Random Forests and Big Data

Robin Genuer [1], Jean-Michel Poggi [2], Christine Tuleau-Malot[3] & Nathalie Villa-Vialaneix[4]

[1] *INRIA, SISTM team & ISPED, INSERM U-897, Univ. Bordeaux, robin.genuer@isped.u-bordeaux2.fr*
[2] *LMO, Univ. Paris-Sud Orsay & Univ. Paris Descartes, jean-michel.poggi@math.u-psud.fr*
[3] *Lab. Jean-Alexandre Dieudonné, Univ. Nice - Sophia Antipolis, malot@unice.fr*
[4] *INRA, UR 0875 MIAT, 31326 Castanet Tolosan cedex, nathalie.villa@toulouse.inra.fr*

Résumé. Le *Big Data* est un des grands défis que doit relever la statistique et a de nombreuses conséquences sur les plans théorique et algorithmique. Le *Big Data* implique toujours le caractère massif des données mais comprend bien souvent aussi des données en flux (en ligne) et implique le traitement de données hétérogènes. Récemment certaines méthodes statistiques ont été adaptées pour traiter le *Big Data*, par exemple les modèles de régression linéaire, les méthodes de classification et les schémas de rééchantillonnage. Basées sur des arbres de décision et exploitant les idées d'agrégation et de bootstrap, les forêts aléatoires introduites par Breiman en 2001, sont une méthode statistique non paramétrique puissante et versatile permettant de prendre en compte dans un cadre unique tant les problèmes de régression que les problèmes de classification binaire ou multi-classes. Ce papier examine les propositions disponibles de forêts aléatoires en environnement parallèle ainsi que sur les forêts aléatoires en ligne. Ensuite, nous formulons diverses remarques avant d'esquisser quelques directions alternatives pour les forêts aléatoires dans le contexte du *Big Data*.

Mots-clés. *Big Data*, Flux de données, Forêts aléatoires

Abstract. Big Data is one of the major challenges of statistical science and has numerous consequences from algorithmic and theoretical viewpoints. Big Data always involves massive data but it also often includes data streams and data heterogeneity. Recently some statistical methods have been adapted to process Big Data, like linear regression models, clustering methods and bootstrapping schemes. Based on decision trees combined with aggregation and bootstrap ideas, random forests, introduced by Breiman in 2001, are a powerful nonparametric statistical method allowing to consider in a single and versatile framework regression problems as well as two-class or multi-class classification problems. This paper reviews available proposals about random forests in parallel environments as well as about online random forests. Then, we formulate various remarks and sketch some alternative directions for random forests in the Big Data context.

Keywords. Big Data, Data streams, Random Forests

1 Introduction

Big Data is one of the major challenges of statistical science and a lot of recent references start to think about the numerous consequences of this new context not only from the algorithmic viewpoint but also scan theoretical implications (see [10]). *Big Data* always involves massive data but it often also includes data streams and data heterogeneity, leading to the famous three V (Volume, Velocity and Variety) highlighted by the Gartner, Inc., the advisory company about information technology research. Recently some statistical methods have been adapted to process *Big Data*, including linear regression models, clustering methods and bootstrapping schemes.

Based on decision trees and combined with aggregation and bootstrap ideas, random forests (RF in the sequel), introduced by Breiman 2001 [1], are a powerful nonparametric statistical method allowing to consider regression problems as well as two-class or multi-class classification problems, in a single and versatile framework. So RF are widely used (see [16, 17] for recent surveys) and exhibit extremely high performance with only a few parameters to tune. Since, RF include intensive resampling and parallel construction of a lot of models (the trees of a given forest), it is natural to think about the interest of using parallel processing and considering massive data streams from a bootstrapping perspective.

The paper is organized as follows: after this introduction, we briefly recall some basics about RF, in Section 2. Then, Section 3 reviews some proposals about RF in parallel environments and highlights some questions. Section 4 develops a similar outline about online RF. Finally, Section 5 contains remarks and proposals.

2 Random Forests

Let $L = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ be a learning set of independent observations of the random vector (X, Y), where $X = (X^1, ..., X^p) \in R^p$ is the vector of the predictors and $Y \in \mathcal{Y}$ is the explained variable, with Y either a class label or a numerical response. The principle of RF is to aggregate many binary decision trees coming from two random perturbation mechanisms: the use of bootstrap samples of L instead of L and the construction of a randomized tree predictor instead of CART [2] on each bootstrap sample. There are two main differences with respect to CART trees: first, in the growing step, at each node, a fixed number of input variables are randomly chosen and the best split is calculated only among them, and secondly, no pruning is performed.

When using RF, a quantity of interest is the out-of-bag (OOB) error, which allows to quantify the variable importance (VI in the sequel). The quantification of the variable importance is crucial for many procedures involving RF, *e.g.*, for ranking the variables before a stepwise variable selection strategy (see [7]). For each tree t of the forest, consider the associated OOB_t sample (composed of data not included in the bootstrap sample used to construct t). The OOB error rate is defined, in the classification case, by errOOB =

$\frac{1}{n}\text{Card}\,\{i \in \{1,\ldots,n\} \mid y_i \neq \hat{y}_i\}$, where $\hat{y}_i$ is the most frequent label predicted by trees t for which (x_i, y_i) is in the associated OOB_t sample.

Denote by errOOB_t the error (MSE for regression and misclassification rate for classification) of tree t on its associated OOB_t sample. Now, randomly permute the values of X^j in OOB_t to get a perturbed sample denoted by $\widetilde{\text{OOB}_t}^j$ and compute $\text{err}\widetilde{\text{OOB}_t}^j$, the error of tree t on the perturbed sample. The variable importance of X^j is then equal to $\text{VI}(X^j) = \frac{1}{\text{ntree}} \sum_t (\text{err}\widetilde{\text{OOB}_t}^j - \text{errOOB}_t)$, where the sum is over all trees t of the RF and ntree denotes the number of trees of the RF.

3 Random forests in parallel environments

As explained in the introduction, RF are based on the definition of several independent trees. It is thus straightforward to obtain a parallel and faster implementation of the RF method, in which several trees are built in parallel. However, in the *Big Data* framework, this is not enough: the amount of data to be processed is often even too large to be performed by a single processor. One approach to deal with data which size exceeds the computing capability of a single core is to rely on the *MapReduce* (MR) programming paradigm. MR proceeds in two steps: in a first step, called the *Map* step, the data set is split into several smaller chunks of data, each one being processed by a separate core: these different map jobs are independent and produce a list of (key, value), where "key" is a key indexing the data contained in "value". Then, in a second step, called the *Reduce* step, each reduce job proceeds all data corresponding to a given key value.

As indicated in [4], the standard MR version of RF, denoted by MR-RF in the remaining of this paper, is based on the parallel construction of trees obtained on bootstrap *subsamples* of the data: each chunk of data is sent to an independent map job in which a RF is built. There is no reduce job as the output of the map jobs is a collection of RF which, all merged together, give the final forest. This adaptation of RF to the MR framework has several drawbacks: 1) as noted by [12], data are rarely ordered randomly in the *Big Data* world: items clustered on some particular attributes are often placed next to each others on disk due to spatial locality. 2) The samples sent to every different map job might well be specific enough to produce very heterogeneous forests. There would be no meaning in simply averaging all those trees together to make a global prediction. 3) As pointed out by [11], another limit of the approach comes from the fact that the success of m-out-of-n bootstrap samples is highly conditioned on the choice of m, which can not be well controlled or tuned within a MR implementation. 4) Given that the map jobs are processed independently and only have access to a (small) part of the data, the OOB error - and thus, the VI - cannot be obtained directly from the building of the forest.

4 Online random forests

General idea of online RF (ORF in the sequel) is to adapt RF methodology, in order to handle the case where data arrive sequentially (also known as a "data stream"). An online framework suppose that at time t, one does not have access to all the data from the past, but only to the current observation. ORF are first defined in [15] and detailed only for classification problems. They combine the idea of online bagging, from [14], Extremely Randomized Trees (ERT), from [8], and a mechanism to update the forests each time a new observation arrives.

Each time a new data arrives, the online bagging updates k times a given tree, where k is sampled from a Poisson distribution to mimic a batch bootstrap sampling. ERT is used instead of original Breiman's RF, because it allows for a faster update of the forest: in ERT, S splits are randomly drawn for every node, and the final split is optimized only on those S candidate splits. Finally, all decisions made by a tree are only based on the proportions of each class label among observations in a node. ORF keep up-to-date an heterogeneity measure (based on these statistics) online. This measure determines the class label of a node. When a new data arrives in a node, it is updated for all existing leaves and for all potential new leaves created by the several candidate splits.

OOB error rate of a tree t is also estimated online: the current data is OOB for all trees for which the Poisson random variable used to replicate the observation in the tree, is equal to 0. The prediction provided for such a tree t is used to update OOB_t. However, as the prediction cannot be re-evaluated after the tree has been updated with next data, this approach is only an approximation of the original OOB_t.

Finally, the recent article [5] introduces a new variant of ORF. The two main differences with the original ORF are that, 1) no online bootstrap is performed. 2) The data stream is randomly partitioned in two streams, the structure stream and the estimation stream. Data from structure stream only participate on the splits optimization, while data from estimation stream are only used to allocate a class label to a node. Thanks to this partition, authors manage to obtain consistency results.

5 Perspectives

This section's purpose is to provide a brief discussion about possible alternatives to overcome the limits of the existing solutions that we have described previously.

As explained in Section 2, **OOB error** and **VI** are important diagnostic tools but these quantities may be unavailable directly in the RF variants described above. First, due to the independence of the different map jobs, the OOB error cannot be computed in the RF-MR variant: each forest trained in a map job is unaware of the data which have not be sent to its map and the indexes of the data that have been used to train a given forest are lost in the output of the map jobs so there is no way to obtain the OOB error in the sense that is given in Section 2. However, an approximation of this error can

be obtained restricting its computation to a given chunk of data and averaging them all. The ORF provides an estimation of errOOB_t, as described in Section 4. This estimation could be used to update online an estimation of errOOB. However, permuting the values of a given variable when the data arrive in stream and are not stored after they have been processed is still an open issue for which [15, 5] give no solution. Hence, VI cannot be simply defined in the online framework.

The first limit of MR-RF that has been pointed in Section 3 is the **possible sampling bias** coming from the split of data in several chunks for the parallel map jobs. This could be addressed with a more careful design of the data partition, using at least a random partition of the data or, probably more efficiently, a stratification performed on the explained variable values. [12, 9, 13] propose methods to subsample the data in a *Big Data* context, some of these methods allowing for stratified subsampling or subsampling that satisfies a given predicate. Using these methods prior to RF or MR-RF could help to overcome the problem of the splitting bias. An interesting alternative to a simple random subsample would be to use the BLB (Bag of Little Bootstrap) method described in [11], which aims at building bootstrap samples of size n, each one containing only $m \ll n$ different data. The size of the bootstrap sample is the classical one (n), thus avoiding the problem of the m-out-of-n bootstrap method, but the processing of this sample is simplified and manageable even for very large n because it contains only a small fraction (m/n) of the original data set.

Finally, we conclude with a few proposals for **alternative variants** to MR-RF. First, the computational cost of building a single tree could be reduced: many simplifications for the construction of trees are available. The idea is not to choose the variable involved in a split and the associated threshold from the data but to randomly choose them according to different schemes (see [3] for PERT, Perfect Random Tree Ensembles, [8] for ERT, Extremly Randomized Trees, and [6] for PRF, Purely Random Forests). Another possible variant would be to consider the whole forest as an ensemble of forests and to adapt the majority vote scheme with weights that address, *e.g.*, the issue of the sampling bias. Finally, another direction would be to use ORF in the *Big Data* framework, addressing both the issue of Volume and Velocity: using an update scheme, instead of the building forest in parallel until all data have been processed, could be a way to use only a portion of the data set until the forest is accurate enough.

References

[1] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[2] L. Breiman, J. Friedman, R. Olsen, and C. Stone. *Classification and Regression Trees*. Chapman and Hall, New York, USA, 1984.

[3] A. Cutler and G. Zhao. Pert-perfect random tree ensembles. *Computing Science and Statistics*, 33:490–497, 2001.

[4] S. del Rio, V. López, J.M. Beniítez, and F. Herrera. On the use of MapReduce for imbalanced big data using random forest. *Information Sciences*, 285:112–137, 2014.

[5] M. Denil, D. Matheson, and N. de Freitas. Consistency of online random forests. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, pages 1256–1264, 2013.

[6] R. Genuer. Variance reduction in purely random forests. *Journal of Nonparametric Statistics*, 24(3):543–562, 2012.

[7] R. Genuer, J.M. Poggi, and C. Tuleau-Malot. Variable selection using random forests. *Pattern Recognition Letters*, 31(14):2225–2236, 2010.

[8] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[9] R. Grover and M.J. Carey. Extending map-reduce for efficient predicate-based sampling. In *Proceedings of IEEE International Conference on Data Engineering (ICDE 2012)*, pages 486–497, Washington, DC, USA, 2012.

[10] M.I. Jordan. On statistics, computation and scalability. *Bernoulli*, 19(4):1378–1390, 2013.

[11] A. Kleiner, A. Talwalkar, P. Sarkar, and M.I. Jordan. The big data bootstrap. In *Proceedings of 29th International Conference on Machine Learning (ICML 2012)*, Edinburgh, Scotland, UK, 2012.

[12] N. Laptev, K. Zeng, and C. Zaniolo. Early accurate results for advanced analytics on mapreduce. In *Proceedings of the 28th International Conference on Very Large Data Bases*, volume 5 of *Proceedings of the VLDB Endowment*, Istanbul, Turkey, 2012.

[13] X. Meng. Scalable simple random sampling and stratified sampling. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, volume 28 of *JMLR: W&CP*, Georgia, USA, 2013.

[14] N.C. Oza. Online bagging and boosting. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2340–2345. IEEE, 2005.

[15] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In *Proceedings of IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1393–1400. IEEE, 2009.

[16] A. Verikas, A. Gelzinis, and M. Bacauskiene. Mining data with random forests: a survey and results of new tests. *Pattern Recognition*, 44(2):330–349, 2011.

[17] A. Ziegler and I.R. König. Mining data with random forests: current options for real-world applications. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(1):55–63, 2014.