

# Random Forests for Big Data

R. Genuer<sup>a</sup>, J.-M. Poggi<sup>b</sup>, C. Tuleau-Malot<sup>c</sup>, N.  
Villa-Vialaneix<sup>d</sup>

<sup>a</sup>Bordeaux Univ. & INRIA Bordeaux

<sup>b</sup>Paris-Sud Univ. & Paris-Descartes Univ.

<sup>c</sup>Nice - Sophia Antipolis Univ.

<sup>d</sup>INRA Toulouse

22 juin 2016

5èmes Rencontres R, Toulouse

# Context

- Random Forests (RF):
  - Popular statistical machine learning method
  - Remarkable performance in a lot of applied problems
- Big Data (BD):
  - Massive, heterogeneous, streaming data
  - Major challenge to analyse those

See [Jordan, \*On statistics, computation and scalability\*, Bernoulli, 2013](#) for a very good introduction to statistics in Big Data

# Big Data characteristics

- The three V (highlighted by Gartner, Inc.) :
  - **Volume**: massive data
  - **Velocity**: data stream
  - **Variety**: heterogeneous data
- Focus on the Volume characteristic in this talk: data are so large that you can not store them on one single computer.

# Strategies for analyzing Big Data

- **Subsampling**: choose a tractable subset of data, perform a classical analysis on it, and repeat this several times (e.g. [Bag of Little Bootstrap](#), [Kleiner et.al. 2012](#))
- **Divide and Conquer**: split the data into a lot of tracktable subsamples, apply classical analysis on each of them, and combine the collection of results (e.g. [MapReduce framework](#))
- **Sequential Updating for Stream Data**: conduct an analysis in an online manner, by updating quantities along data arrival (e.g. [Schifano et.al. 2014](#))

See [Wang et.al. 2015](#) for a good introduction.

# (Motivating) example: Airline data

- Benchmark data in Big Data articles (e.g. [Wang et.al. 2015](#)):  
**124 millions of observations** and **29 variables**
- Aim: predict `delay_status` of a flight using 4 explanatory variables
- Not really massive data: **12 Go** csv file
- Still useful to illustrate some Big Data issues:
  - too large to fit in RAM (of most of nowadays laptops)
  - R struggles as soon as data take less than 10% – 20% of RAM
  - very long computation times to deal with this dataset
- Experiments on a Linux 64 bits server with 8 processors, 32 cores and **256 Go** of RAM

# Notations

$\mathcal{L}_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$  i.i.d. r.v. with the same distribution as  $(X, Y)$ .

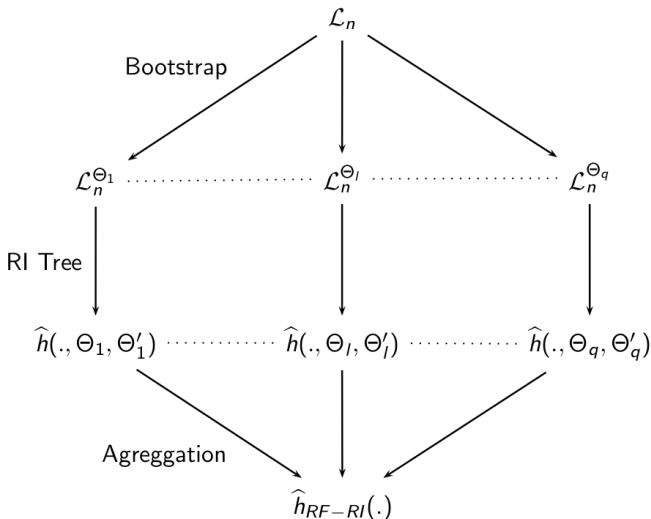
$X = (X^1, \dots, X^p) \in \mathbb{R}^p$  (input variables)

$Y \in \mathcal{Y}$  (response variable)

- $\mathcal{Y} = \mathbb{R}$ : regression
- $\mathcal{Y} = \{1, \dots, L\}$ : classification

**Goal:** build a predictor  $\hat{h} : \mathbb{R}^p \rightarrow \mathcal{Y}$ .

# Breiman's (2001) RF



# RI Tree

Variant of **CART**, **Breiman et.al. (1984)**: piece-wise constant predictor, obtained by a recursive partitioning of  $\mathbb{R}^p$ .

Restriction : splits parallel to axes.

At each step of the partitioning, we search for the “best” split of data among `m` try randomly picked directions.

No pruning.

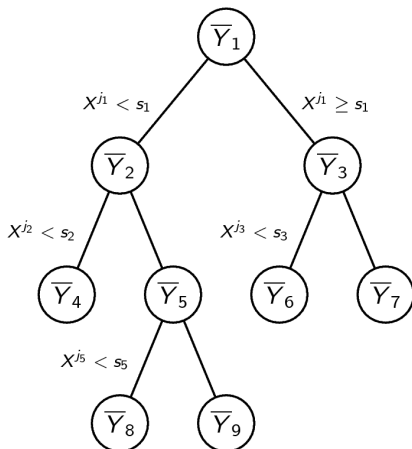


Figure: Regression tree



# OOB error

OOB = Out Of Bag ( $\approx$  "Out Of Bootstrap")

## Out-Of-Bag error

To predict  $X_i$ , we only aggregate trees built on bootstrap samples **which does not contain**  $(X_i, Y_i)$  and get  $\hat{Y}_i$

$\Rightarrow$  OOB error:

- $\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$  regression
- $\frac{1}{n} \sum_{i=1}^n \mathbb{1}_{Y_i \neq \hat{Y}_i}$  classification

# Subsampling and BLB

- Draw a **random subsample** of size  $m$  (which is not trivial in the Big Data context) and build a RF on it. But bias induced by  $m$  out of  $n$  bootstrap arises.
- **Bag of Little Bootstrap (BLB)** : draw bootstrap samples of size  $n$ , each containing only  $m \ll n$  different observations.

# MapReduce RF

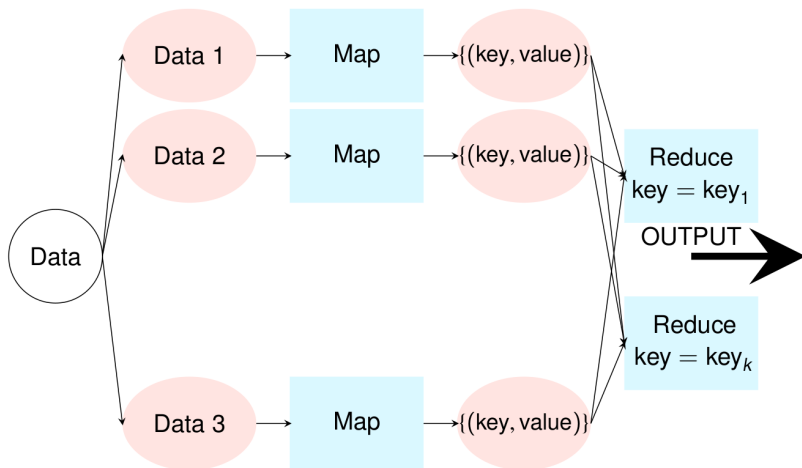


Figure: the *MapReduce* programming paradigm

# Online RF

- Handle **data streams** (data arrive sequentially) in an online manner (no memory of data from the past): **Saffari et.al. 2009**
- Can deal with massive data streams (addressing **both Volume and Velocity** characteristics), but also massive (static) data, by running through the data sequentially
- In depth adaptation of Breiman's RF: even the tree growing mechanism is changed. **Main idea**: think only in terms of **proportions** of output classes, instead of observations
- Consistency results in **Denil et.al. 2013**

# OOB and VI discussion

Keep in mind that in Big Data, all the data can never be accessed entirely at once.

- In **MapReduce RF**: there is no communication between map jobs, so OOB error can not be computed. However it can be approximated by the mean of OOB errors of each map (similarly for variable importance).
- In **Online RF**: OOB error has to be updated each time an observation arrives. It leads to another definition of OOB error (however variable importance calculation is an open issue in this setting).

“Toys data”, Weston *et al.* (2003)

two-class problem,  $Y \in \{-1, 1\}$

6 true variables + noise variables:

- two independent groups of 3 significant variables, related to  $Y$
- an group of noise variables, independent with  $Y$

Model defined through the conditional distributions of the  $X^j$  conditionnally to  $Y = y$ :

- for 70% of data,  $X^j \sim \mathcal{N}(jy, 1)$  for  $j = 1, 2, 3$  and  $X^j \sim \mathcal{N}(0, 1)$  for  $j = 4, 5, 6$
- for the 30% left,  $X^j \sim \mathcal{N}(0, 1)$  for  $j = 1, 2, 3$  and  $X^j \sim \mathcal{N}((j - 3)y, 1)$  for  $j = 4, 5, 6$
- the other variables are noise,  $X^j \sim \mathcal{N}(0, 1)$  for  $j = 7, \dots, p$

# Toys data results

Method	Comp. time	BDerrForest	errForest
<b>sampling 1%</b>	9 sec	4.6	4.4
<b>sampling 0.01%</b>	0.3 sec	4.7	6
<b>BLB-RF 5/20</b>	1 min	4.1	4.3
<b>BLB-RF 10/10</b>	3 min	4.1	4.3
<b>MR-RF 100/1</b>	2 min	14	4.2
<b>MR-RF 10/10</b>	6 min	8.5	4.3

**Table:** Estimated prediction errors (times  $10^3$ ),  $n = 15$  millions. Standard RF give an error of  $4.6e(-3)$  and run in 7 hours.

# Results on permuted toys data

Method	Comp. time	BDerrForest	errForest
<b>unb 100/1/0.1</b>	2 min	5.9	4.5
<b>unb 10/10/0.1</b>	3 min	4.2	4.5
<b>unb 100/1/0.01</b>	1 min	1.9	87
<b>unb 10/10/0.01</b>	4 min	0.9	76
<b>x-biases 100/1</b>	3 min	3.5	101
<b>x-biases 100/10</b>	3 min	2.1	101

**Table:** For "unb" chunks are unbalanced in terms of  $Y$ . For "x-biases" chunks are unbalanced in terms of sub-models of toys data.



# Results on Airline data

Method	Comp. time	BDerrForest	errForest
<b>sampling 1</b>	2 min	18.35	18.33
<b>sampling 0.01</b>	2 sec	18.44	18.49
<b>BLB-RF 15/7</b>	25 min	18.35	18.33
<b>MR-RF 15/7</b>	15 min	18.33	18.27
<b>MR-RF 100/10</b>	17 min	18.33	18.20

**Table:** Estimated prediction errors (times 100),  $n = 124$  millions.  
 Standard RF give an OOB error of  $18.33e(-3)$  and run in 16 hours.

# Perspectives

- Sampling and MapReduce-RF:
  - Use a stratified random subsample
  - Use a partition into map jobs stratified on  $Y$ , or at least a random partition
- Possible variants for MapReduce RF:
  - Use simplified RF, e.g. Extremely Randomized Trees, [Geurts et.al. 2006](#) (as in Online RF)
  - See the whole forest as a forest of forests and adapt the majority vote scheme using weights
- Online RF in practice ([python](#) code available).

# Short bibliography



Breiman, L. *Random Forests*. Machine Learning (2001)



S. del Rio, V. López, J.M. Benítez, and F. Herrera. *On the use of MapReduce for imbalanced big data using random forest*. Information Sciences (2014)



M. Denil, D. Matheson, and N. de Freitas. *Consistency of online random forests*. ICML 2013 (2013)



R. Genuer, J.-M. Poggi, C. Tuleau-Malot, N. Villa-Vialaneix. *Random Forests for Big Data*. Arxiv (2015)



A. Kleiner, A. Talwalkar, P. Sarkar, and M.I. Jordan. *The big data bootstrap*. ICML 2012 (2012)



A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. *On-line random forests*. ICCV Workshops (2009)



C. Wang, M.-H. Chen, E. Schifano, J. Wu, and J. Yan. *A survey of statistical methods and computing for big data*. arXiv (2015)

# Depth of trees

Samp. frac.	Comp. time	Max. size	Pruned size	mean Gini
<b>100%</b>	5 hours	60683	3789	0.233
<b>10%</b>	13 min	6999	966	0.183
<b>1%</b>	23 sec	906	187	0.073
<b>0.1%</b>	0.01 sec	35	10	0.000

Table: Number of tree leaves.

Method	Computational time	errTest
<b>standard</b>	8 hours	3.980e(-3)
<b>sampling 10%</b>	4 min	3.933e(-3)
<b>sampling 1%</b>	10 sec	4.313e(-3)
<b>MR-RF 100/1</b>	2 min	4.033e(-2)
<b>MR-RF 10/10</b>	4 min	3.987e(-3)

Table: Performance obtained using maximal trees.

# Airline data

- Benchmark data in Big Data articles (e.g. [Wang et.al. 2015](#)):  
**124 millions of observations** and **29 variables**
- Aim: predict `delay_status` of a flight using 4 explanatory variables
- Not really massive data: **12 Go** csv file
- Still useful to illustrate some Big Data issues:
  - too large to fit in RAM (of most of nowadays laptops)
  - R struggles as soon as data take less than 10% – 20% of RAM
  - very long computation times to deal with this dataset
- Experiments on a Linux 64 bits server with 8 processors, 32 cores and **256 Go** of RAM

# Airline data with Breiman's RF

- Standard setup, using R and the `randomForest` package (possible thanks to the efficient server at our disposal!)
- 30 min to load data (with `read.table`) and transform data (creation and delation of variables)
- 16 h to grow a RF of **100 trees** with **500 leaves**
- OOB estimate of error rate of 18.37%: performance suffers from the fact that data are unbalanced (**del Rio et.al. 2014**)