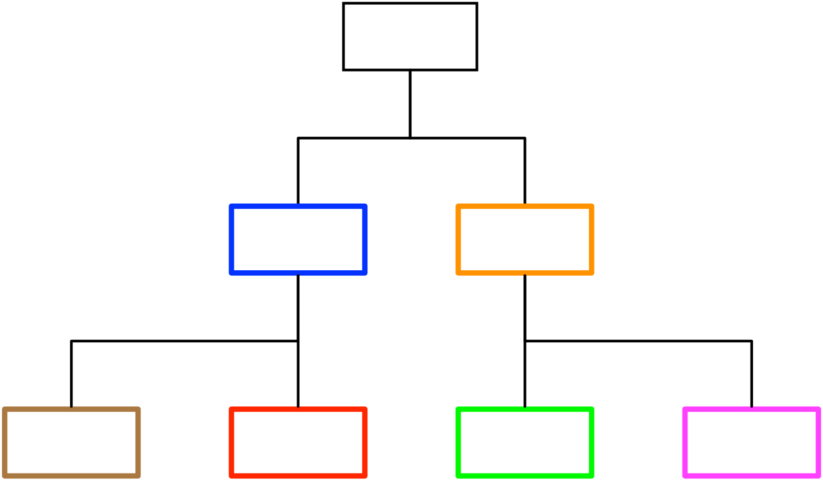
$$x_{ij} \in \mathbb{R} \xrightarrow{B\text{ quantiles}} \begin{array}{|c|c|c|c|c|}\hline 0 & 0 & \overbrace{0}^{b_2 < x_{ij} \le b_3} & \underbrace{1}_{b_3 < x_{ij} \le b_4} & 0 \\ \hline \end{array} = \{x_{ijk} : k = 1, \dots, B\}$$

2. Then,

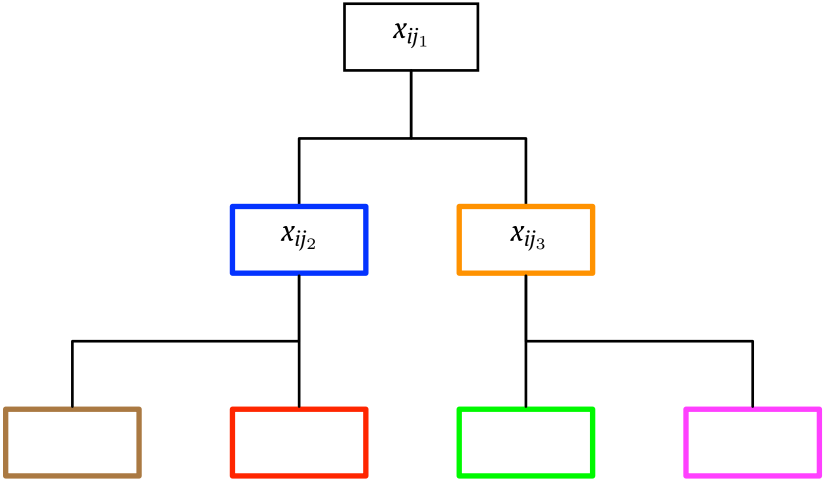
$$\mathbb{I}(x_{ij} \le b_l) = \sum_{k=1}^{l} x_{ijk} \quad \text{and} \quad \mathbb{I}(x_{ij} > b_l) = \sum_{k=l+1}^{B} x_{ijk}$$

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

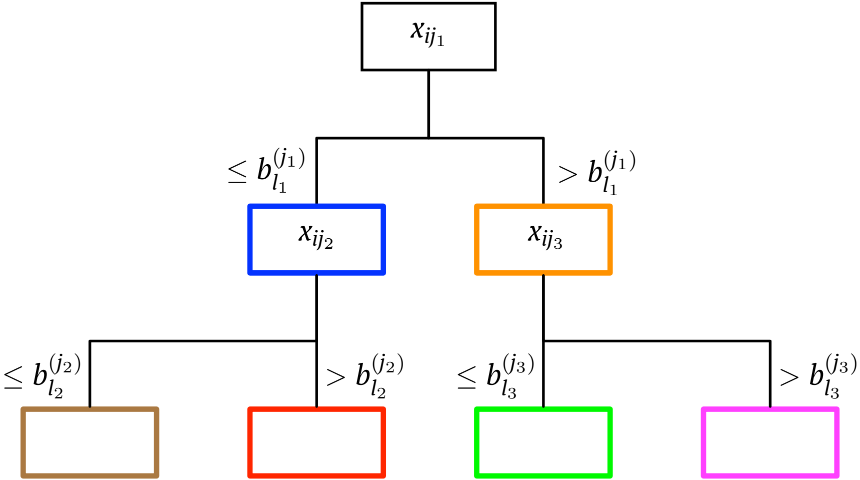
CRFs — Tree ‘fitting’, I



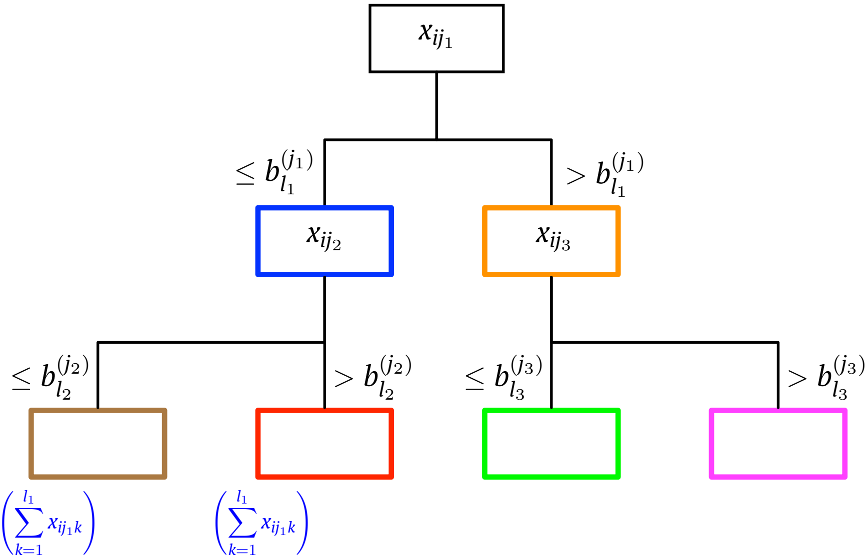
CRFs — Tree 'fitting', I



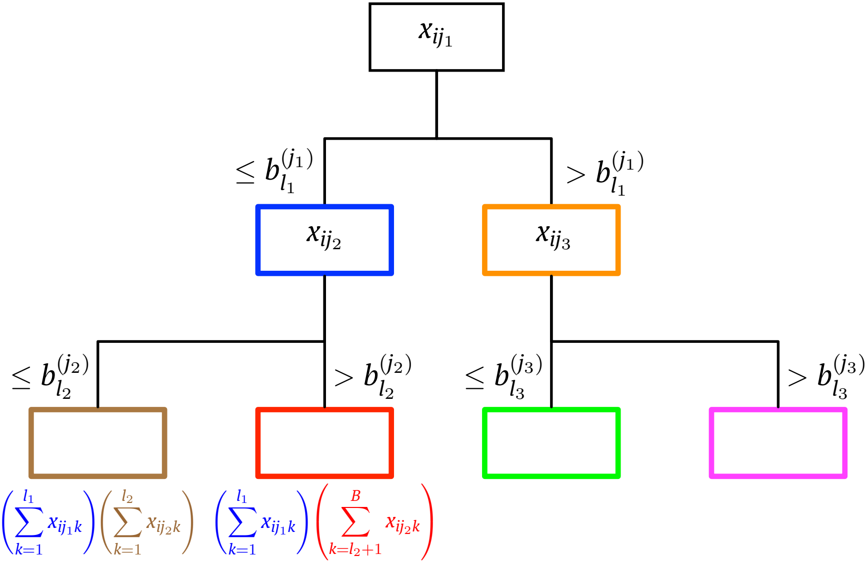
CRFs — Tree ‘fitting’, I



CRFs — Tree 'fitting', I

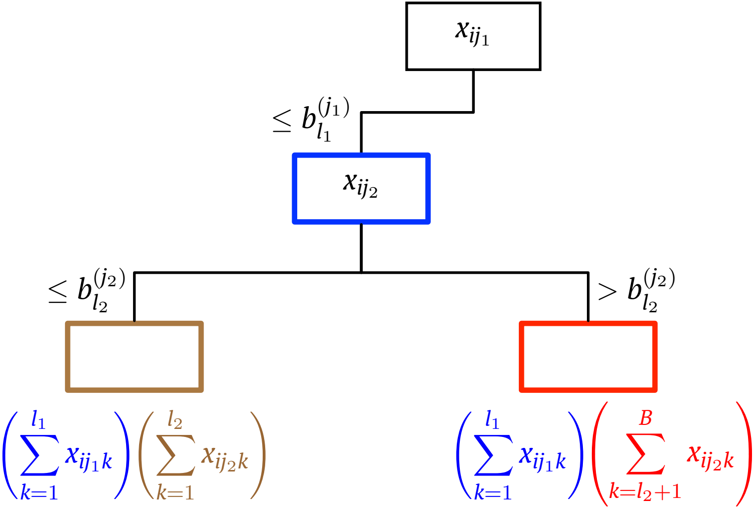


CRFs — Tree ‘fitting’, I

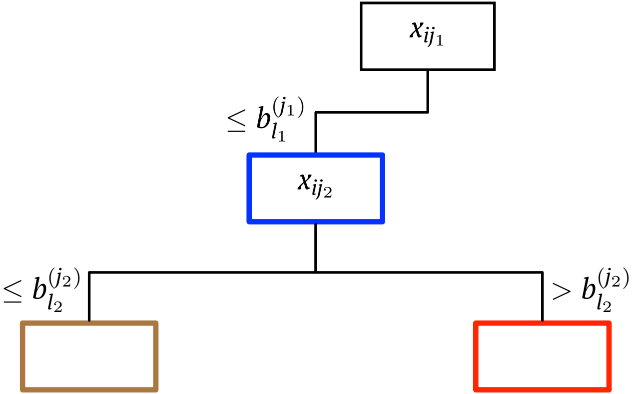


Exactly one terminal leaf indicator evaluates to 1, encrypted.

CRFs — Tree 'fitting', II



CRFs — Tree ‘fitting’, II



$$\nu_{ic} = y_{ic} \left(\sum_{k=1}^{l_1} x_{ij_1 k} \right) \left(\sum_{k=1}^{l_2} x_{ij_2 k} \right) \qquad \nu_{ic} = y_{ic} \left(\sum_{k=1}^{l_1} x_{ij_1 k} \right) \left(\sum_{k=l_2+1}^{B} x_{ij_2 k} \right)$$

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

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.

CRFs — Prediction

Prediction involves:

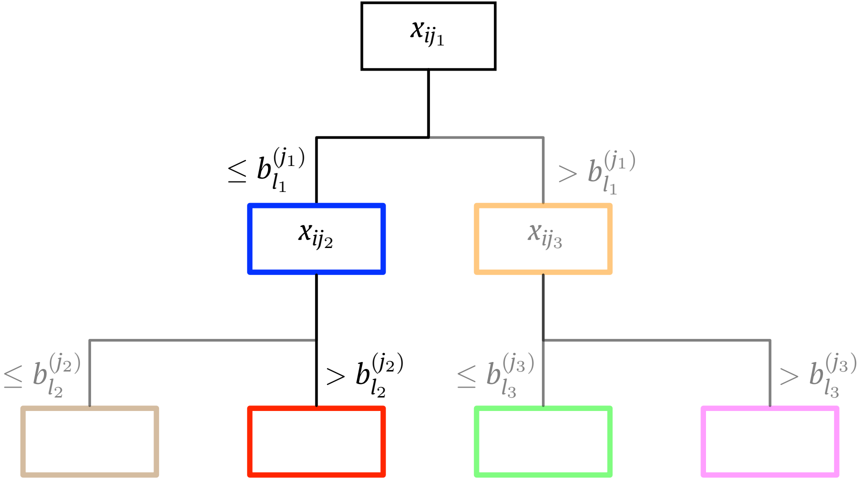
- evaluating a new observation through all branches;
- taking product with corresponding vote totals for each class;
- summing across trees and across leaves to get total votes for each class.

Random Forests usually use:

1. single vote per tree (requires comparison to find max)
2. relative class frequencies (requires division and $[0, 1]$ value)

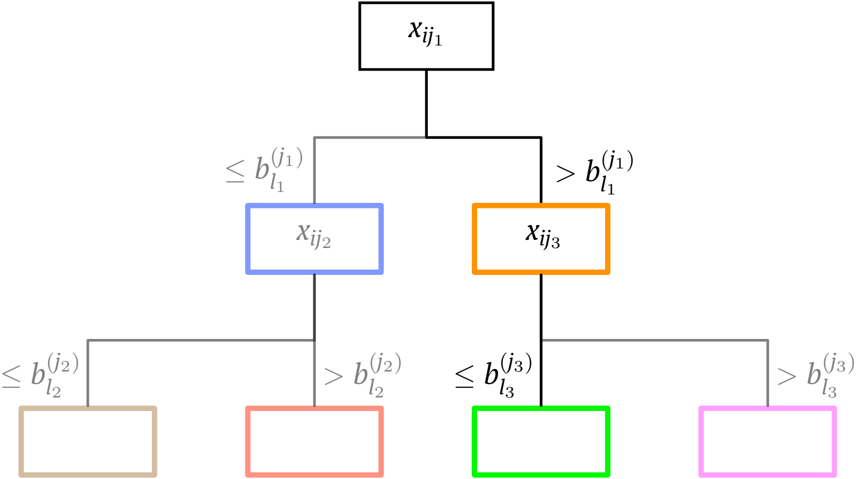
But here trees contribute raw ‘vote’ totals to the prediction: confused leaves with many votes can overwhealm certain ones with few.

CRFs — Raw votes problem



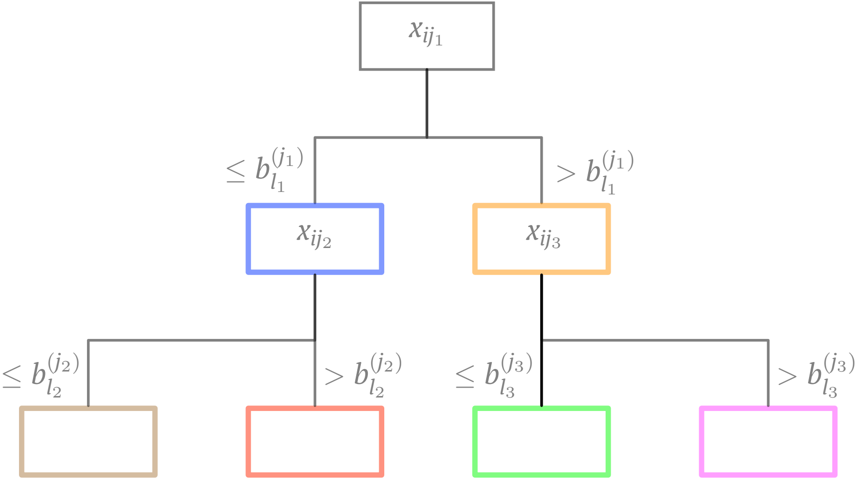
class	A	B	C
votes, $\sum_i \nu_{ic}$	78	7	15

CRFs — Raw votes problem



class	A	B	C
votes, $\sum_i \nu_{ic}$	1007	1080	1015

CRFs — Raw votes problem



class	A	B	C
$\Rightarrow$ overall votes, $\sum_i \nu_{ic}$	1085	1087	1030

Relative class frequencies

Let ν_c be the number of votes for class c in a leaf. The relative class frequency contribution should be:

$$\frac{\nu_c}{\sum_c \nu_c}$$

But, this belongs to $[0, 1]$ which we can't represent and involves division.

Relative class frequencies

Let ν_c be the number of votes for class c in a leaf. The relative class frequency contribution should be:

$$\frac{\nu_c}{\sum_c \nu_c}$$

But, this belongs to $[0, 1]$ which we can't represent and involves division.Target equivalently:

$$\nu_c \left\lfloor \frac{N}{\sum_c \nu_c} \right\rceil$$

where N is the number of training observations.

- By construction $\sum_c \nu_c \leq N$, so $0 \leq \frac{\sum_c \nu_c}{N} \leq 1$
- Recall, $X \sim \text{Geometric}(p) \implies \mathbb{E}[X] = p^{-1}$

Stochastic fraction estimate (I)

Thus, an unbiased approximation to fraction is draw from Geometric distribution with probability $\frac{\sum_c \nu_c}{N}$.

Not really helping … any better than division?!

Stochastic fraction estimate (I)

Thus, an unbiased approximation to fraction is draw from Geometric distribution with probability $\frac{\sum_c \nu_c}{N}$.

Not really helping … any better than division?!

Crucial observation: $\nu_c := \sum_{i=1}^N \nu_{ic}$ where $\nu_{ic} \in \{0, 1\} \,\forall\, i, c$.

(recall ν_{ic} is 1 if training obs. i was of class c and fell in this leaf of the decision tree … leaf indices supressed)

Stochastic fraction estimate (I)

Thus, an unbiased approximation to fraction is draw from Geometric distribution with probability $\frac{\sum_c \nu_c}{N}$.

Not really helping … any better than division?!

Crucial observation: $\nu_c := \sum_{i=1}^N \nu_{ic}$ where $\nu_{ic} \in \{0, 1\} \,\forall\, i, c$.

(recall ν_{ic} is 1 if training obs. i was of class c and fell in this leaf of the decision tree … leaf indices supressed)

$\implies$ blind sampling with replacement from $\{\sum_c \nu_{ic} : i = 1, \dots, N\}$ will produce an encrypted 1 with probability exactly $\frac{\sum_c \nu_c}{N}$.

$\implies$ can blind sample the latent bernoulli process underlying a Geometric$\left(p = \frac{\sum_c \nu_c}{N}\right)$ random variable.

Stochastic fraction estimate (II)

New problem! count number of leading zeros in an encrypted Bernoulli process.

Stochastic fraction estimate (II)

New problem! count number of leading zeros in an encrypted Bernoulli process.

Inspiration from CPU hardware algorithm for renormalising the mantissa of an IEEE floating point number.

Let $\xi_1, \dots, \xi_M$ be a resampled vector ($\xi_i = \sum_c \eta_{cj}$, some j) and assume M is a power of 2.

1. For $l \in \{0, \dots, \log_2(M) - 1\}$:
   - Set $\xi_i = \xi_i \vee \xi_{i-2^l} = \xi_i + \xi_{i-2^l} - \xi_i \xi_{i-2^l} \quad \forall\, 2^l + 1 \leq i \leq M$
2. The number of leading zeros is $M - \sum_{i=1}^{M} \xi_i$

Corresponds to increasing power of 2 bit-shifts OR'd with itself, all computable encrypted.

$$\implies \left\lfloor \frac{N}{\sum_c \nu_c} \right\rfloor \approx M - \sum_{i=1}^{M} \xi_i + 1$$

Stochastic fraction estimate (III)

CPU hardware algorithm for mantissa normalisation

	0	0	1	0	1	1	0	0					
$l = 0$		0	0	1	0	1	1	0	0				
$\vee$	0	0	1	1	1	1	1	0					
$l = 1$			0	0	1	0	1	1	0	0			
$\vee$	0	0	1	1	1	1	1	1					
$l = 2$					0	0	1	0	1	1	0	0	
$\vee$	0	0	1	1	1	1	1	1					$\Rightarrow M - \sum = 2$

Stochastic fraction estimate (IV)

Bias

Clearly, since blindfolded can't sample *until* a 1 observed, so choose a fixed M and accept small bias.

Stochastic fraction estimate (IV)

~~Bias~~ Shrinkage

Clearly, since blindfolded can't sample *until* a 1 observed, so choose a fixed M and ~~accept~~ celebrate small ~~bias~~ shrinkage.

Stochastic fraction estimate (IV)

~~Bias~~ Shrinkage

Clearly, since blindfolded can't sample *until* a 1 observed, so choose a fixed M and ~~accept~~ celebrate small ~~bias~~ shrinkage.

The shrinkage is mild unless there are fewer than $\frac{N}{M}$ observations in the leaf, in which case the shrinkage is more extreme: this is desirable because it shrinks the influence of underpopulated leaves.

e.g. $N = 1000, M = 32 \implies$ heavy shrinkage for leaves with < 31 observations.

Stochastic fraction estimate (IV)

~~Bias~~ Shrinkage

Clearly, since blindfolded can't sample *until* a 1 observed, so choose a fixed M and ~~accept~~ celebrate small ~~bias~~ shrinkage.

The shrinkage is mild unless there are fewer than $\frac{N}{M}$ observations in the leaf, in which case the shrinkage is more extreme: this is desirable because it shrinks the influence of underpopulated leaves.

e.g. $N = 1000, M = 32 \implies$ heavy shrinkage for leaves with < 31 observations.

Computational consideration

Multiplicative depth of this algorithm is M, which must be factored into tree building.

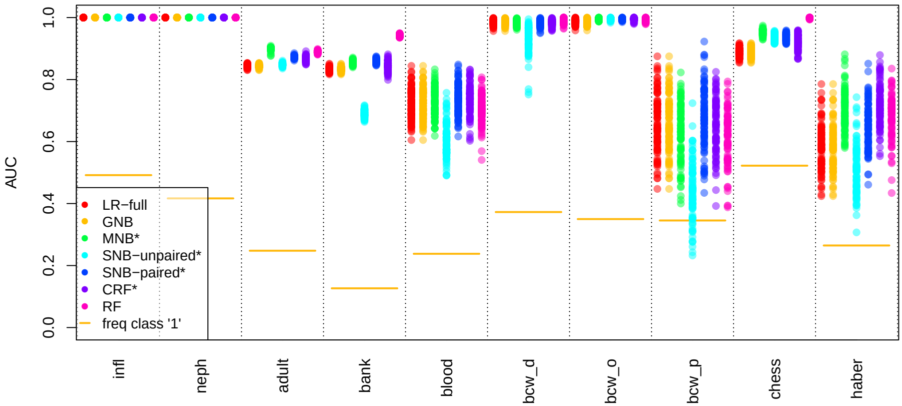
Theoretical homomorphic scheme requirements

To build a forest of trees with L levels, the homomorphic encryption scheme must support:

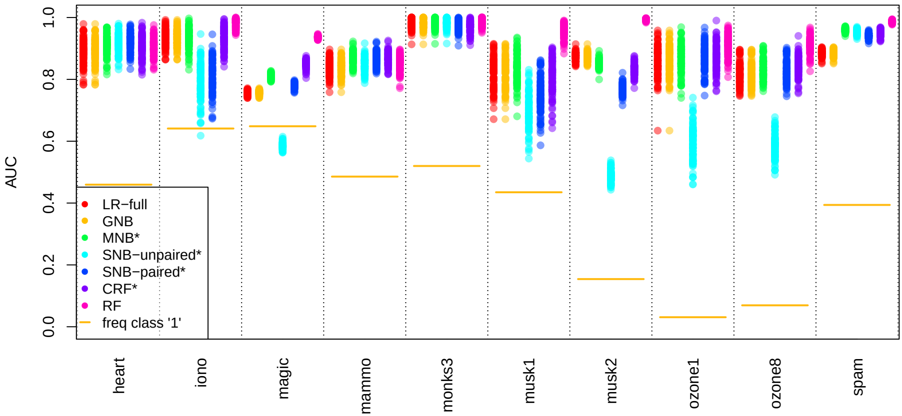
- depth L multiplications for tree building
- depth $L + M$ for stochastic fraction adjustment
- depth $2L + M$ for building, adjustment and prediction.

Furthermore, for the current generation of Ring Learning With Errors encryption schemes where the message space is a polynomial ring, it must support coefficients up to $T \max\{\sum_i y_{ic} : c = 1, \dots, |\mathcal{C}|\}$.

Results (I)

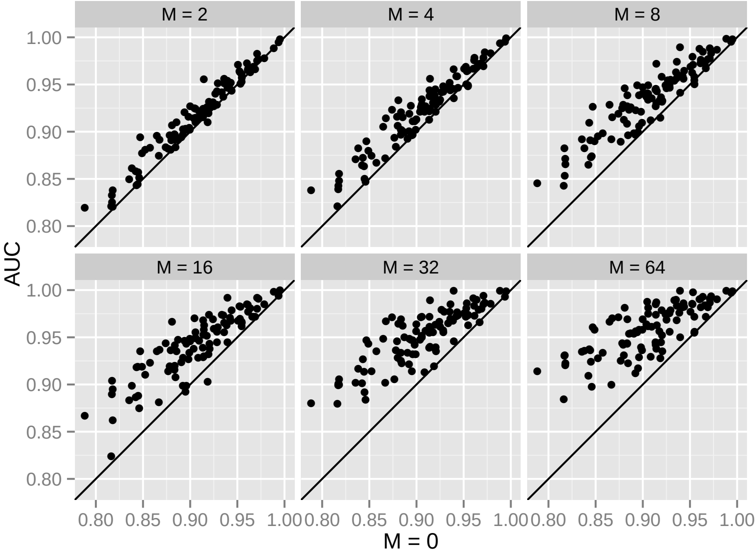

AUC
0.0
0.2
0.4
0.6
0.8
1.0
LR−full
GNB
MNB*
SNB−unpaired*
SNB−paired*
CRF*
RF
freq class '1'
infl
neph
adult
bank
blood
bcw_d
bcw_o
bcw_p
chess
haber

Results (II)

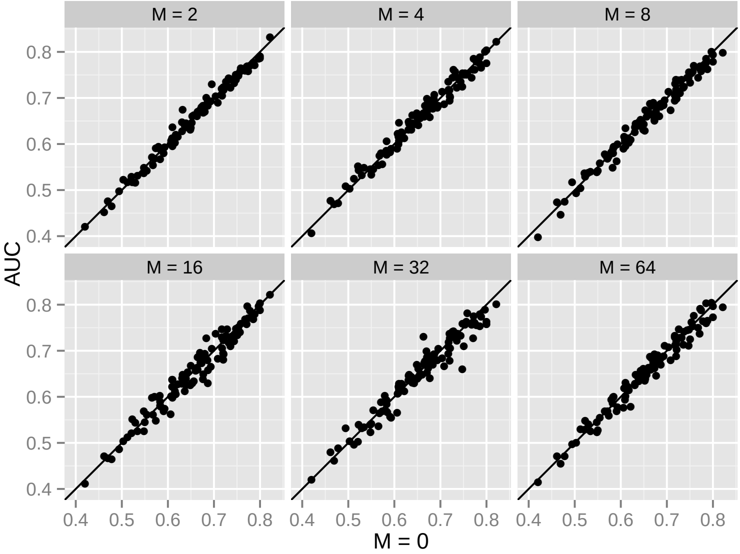


AUC
0.0
0.2
0.4
0.6
0.8
1.0
LR−full
GNB
MNB*
SNB−unpaired*
SNB−paired*
CRF*
RF
freq class '1'
heart
iono
magic
mammo
monks3
musk1
musk2
ozone1
ozone8
spam

Stochastic fraction effect (best)



Stochastic fraction effect (worst)



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

amazon web services

1h 36m

US$ 23.86

Other / Future Work

Other / Future Work

1. Semi-parametric naive Bayes with logistic decision boundary
   - embedded approximation to logistic regression
2. Linear models (see Pedro's talk)
   - gradient decent based method
   - ridge penalties
   - lasso(?)
3. Multi-party evaluation of system reliability
   - keep system design secret
   - keep component lifetime test data secret
4. Approximate Bayesian Computation
   - classifier replacing summary statistics

References

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. *arXiv:1604.05180 [cs.CR]*.

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

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

Esperança, P. M., Aslett, L. J. M., & Holmes, C. C. (2017). Encrypted accelerated least squares regression. *AISTATs*, 54: 334–43.

Fan, J., & Vercauteren, F. (2012). Somewhat practical fully homomorphic encryption. *IACR Cryptology ePrint Archive*.

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.