Random Uniform Forests

Saïp Ciss

▶ **To cite this version:**

Saïp Ciss. Random Uniform Forests. 2015. hal-01104340v1

HAL Id: hal-01104340
https://hal.science/hal-01104340v1

Preprint submitted on 16 Jan 2015 (v1), last revised 19 Jan 2015 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Public Domain

Random Uniform Forests

Saïp Ciss*

January 14, 2015

Abstract

Random Uniform Forests are a variant of Breiman's *Random Forests* (tm) (Breiman, 2001) and *Extremely randomized trees* (Geurts et al., 2006). Random Uniform Forests are designed for classification, regression and unsupervised learning. They belong to the family of *ensemble learning* and build many unpruned and randomized binary decision trees then use averaging (regression) or majority vote (classification) to take a decision. Unlike Random Forests, they draw *random cut-points, using the continuous Uniform distribution* for each node (region) to grow each tree. Unlike Extremely randomized trees, they use *bootstrap* (only for classification) and *subsampling*, since *Out-of-bag (OOB)* modeling plays a key role. Unlike both algorithms, for each node *sampling with replacement* is done to select features. Random Uniform Forests are aimed to get low correlated trees, to allow a deep analysis of variable importance (Ciss, 2015b) and to be natively incremental. *Random uniform decision trees* are the core of the model. We provide an R package, *randomUniformForest*, and present main theoretical arguments. The algorithm follows and extend Breiman's key idea : increase diversity to build uncorrelated trees. Hence the main motivation of Random Uniform Forests is to be more weakly dependent to the data than Random Forests while giving similar performance and inheriting of all their theoretical properties.

Keywords : Random Uniform Forests, Random Forests, statistical learning, machine learning, ensemble learning, classification, regression, R package.

*PhD. University Paris Ouest Nanterre La Défense. Modal'X. saip.ciss@wanadoo.fr

1 Introduction

Random Uniform Forests belong to the family of ensemble models that build many base learners then combine them to achieve tasks like classification, regression or unsupervised learning. They are first designed to be less computationally expensive than Breiman's Random Forests while keeping all properties of the latter. They also follow the idea of Extremely Randomized Trees (Extra-Trees) but, here, we do not want to loose the OOB (Out-of-bag) evaluation. Implementations of ensemble learning are widespread, using different manners; however we will be focused on Random Forests and Extra-Trees since they provide efficient algorithms that can be used in R (R core Team, 2014) for real life problems and are close to Random Uniform Forests for their decision rule. Base learners are decision trees, like the CART (Breiman et al., 1984) paradigm, but in Random Uniform Forests they are unpruned and not deterministic.

In Random Uniform Forests, we seek strong randomization and some kind of global optimization. Hence, we want to achieve low correlation between trees (or trees residuals) leaving average variance of trees, eventually, increase but not to much. We are not concerned by bias, since we assume that if trees are enough randomized, ensemble model should have low bias. If not, post-processing functions are implemented to reduce bias. So, conceptually, we build the model using the following steps.
i) For classification, we first draw, for each tree and with replacement, n observations among the n of the training sample (the *bootstrap*). For regression, we use *subsampling* (drawing, without replacement, m points out-of n, $m < n$).
ii) Each tree is grown by sampling randomly and *with replacement* a set of variables, for the candidate nodes (regions) at each step. It means that, if the dimension of the problem is d, one can choose $\lceil \beta d \rceil$ variables, $\beta \geq 1/d$, for the candidate nodes to grow the tree.
iii) Each cut-point is generated randomly, according to *the continuous Uniform distribution on the support of each candidate variable* or, more computationally efficient and for classification, between two random points of each candidate variable. Hence, we do not use any local optimization to find the best cut-point. Each one follows exactly the continuous Uniform distribution and, so, is not exactly one point among the ones in the node.
iv) Optimal random node is, then, selected by maximizing *Information Gain* (classification) or minimizing *'L2' (or 'L1') distance* (regression). For the classification case, current and candidate nodes are used to compute Information Gain while in regression, only candidates nodes are used to compute distances.

Main arguments reside in drawing with replacement variables, using the continuous Uniform distribution and choosing the optimization criterion. However, at the algorithmic level, some others arguments are used (e.g. for categorical variables or for removing useless variables in the case of sparse data). In both Extra-Trees and Random Forests, there is no replacement when selecting variables. Extra-Trees use random cut-point, which is drawn uniformly between the minimal and the maximal value of each candidate variable for the current node. Like Random Forests, Random Uniform Forests use bootstrap for classification, but in the case of regression, subsampling is preferred, giving better results

than bootstrap. All three algorithms use a different criterion to find the optimal cut-point and variable, for each node, between candidates.
In the next section we focus on theoretical arguments of *random uniform decision trees*, the base learners.
In section 3, we provide the details of Random Uniform Forests.
In section 4, we provide benchmarks on many datasets and compare the results with, both Extra-Trees and Random Forests, an some others state-of-the-art algorithms. In section 5, we conclude our analysis.

2 Random uniform decision trees

Random uniform decision trees are unpruned and randomized binary decision trees. They use the continuous Uniform distribution to be built (that gives them their name) and, unlike CART, do not seek optimality. They, first, used to be the core of Random Uniform Forests and just growing one random uniform decision tree is useless (in comparison to CART). But understanding Random Uniform Forests needs first to know mechanisms of the former. A binary decision tree is usually grown by recursively partitioning the data in two regions until some stopping rules are met. Then, a decision rule is applied to the current region. One needs to know, usually, three aspects :
- how to grow the tree and choose a node,
- when to stop growing tree,
- how to define and build the decision rule.

In the CART paradigm, for each candidate region, one chooses each coordinate (variable) in the data and find the best cut-point (looking for all observations) that minimizes an optimization criterion. Then, for all coordinates, the optimal value of the criterion leads to the coordinate and associated cut-point that will be chosen to partition the current node. In Breiman's Random Forests, a few coordinates are first randomly and uniformly chosen. Then observations are drawn randomly with replacement, *the Bootstrap* (Efron, 1979), before applying the CART paradigm. In Extremely randomized trees, there is no bootstrap and both coordinates (a few again) and cut-points are chosen randomly. Hence, only the last part of the CART paradigm is applied. We follow the same principle in Random Uniform Forests, but make modifications in each part of the procedure when growing a Random Unifom Decision Tree.

`Definition.` *A random uniform decision tree is a binary decision tree in which nodes are built using random cut-points. For each step of the recursive partitioning,* $\lceil \beta d \rceil$ *variables,* $\beta \geq 1/d$*, are drawn with replacement. Then for each candidate variable, a cut-point* α *is drawn using the continuous Uniform distribution on the support of each candidate or between two random points of the latter. Optimal random node is the one that maximizes information gain (in classification) or that minimizes a* $\boldsymbol{L}^2$ *distance (or another one) in regression. The recursive partitioning is pursued unless a stopping rule is matched. The decision rule is then applied and exists only for terminal nodes.*

2.1 Regions

Random uniform decision trees are close to other types of decision trees and a node has always two or zero children nodes. We first have to define what a region is and for that, we call $\mathcal{P}$, a partition of the data. Following Devroye et al. (1996), we propose this definition.

`Definition.` *A is a region of the partition $\mathcal{P}$ if, for any $B \in \mathcal{P}, A \cap B = \emptyset$ or $A \subseteq B$.*

Hence, we suppose that we have $D_n = \{(X_i, Y_i), 1 \leq i \leq n\}$, corresponding to the observations and responses of the training sample, where (X, Y) is a $\mathbb{R}^d \times \mathcal{Y}$-valued random pair, with respect to the *i.i.d.* assumption. A is an *optimal region* of the random uniform decision tree if :

$$for\ any\ A \in \mathcal{P}, \left\{X_i^{(j^*)} \leq \alpha_{j^*} | D_n\right\},\ 1 \leq j \leq d, 1 \leq i \leq n,$$
$$for\ any\ A^C \in \mathcal{P}, \left\{X_i^{(j^*)} > \alpha_{j^*} | D_n\right\},\ 1 \leq j \leq d,\ 1 \leq i \leq n,$$

where, for classification :
$\alpha_j \sim \mathcal{U}\left(\texttt{min}(X^{(j)}|D_n), \texttt{max}(X^{(j)}|D_n)\right)$ and $j^* = \underset{j \in \{1,\dots,d\}}{\texttt{arg max}}\ \texttt{IG}(j, D_n)$,
and for regression :
$\alpha_j \sim \mathcal{U}\left(\texttt{min}(X^{(j)}|D_n), \texttt{max}(X^{(j)}|D_n)\right)$ and $j^* = \underset{j \in \{1,\dots,d\}}{\texttt{arg min}}\ \texttt{L}_2(j, D_n)$,
where `IG` is the Information Gain function, $\texttt{L}_2$, an Euclidean distance function, and are both defined in section 2.3.

2.2 Stopping rules

Once A and A^C, its complementary region, are found, we repeat the recursive partitioning for the two regions (randomly drawing variables and cut-points, and choosing the optimal region) until we met some conditions, which are :
- the minimal number of observations is reached (usually one observation),
- for one region, all the observations have the same label (or value),
- for one region, all the observations are the same,
- for one region, there is no more variables to select (since the algorithm can delete any or many of them internally),
- $\texttt{IG}(j, D_n)$ (or, in regression, $\texttt{L}_2(j, D_n)$) reached a threshold (usually 0).

2.3 Optimization criterion

It remains to define the `IG` function and a *decision rule* for the tree. For classification, let us suppose, for simplicity, that $Y \in \{0, 1\}$. We have :

$$\texttt{IG}(j, D_n) = \texttt{H}(Y|D_n) - \left[\texttt{H}\left(\left(Y|X^{(j)} \leq \alpha_j\right)|D_n\right) + \texttt{H}\left(\left(Y|X^{(j)} > \alpha_j\right)|D_n\right)\right],$$

where `H` is the *Shannon entropy* (note that we use it with the natural logarithm),
and

$$\texttt{H}(Y|D_n) = -\sum_{c=0}^{1}\left\{\frac{1}{n}\sum_{i=1}^{n}\mathbf{I}_{\{Y_i=c\}}\ \texttt{log}\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{I}_{\{Y_i=c\}}\right)\right\},$$

with, by definition, $0 \log 0 = 0$, so that $\mathtt{H}(Y) \geq 0$.

Let $n^{'} = \sum_{i=1}^{n} \mathbf{I}_{\left\{X_i^{(j)} \leq \alpha_j\right\}}$, then

$$\mathtt{H}\left(\left(Y|X^{(j)} \leq \alpha_j\right)|D_n\right) = -\frac{n^{'}}{n} \sum_{c=0}^{1} \left\{ \frac{1}{n^{'}} \sum_{i=1}^{n} \mathbf{I}_{\{Y_i=c\}} \mathbf{I}_{\left\{X_i^{(j)} \leq \alpha_j\right\}} \log \left(\frac{1}{n^{'}} \sum_{i=1}^{n} \mathbf{I}_{\{Y_i=c\}} \mathbf{I}_{\left\{X_i^{(j)} \leq \alpha_j\right\}} \right) \right\},$$

and

$$\mathtt{H}\left(\left(Y|X^{(j)} > \alpha_j\right)|D_n\right) = \\ -\frac{n-n^{'}}{n} \sum_{c=0}^{1} \left\{ \frac{1}{n-n^{'}} \sum_{i=1}^{n} \mathbf{I}_{\{Y_i=c\}} \mathbf{I}_{\left\{X_i^{(j)} > \alpha_j\right\}} \log \left(\frac{1}{n-n^{'}} \sum_{i=1}^{n} \mathbf{I}_{\{Y_i=c\}} \mathbf{I}_{\left\{X_i^{(j)} > \alpha_j\right\}} \right) \right\}.$$

For regression, we define $\mathtt{L}_2(j, D_n)$ by

$$\mathtt{L}_2(j, D_n) = \sum_{i=1}^{n} \left(Y_i \mathbf{I}_{\left\{X_i^{(j)} \leq \alpha_j\right\}} - \hat{Y}_A \mathbf{I}_{\left\{X_i^{(j)} \leq \alpha_j\right\}} \right)^2 + \sum_{i=1}^{n} \left(Y_i \mathbf{I}_{\left\{X_i^{(j)} > \alpha_j\right\}} - \hat{Y}_{A^C} \mathbf{I}_{\left\{X_i^{(j)} > \alpha_j\right\}} \right)^2,$$

with

$$\hat{Y}_A = \frac{1}{n'} \sum_{i=1}^{n} Y_i \mathbf{I}_{\left\{X_i^{(j)} \leq \alpha_j\right\}} \quad and \quad \hat{Y}_{A^C} = \frac{1}{n-n'} \sum_{i=1}^{n} Y_i \mathbf{I}_{\left\{X_i^{(j)} > \alpha_j\right\}}.$$

2.4 Decision rule

Each time a stopping criterion is met, we define the decision rule, $g_{\mathcal{P}}$, for an optimal (and terminal) node A. We have for binary classification :

$$g_{\mathcal{P}}(x, A, D_n) = g_{\mathcal{P}}(x) = \begin{cases} 1, \ if \ \sum_{i=1}^{n} \mathbf{I}_{\{X_i \in A, Y_i=1\}} > \sum_{i=1}^{n} \mathbf{I}_{\{X_i \in A, Y_i=0\}}, \ x \in A \\ \\ 0, \ otherwise. \end{cases}$$

And for regression :

$$g_{\mathcal{P}}(x, A, D_n) = g_{\mathcal{P}}(x) = \frac{1}{\sum_{i=1}^{n} \mathbf{I}_{\{X_i \in A\}}} \sum_{i=1}^{n} Y_i \mathbf{I}_{\{X_i \in A\}}, \ x \in A.$$

2.5 Categorical variables

The description we give above work well for numerical variables but one may wonder how to treat categorical variables which usually have no order in the values they take. One may use dummies, coding each categorical variable as many binary ones, but it leads to increase the dimension of data and may not be suitable for the algorithm. In Random Uniform Forests, categorical variables are treated at the node level. More precisely, the algorithm selects for each candidate node, randomly and before the splitting process, two values of the categorical variable. The first one keeps its positions within the variable while the second replaces, temporarily all others values. This leads to a binary variable

that can be treated like a numerical one. After the splitting, the variable recovers its original values. Since cut-points are almost virtual and random (a cut-point is usually not an observed point of the training sample), one has just to take care that the random binarization would not weaker the variable. This is avoidable, since all nodes of all trees are treated in a random manner and any variable can be selected many times in each node (the *sampling with replacement variables* procedure).

2.6 Algorithm

We can summarize the different tasks mentioned above by the following algorithm :

1- draw at random and with replacement (or subsample, m out-of n, in case of regression) n observations of D_n.
a) select at random and with replacement $\lceil \beta d \rceil$ variables,
b) for each of the $\lceil \beta d \rceil$ variables, draw α using the continuous Uniform distribution on the support of each candidate variable or between two random points,
2- for the $\lceil \beta d \rceil$ variables, choose the pair (j^*, α_{j^*}) that maximizes $\texttt{IG}(j, D_n)$ for classification, and for regression the one that minimizes $\texttt{L}_2(j, D_n)$,
3- (j^*, α_{j^*}) is the pair that defines the regions A and A^C,
4- If a stopping rule is met, stop the partitioning and build the decision rule $g_{\mathcal{P}}$,
5- If not, pursue step 1 to 5 for A and A^C.

In a random uniform decision tree, partitioning leads to a large and deep tree. If balanced, maximal depth is $\texttt{log}(n)/\texttt{log}(2)$ which is usually obtained for regression, while for classification the depth is usually lower. Since no pruning is done, terminal nodes tend to have a very few values leading to a high variance of the tree. Hence, in a random uniform decision tree prediction error is not an achievement but high variance is, as a condition for getting low correlation between trees. To avoid it increase too much, perturbations on the training sample sent to the optimization criterion are the main argument. The second one is that no local optimization is done since we want to lower the correlation between trees: following Breiman's ideas, as variance for a single tree is high and hard to reduce, one can let it get high (introducing more diversity) and let ensemble do the reduction using the Law of Large Numbers, if trees are independent. In practice, they are not and one can use correlation to measure the level of dependence. Comparing to CART, one single random uniform decision tree will have an higher variance, but average variance of trees will be close, if not lower, in most cases than CART variance. Then, next step is to know how averaging will affect prediction error.

3 Random Uniform Forests

As Random Forests do, we use ensemble of random uniform decision trees to build a Random Uniform Forest. Algorithm is straightforward but one can note that many improvements come from the algorithmic level where a lot of randomness is essential to get improvements over a simple averaging of trees. Another point of view is that Random Uniform Forests use the Bayesian paradigm : *the forest classifier is the result of almost all (trees) parameters for a fixed data set.*

3.1 Algorithm

Once the structure of a random uniform decision tree is known, algorithm needs a few lines (but much more in practice) :
1- For each tree, from $b = 1$ to B, grow a random uniform decision tree and build its decision rule
2- For the B trees, apply the rule $\bar{g}_{\mathcal{P}}^{(B)}$ (see section 3.3).

3.2 Key concepts

In Random Forests, a tree is grown by choosing, at each step of the recursive partitioning, the optimal region among a few locally optimal, but randomized, regions. In Random Uniform Forests, the optimal region is chosen among many, possibly overlapping, random regions. In Extremely Randomized Trees, the optimal region is chosen among a few non-overlapping random regions. In all cases, guarantees are due to the Law of Large Numbers that generates convergence and needs trees to be theoretically independent.
The main difference with Random Forests appears in *average trees correlation* which is usually higher in the latter (at least for regression) while the *average variance of trees* is smaller. The motivation of pursuing low correlation is directly linked with the Law of Large Numbers and theoretical properties of Random Forests, and Random Uniform Forests, that we want to be the closest to the practice. In other words, Random Uniform Forests are an application of Breiman's ideas using (strong) randomization to the point of view of observations rather than to the dimension whose we need, here, more for optimization.

3.3 Decision rule

Let us write $g_{\mathcal{P}}(X) \stackrel{def}{=} g_{\mathcal{P}}(X, \theta)$, where θ is the parameter that translates the randomness introduced in the tree. For the b-th tree, $1 \leq b \leq B$, we have $g_{\mathcal{P}}^{(b)}(X) \stackrel{def}{=} g_{\mathcal{P}}(X, \theta_b)$. For an ensemble of trees, the decision rule, $\bar{g}_{\mathcal{P}}^{(B)}$, is easy to write. We have for binary classification :

$$\bar{g}_{\mathcal{P}}^{(B)}(x) = \begin{cases} 1, \ if \ \ \sum_{b=1}^{B} \mathbf{I}_{\{g_{\mathcal{P}}^{(b)}(x)=1\}} > \sum_{b=1}^{B} \mathbf{I}_{\{g_{\mathcal{P}}^{(b)}(x)=0\}} \\ \\ 0, \ otherwise. \end{cases}$$

And for regression :

$$\bar{g}_{\mathcal{P}}^{(B)}(x) = \frac{1}{B} \sum_{b=1}^{B} g_{\mathcal{P}}^{(b)}(x).$$

The decision rule is simple and one can ask how to find interesting properties and how to explain good results of ensemble models. Trees in Random (Uniform) Forests are designed to be weakly dependent in order to apply most main theoretical properties provided by Breiman and Random Uniform Forests simply inherit from them. In fact, the most important aspect is the simplification introduced when growing trees. This leads to many, new or updated, applications like procedure for prediction and confidence intervals, missing values imputation, many measures of variable importance and deep analysis, or

incremental learning. From the theoretical side, *the main argument of Random Uniform Forests is to lower correlation between trees faster or as fast as the increase of average variance of trees introduced by the strong randomization.* To be more precise :
- in classification, the theoretical prediction error is bounded by the product of average correlation between trees and a function of the squared strength (or margin), which is the difference, in frequency, between observations that are classified correctly and misclassified observations over all trees. Since, for binary classification, the misclassification rate can be written as a function of (among others terms) the parameter of the class distribution, the study of bias is the main object to assess when lowering correlation. But, since Random Uniform Forests converge (hence strength reaches is theoretical value) the main practical consequence of lowering correlation is the number of observations needed to get an optimal error, with respect to the training sample structure.
- In regression, the theoretical prediction error is bounded by the product of average correlation between trees residuals and the average prediction error of trees, which depends to the average variance of trees residuals. Hence, if one supposes that the bias can not be reduced, then lowering correlation implies to control variance.

3.4 The use of random cut-points

For practitioners, the use of random cut-points may seem non optimal since variance (stated as average variance of trees residuals) will get higher. The first essential point is that as the same time correlation (stated as average correlation between trees residuals) will decrease faster than most of others splitting rules. Since *low dependence between trees is a necessary condition for convergence*, if this latter happens for others splitting rules it will also happen, possibly slowly, when using random cut-points. In fact, random cut-points ensure that conditions for convergence are provided. However, variance will still not be controlled, especially if we let the trees growing to their maximum size.

3.5 Sampling, with replacement, features

To overcome issues with a possible increasing variance, by using random cut-points, one needs to introduce techniques that do not affect correlation but lead to control variance. Unfortunately, there is not a unique one on which we can count every time. For regression, subsampling is, empirically, a good choice. But the most important is to use the dimension of the problem. It is done by sampling with replacement features for each node. Hence, competition between (fully random) candidate nodes increases with the number of selected features, and the duplication of some, while variance is reduced up to a point. Empirically, the main effects reside in the following steps :
i) since Random Uniform Forests do not have bias around the mean over all trees, correlation can be easily decreased,
ii) using random cut-points, with other techniques, allows to decrease correlation as fast as possible but increases variance,
iii) sampling, with replacement, features, allows to decrease variance up to a point, without touching correlation, or can decrease both variance and correlation.

In Extra-trees, there is not sampling with replacement when selecting candidate variables.

Hence, more work is done on the optimization criterion in order to reduce variance, also leading to a slightly increase of correlation, in comparison to Random Uniform Forests. In Breiman's Random Forests, the correlation is usually higher, since the main source of randomness comes from sampling, without replacement, a few coordinates at each node. average variance of trees residuals is reduced since cut-points are not random with respect to a candidate variable.

The main point and source of improvements is to know how to find a way to further reduce average variance of trees residuals with, in all cases, a low (or monotonically decreasing) average correlation between trees residuals. Most of the possible solutions reside on the algorithmic side where a corollary seems to make one case (decrease variance or decrease correlation) arise (much) faster than the other (the increase of one of them).

3.6 The OOB classifier

The Out-of-bag (OOB) informations are the observations that do not participate to the trees growth. For each tree, due to bootstrap or subsampling, some observations are not chosen and are stored in order to build the OOB classifier whose decision rule is $\bar{g}^{(B)}_{\mathcal{P},oob}(X)$. *The OOB classifier exists only for the training sample* and use B' trees, $B' \simeq \texttt{exp}(-1)B$, with n observations. Note that the B' trees are not necessary the same for each observation that needs to be evaluated.
We have for an observation x and for only D_n,

$$\bar{g}^{(B)}_{\mathcal{P},oob}(x) = \begin{cases} 1, \ if \ \sum_{b=1}^{B} \mathbf{I}_{\left\{g^{(b)}_{\mathcal{P}}(x)=1\right\}} \mathbf{I}_{\left\{b \in G^{-}(x,B)\right\}} > \sum_{b=1}^{B} \mathbf{I}_{\left\{g^{(b)}_{\mathcal{P}}(x)=0\right\}} \mathbf{I}_{\left\{b \in G^{-}(x,B)\right\}} \\ \\ 0, \ otherwise. \end{cases}$$

And, for regression :

$$\bar{g}^{(B)}_{\mathcal{P},oob}(x) = \frac{1}{\sum_{b=1}^{B} \mathbf{I}_{\left\{b \in G^{-}(x,B)\right\}}} \sum_{b=1}^{B} g^{(b)}_{\mathcal{P}}(x) \mathbf{I}_{\left\{b \in G^{-}(x,B)\right\}},$$

where $G^{-}(x, B)$ is the set of trees, among the B, which have never classified x.

The OOB classifier gives an estimate of prediction error and lead to many improvements, like post-processing, in order to control prediction error or for others purposes. One of the most important is a way to prevent overfitting, using the OOB classifier in conjunction with the Breiman's bounds.

3.7 Incremental learning

Incremental learning is a way to do learning for streaming data. If data are very large or come by chunks, then incremental learning is one way to treat them whatever their size is. But, it will usually have a cost, at least a (slight or large) loss of accuracy. Incremental learning proceeds by chunks of data and the regression case is usually less easy than classification for achieving low prediction error (in contrast of a computation of all the

data at once). Incremental learning has two cases :
- the *i.i.d.* assumption holds for all chunks (or, at least, many),
- the (joint) distribution of (X, Y) is shifting (either the distribution itself or its parameters or the target variable).

Random Uniform Forests are natively incremental. We consider the *i.i.d* case, but it also applies to the *non-i.i.d.* one by considering sub-forests. Let us call $\bar{g}^{(T)}_{\mathcal{P},inc}$ the *incremental forest classifier*. We simply have

$$\bar{g}^{(T)}_{\mathcal{P},inc}(x) = \begin{cases} 1, \ if \ \sum_{t=1}^{T}\sum_{b=1}^{B_t} \mathbf{I}_{\left\{g^{(b)}_{\mathcal{P}_t}(x)=1\right\}} > \sum_{t=1}^{T}\sum_{b=1}^{B_t} \mathbf{I}_{\left\{g^{(b)}_{\mathcal{P}_t}(x)=0\right\}} \\ \\ 0, \ otherwise. \end{cases}$$

And for regression,

$$\bar{g}^{(T)}_{\mathcal{P},inc}(x) = \frac{1}{\sum_{t=1}^{T} B_t} \sum_{t=1}^{T}\sum_{b=1}^{B_t} g^{(b)}_{\mathcal{P}_t}(x),$$

where $\mathcal{P}_t$ is a partition of the data for the slice time t,
B_t, is the number of trees for the slice time t,
T, is the number of slices time.

Incremental learning is a subsampling process that we apply on both data and decision rule. Each tree sees only a part of the data and the forest itself sees a (bigger) part. The main argument here is that cut-points are random, so see a part or whole data does not change, in practice, many things in the *i.i.d.* case. Moreover, for very large datasets, one will never be able to compute all data at once. The problem relies more on the informations retained. Some informations that are important for a slice time, can be obsolete in the next slice time or worst, leading to confuse the forest. The only hope is, then, to adapt the classifier to the new situation without loosing all the informations memorized before. Since a sub-forest is a forest, with just less trees (like the OOB classifier is), we can adapt Random Uniform Forests by assessing $\bar{g}^{(T)}_{\mathcal{P},inc}$, $\bar{g}^{(T-1)}_{\mathcal{P},inc}$ and $\bar{g}^{(B_{T-1})}_{\mathcal{P}}$. Those sub-forests will also be a little correlated (since they will produce at the end an unique forest of all trees), leading to convergence with the increasing number of chunks. Hence, one can compute many different models (with different parameters and data) and just combine them. One alternative is to use incremental trees that update themselves with new data. While possibly more powerful with deeper trees, this technique requires more parameters and much more attention to the tree structure. In the (wild) *non-i.i.d* case, there is no result in supervised learning we can use to assess the model, since even very simple rule linking target and predictors can no more be learned. We found in this case that unsupervised learning, as a companion of the supervised case, seems a promising alternative.

4 Experiments

To better understand how Random Uniform Forests works in practice, we provide an R package with a large number of functionalities, *randomUniformForest*, abbreviated rUF.

We use 34 datasets, from both classification and regression, to show how the algorithm performs. For comparison, we use Random Forests (RF), Extra-Trees (ET), Gradient Boosting Machines (GBM), SVM and CART as companions, using, respectively, the packages *randomForest*, *extraTrees* (a free implementation of the original algorithm), *gbm*, *e1071* and *rpart*.

4.1 Protocol

All experiments are done using R (R Core Team, 2014), the free software environment for statistical computing and graphics.
For almost all the datasets, we compute the 10-fold cross-validation error, 5 times, and report the mean and the standard deviation in parenthesis.

- Random Uniform Forests always applies (internally) a preprocessing step to the data. Algorithms that only allow matrix, or that can not handle some categorical variables, see the data to be converted, using the same preprocessing step than Random Uniform Forests. Others algorithms take the data as they come except when they were too sensitive to the data (like the GBM algorithm in the *credit approval* dataset).
- For all algorithms we use their default parameters, to preserve fairness, except for GBM whose default ones give in too many cases the worst results. Hence we use for this algorithm parameters (that replace well the default ones but with longer computation times) namely : *shrinkage = 0.05, interaction.depth*[1] *= 12, n.minobsinnode = 1.*

- Default parameters for the *randomUniformForest* algorithm :
number of trees, *ntree = 100*,
number of variables to select at each node, $mtry = \lceil \frac{4}{3}p \rceil$, where p is the number of variables of the training sample,
minimal node size, *nodesize = 1*,
for classification, *replace = TRUE*, meaning that a random bootstrap sample is drawn for each tree,
for regression, *subsamplerate = 0.7*, meaning that a random subsample, of size equal to 70% of the training sample size and without replacement, is drawn for each tree.

- Default parameters for the *randomForest* algorithm :
ntree = 500,
$mtry = \lceil \sqrt{p} \rceil$ for classification, $mtry = \lceil p/3 \rceil$ for regression,
nodesize = 1 for classification, *nodesize = 5* for regression,
bootstrap always enabled.

- Default parameters for the *extraTrees* algorithm :
same as randomForest ones, except that the whole sample is used for each tree, without any modification.

[1]The value of interaction.depth was initially set to 24 but after some issues in multi-class classification and slow computation time, we decided to set it to 12.

Classification datasets : *Heart disease (with 13 attributes), Liver, Ionosphere, SAheart, Credit approval, Climate model, Pima indians diabetes (Diabetes), Vehicle, German credit, QSAR biodegradation, Banknote authentification, Yeast, Car evaluation, Steel plates faults, Wine quality (white), Musk, Bank marketing.*

Regression datasets : *Yacht Hydrodynamics, Auto MPG, Boston housing, Forest fires, Energy efficiency, Stock Prices, Friedman_c3_1000_25, Mortgage, Concrete compressive strength, Airfoil self-noise, Treasury bounds, Weather Ankara (Wankara), Puma32h, Pole telecommunications, California housing, YearPredictionMSD.*

Most of the datasets are available from the *UCI repository*, except the Friedman data set, SAheart (in the R package *ElemStatLearn*), Stock prices, Mortgage, Treasury, Weather Ankara (Wankara), Puma32h, Pole telecommunications, California housing.

4.2 Results

4.2.1 Classification

	RF	ET	SVM	GBM	CART	rUF
Heart disease	15.89 (6.99)	17.20 (6.02)	17.21 (6.48)	19.57 (6.99)	19.20 (5.38)	18.26 (6.14)
Liver	26.2 (6.96)	27.48 (6.3)	29.52 (6.73)	30.15 (6.54)	31.3 (7.34)	27.01 (6)
Ionosphere	7.52 (5.83)	7.03 (4.50)	9.07 (6.55)	8.08 (5.30)	11.14 (7.04)	7.19 (5.44)
SAheart	30.99 (6.29)	32.24 (6.91)	28.09 (6.15)	35.72 (6.57)	31.28 (6.56)	32.11 (5.97)
Credit approval	12.78 (3.53)	12.72 (3.61)	13.65 (3.39)	13.27 (4.06)	13.94 (4.04)	12.40 (3.87)
Climate model	8 (3.44)	8.48 (3.47)	8.14 (3.46)	6.22 (2.69)	7.37 (3.37)	6.40 (2.72)
Diabetes	23.09 (4.43)	23.17 (4.54)	23.95 (5.03)	25.62 (5.10)	25.33 (5.24)	23.69 (4.33)
Vehicle	24.72 (4.86)	25.93 (4.55)	23.23 (4.05)	21.58 (4.15)	32.21 (4.77)	24.28 (5.72)
German credit	23.16 (3.96)	24.58 (4.04)	24.30 (4.68)	23.46 (3.85)	26.20 (3.01)	24.30 (4.12)
QSAR biodegradation	12.64 (3.27)	12.59 (3.70)	12.43 (3.33)	12.79 (3.19)	17.41 (3.29)	12.79 (2.49)
Banknote	0.62 (0.7)	0.1 (0.25)	0 (0)	0.45 (0.56)	3.87 (2.5)	0.26 (0.38)
Yeast	35.95 (3.52)	36.80 (3.40)	38.36 (3.86)	32.5 (2.85)	43.39 (3.86)	36.95 (4.48)
Car evaluation	2.66 (1.38)	0.89 (0.75)	5.66 (2.18)	0.68 (0.72)	5.67 (1.66)	2.16 (1.38)
Steel plates faults	21.42 (2.53)	21.27 (2.74)	19.39 (2.53)	-	31.03 (3.67)	19.87 (2.97)
Wine quality (white)	29.48 (2.07)	-	26.58 (2.26)	31.71 (2.04)	47.14 (2.02)	29.59 (2.22)
Musk	2.1 (0.62)	1.83 (0.51)	2.95 (0.63)	1.24 (0.51)	5.75 (0.93)	1.91 (0.63)
Bank marketing	8.44 (0.32)	8.92 (0.38)	9.04 (0.31)	-	8.78 (3.5)	8.51 (0.43)

Table 1: Classification. Reporting test error in % and standard deviation (in parenthesis) of the 10-fold cross-validation repeated 5 times. Second part of the table are datasets with more than 1000 rows. Default parameters for all classifiers except GBM.

From all the datasets, the only classifier that is clearly behind the others is CART. From the theoretical side, using the Breiman's bound, one can explain that most of the improvements of random forests like classifiers come from the little correlation of trees. Even if one unpruned and randomized binary tree is worst than a CART one, using many will enough improve the average margin (difference, in frequency, between well-classified and misclassified examples) so that the Breiman's bound will usually be lower for an ensemble model. From the others classifiers, the difference in test error is so close, in a sense or another, that results depend most on the dataset and on the default parameters.

For example, Random Uniform Forests use 100 trees while Random Forests and Extra-Trees use 500. But, considering the number of selected variables for each node, Random Uniform Forests use $4/3\sqrt{p}$ times more variables. Considering the sample, Extra-Trees use the whole one while Random Forests and Random Uniform Forests use a bootstrap one. Average test error for all datasets is 16.80% for Random Forests and 16.92% for Random Uniform Forests, with almost the same variance. Hence, to our opinion, a classifier can only be better (or with more predictive abilities) than another if it outperforms for almost all datasets. Here each classifier has its strength. For some datasets, SVM pushes the test error lower than others one or it can be GBM. Others ensemble models seem, overall, to maintain an enough low variance of their prediction error which is an essential property for models that are almost random. In the case of Random Uniform Forests, the model goes a bit further, since its algorithm is clearly stochastic, meaning that one will not be able to reproduce exactly any prediction error with the same data, even by fixing the seed (that generates random numbers). In fact, this aspect is something important in the model since convergence is the master word of Random Uniform Forests. Hence, most of the efforts are done to reduce correlation between trees (for most of the datasets above it is lower than 0.1) and the OOB error (from which one can derive non-asymptotic bounds) is the one from which prediction error is assessed.

4.2.2 Regression

	RF	ET[a]	SVM	GBM	CART	rUF
Yacht hydrodynamics	14.93 (9.1)	18.61 (11.75)	41.06 (6.55)	0.46 (0.51)	4.91 (2.01)	1.5 (1.41)
Auto MPG	7.64 (3.05)	7.22 (3.23)	7.84 (3.35)	7.52 (2.91)	13.58 (4.14)	7.17 (3.26)
Boston housing	10.28 (4.56)	9.62 (4.36)	14.57 (8.81)	8.41 (4.15)	22.41 (10.27)	10.25 (5.9)
Forest fires	4249 (6989)	4307 (6967)	4095 (7239)	5994 (7068)	-	4634 (6969)
Energy efficiency 1	1.18 (0.27)	0.2079 (0.05)	5.03 (1.26)	0.11 (0.05)	6.80 (1.44)	0.21 (0.05)
Energy efficiency 2	3.37 (0.78)	3.06 (0.66)	7.04 (1.52)	0.52 (0.15)	9.42 (1.48)	2.84 (0.47)
Stock prices	0.531 (0.098)	0.421 (0.082)	0.765 (0.12)	0.563 (0.178)	3.95 (0.66)	0.413 (0.072)
Friedman (c3_1000_25)	0.135 (0.018)	0.194 (0.02)	0.621(0.08)	0.052 (0.009)	0.286 (0.038)	0.092 (0.01)
Mortgage	0.017 (0.007)	0.011 (0.005)	0.032 (0.004)	0.01 (0.004)	0.48 (0.08)	0.012 (0.006)
Concrete compressive	27.63 (3.96)	33.38 (8.03)	42.64 (4.68)	15.35 (4.66)	84.34 (13.36)	21.21 (5.86)
Airfoil self-noise	12.67 (1.43)	4.59 (0.86)	10.49 (1.92)	2.42 (0.49)	19.68 (2.27)	4.08 (0.70)
Treasury	0.046 (0.022)	0.036 (0.015)	0.066 (0.021)	0.039 (0.017)	0.4424 (0.108)	0.034 (0.015)
Wankara	3.18 (1.52)	3.1 (1.17)	7.37 (3.74)	2.15 (0.64)	17.19 (3.39)	2.17 (0.58)
Puma32h*(×100)	0.014	0.0281	0.072	0.006	0.0185	0.007
Pole*	32.32	38.11	216.34	28.48	244.93	28.88
California housing*[b]	0.14	0.148	0.108	0.15	0.187	0.137
YearPredictionMSD[c]	-	-	-	81.88	109.99	86.56

[a] Results of Extra-Trees were got with 100 trees.

[b] we used the transformation $Z = log(Y + 1)$ to assess models

[c] The last 51,630 examples (of 515,345 ones) are taken as test set. Out-of-memory problems occurred with ET and we stopped RF after more than 10 hours. rUF used their incremental learning mode to reduce computation time.

Table 2: Regression. Reporting mean squared error and standard deviation (in parenthesis) of the 10-fold cross-validation repeated 5 times (except for the four last datasets).
* : For these datasets, the second half was taken as a test set.

From the table above, things are now very different. At first, GBM dominates, by far, most of the results for small datasets. What we observed is that GBM is (one of)

the best algorithm(s) as soon as dataset is small or with a small number of covariates and/or the distribution of responses is close to a Gaussian one. For large datasets or when assessing a single test set (second part of the table), things begin to change. We set GBM parameters that might be close to the optimal ones, but they never changed for all the datasets (classification and regression). Hence, it remains that GBM is very impressive and some functionalities in Random Uniform Forests are directly inspired by the former.

From what is reported, Random Forests has most of the times a higher mean squared error than Random Uniform Forests. The first reason is due to the correlation between trees residuals, usually high in regression. This leads to a slower convergence for some ensemble models. In Random Uniform Forests lowering correlation is an achievement and lead to three mechanisms : use random cut-points, exploit the dimension of the problem, use subsampling. As a consequence, Random Uniform Forests are less sensitive to overfitting while their greater randomness does not let the average variance of trees getting high. Extra-Trees seems to also get benefits from random cut-points. To better show the difference between these models, we computed the average correlation between trees residuals and the average variance of trees residuals whose the product leads to the upper bound of the prediction error for a random forest like model. We got :

	RF	ET	rUF
Average correlation between trees residuals (1)	0.4443	0.4243	0.4073
Average variance of trees residuals (2)	0.3239	0.3572	0.3446
Estimate of the theoretical prediction error	0.1428	0.1462	0.1362
Upper bound of the prediction error (1×2)	0.1439	0.1516	0.1402

Table 3: Correlation, variance and prediction error for the test sample of the California housing dataset for a forest of 100 trees. The second half of the dataset was taken as a test set. Size of the test set : 10320×10.

From the table above, we can see that Random Uniform Forests reduce correlation stronger than the two others models (up to 10% below) while the average variance of trees increases by 6% (at most). That leads to a lower theoretical prediction error of the forest (Breiman, 2001) due to the correlation that is applied to all the trees residuals. The gap with the upper bound gets higher than the Random Forest one, due to the higher average variance of trees. Random Uniform Forests, in regression, decrease the correlation faster than they increase the average variance of trees residuals. That is one of the manners to lower the prediction error. In the case of Extra-Trees, the random cut-poins also lead to a lower correlation but, at least for this dataset, the variance does not decrease enough because of the number of selected features at each node (one third of the total number of variables, like Random Forests). Increasing the number of trees decrease all the measures but does not change the order.

One way to further assess the models is to find the best number of selected features per node (the *mtry* value) and to look how it affects measures:
In the previous table, the values of *mtry* were 2 (RF), 2 (ET) and 10 (rUF). Increasing the default values reduce the prediction error, by reducing both correlation and variance except for Random Forests (while we optimized the mtry parameter using the OOB er-

	RF (mtry = 8)	ET (mtry = 8)	rUF (mtry = 20)
average correlation between trees residuals (1)	0.5329	0.4165	0.3944
average variance of trees residuals (2)	0.2889	0.3083	0.3373
Estimate of the theoretical prediction error	0.1531	0.1267	0.1298
Upper bound of the prediction error (1×2)	0.1539	0.1284	0.1330
Mean squared error	0.15	0.1261	0.1264

Table 4: Correlation, variance and prediction error for the test sample of the California housing dataset for a forest of 100 trees with optimized *mtry* value. Test set size: 10320 × 10.

ror). Default values in regression are not easy to set for Random Forests, since the model has to avoid the correlation getting high, which will happen when mtry is increasing, and in the same time reduce variance, which will not happen fairly rapidly for low values of mtry. Extra-Trees take benefits from the increased number of selected features and this seems to be a paradigm for methods that use random cut-points. In comparison to Random Uniform Forests they use around 40% more examples for growing a tree (no subsampling), but around half of the features used by the former for this dataset (sampling with replacement features in Random Uniform Forests) and a minimal number of observations in a node of 5 (against 1).

We show in the figure below how correlation and variance evolve with either the number of trees or the mtry value :

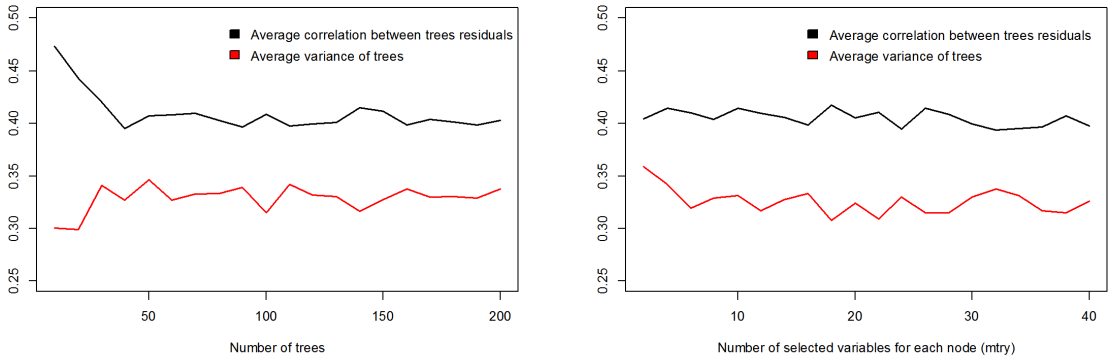

Figure 1: Correlation and variance in Random Uniform Forests for the California housing dataset.

One can see that reducing correlation depends more on the random cut-points. Number of trees is important to get stability, while mtry value does not matter a lot. Considering the average variance of trees, it is reduced up to a point by using larger values of mtry (meaning that an increasing number of features are selected twice or more times) while the number of trees is, again, essential to get stability. For this dataset, much of the work to reduce correlation is natively done by the algorithm structure. What is interesting is both correlation (as low as 0.2) and variance (as low as 0.28) can be reduced further using options, but hardly in the same time.

5 Discussion

We have proposed in this article a novel extension of random forests, firstly designed to reduce correlation between trees (and trees residuals). To achieve this goal, we chose *random cut-points using the continuous Uniform distribution on the support of each candidate variable, or between two random points of the latter*. This first step is enough but may lead to an higher prediction error, comparing to the Breiman's procedure. To narrow the gap, we used *for each node a set of features drawn randomly, and uniformly, with replacement*. This lead to have similar performance than Random Forests, making the cost less expensive, especially for large datasets. The last step, that is receding our model from the original one, is the use of a *different optimization criterion (the information gain) in classification and the use of subsampling in regression*. However in practice, Random Uniform Forests can also use the Gini criterion which does not really seem to change performance. Random Uniform Forests can be viewed as an ensemble of randomized trees which produces a whole framework for many tools, while not discussed here, like deep variable importance analysis, partial dependencies and extrapolation, predictions and confidence intervals, post-processing functions for reducing bias, missing values imputation in many ways, native handling on sparse data, imbalanced class techniques, unsupervised learning or straightforward incremental and distributed learning. From the theoretical side, the main mechanism is the use of random cut-points in the splitting procedure rather than finding the best one for each candidate variable. As it can be observed in Extremely Randomized Trees, this has the advantage to decrease, especially in regression, the correlation between trees residuals more than in Breiman's Random Forests. While the average variance of trees residuals increases, it is usually slower than the decrease of correlation. Results observed in regression show that this strategy is effective and the main theoretical argument resides in the fact that convergence, which can be considered as the primal property of random forest models, is more likely to happen as the correlation decreases. It still remains some questions on how to further the reduce the correlation, which seems less complex than the alternative of finding ways to reduce more the average variance of (randomized) trees residuals.

References

Biau, G., 2012. Analysis of a Random Forests Model. *The Journal of Machine Learning Research* 13, 1063-1095.

Biau, G., Devroye, L., Lugosi, G., 2008. Consistency of random forests and other averaging classifiers. *The Journal of Machine Learning Research* 9, 2015-2033.

Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C., 1984. *Classification and Regression Trees*. New York: Chapman and Hall.

Breiman, L., 1996. Bagging predictors. *Machine learning* 24, 123-140.

Breiman, L., 1996. Heuristics of instability and stabilization in model selection. *The annals of statistics* 24, 2350-2383

Breiman, L., 1999. Pasting Small Votes for Classification in Large Databases and On-Line. *Machine Learning* 36, 85-103.

Breiman, L., 2001. Random forests. *Machine learning* 45, 5-32.

Breiman, L., 2001. Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science* 16, 199-231

Ciss, S., 2014. *randomUniformForest: random Uniform Forests for Classification, Regression and Unsupervised Learning.*
R package version 1.1.2, http://CRAN.R-project.org/package=randomUniformForest.

Ciss, S., 2014. *Forêts uniformément aléatoires et détection des irrégularités aux cotisations sociales* (in French). PhD thesis, 2014. Université Paris Ouest Nanterre.

Devroye, L., Györfi, L., Lugosi, G., 1996. *A probabilistic theory of pattern recognition*. New York: Springer.

Dietterich, T.G., 2000. Ensemble Methods in Machine Learning, in: *Multiple Classifier Systems*, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 1?15.

Dietterich, T.G., 2000. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning* 40, 139-157.

Friedman, J.H., 2001. Greedy function approximation: A gradient boosting machine. *Annals of Statistics* 29, 1189-1232.

Friedman, J.H., 2002. Stochastic gradient boosting. *Computational Statistics and Data Analysis* 38, 367-378.

Geurts, P., Ernst, D., Wehenkel, L., 2006. Extremely randomized trees. *Machine Learning* 63, 3-42.

Hastie, T., Tibshirani, R., Friedman, J.J.H., 2001. *The elements of statistical learning.* New York: Springer.

Ho, T.K., 1998. The random subspace method for constructing decision forests.*IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, 832-844.

Liaw, A., Wiener, M., 2002. Classification and Regression by randomForest. *R News* 2(3), 18–22.

Lin, Y., Jeon, Y., 2002. Random Forests and Adaptive Nearest Neighbors. *Journal of the American Statistical Association* 101-474.

Oza, N.C., 2005. *Online bagging and boosting*, in: 2005 IEEE International Conference on Systems, Man and Cybernetics, pp. 2340-2345 Vol. 3.

R Core Team (2014). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Ridgeway, G., with contributions from others, 2013. *gbm: Generalized Boosted Regression Models.* R package version 2.1. http://CRAN.R-project.org/package=gbm

Scornet, E., Biau, G., Vert, J. P., 2014. Consistency of Random Forests. *arXiv* preprint arXiv:1405.2881.

Shannon, C.E., 1949. *The Mathematical Theory of Communication.* University of Illinois Press.

Vapnik, V.N., 1995. *The nature of statistical learning theory.* Springer-Verlag New York.