Forest-RK: A New Random Forest Induction Method

Simon Bernard, Laurent Heutte, Sébastien Adam

Forest-RK : A New Random Forest Induction Method

Simon Bernard, Laurent Heutte, and Sébastien Adam

Université de Rouen, LITIS EA 4108
BP 12 - 76801 Saint-Etienne du Rouvray, France.
{simon.bernard,laurent.heutte,sebastien.adam}@univ-rouen.fr

Abstract. In this paper we present our work on the parametrization of Random Forests (RF), and more particularly on the number K of features randomly selected at each node during the tree induction process. It has been shown that this hyperparameter can play a significant role on performance. However, the choice of the value of K is usually made either by a greedy search that tests every possible value to choose the optimal one, either by choosing *a priori* one of the three arbitrary values commonly used in the literature. With this work we show that none of those three values is always better than the others. We thus propose an alternative to those arbitrary choices of K with a new "push-button" RF induction method, called Forest-RK, for which K is not an hyperparameter anymore. Our experimentations show that this new method is at least as statistically accurate as the original RF method with a default K setting.

Key words: Classification, Classifier Ensemble, Classifier Combination, Random Forests, Decision Trees, Bagging

1 Introduction

Random Forest is a family of classifier ensemble methods that use randomization to produce a diverse pool of individual classifiers, as for Bagging [1] or Random Subspaces methods [2]. It can be defined as a generic principle of classifier combination that uses L tree-structured base classifiers $\{h(x, \Theta_k),\ k = 1, ...L\}$ where $\{\Theta_k\}$ is a family of independent identically distributed random vectors, and x is an input data. The particularity of this kind of ensemble is that each decision tree is built from a random vector of parameters. A Random Forest can be built for example by randomly sampling a feature subset for each decision tree (as in Random Subspaces), and/or by randomly sampling a training data subset for each decision tree (as in Bagging). Since they have been introduced in 2001, RFs have been studied in many ways, both theoretically and experimentally [3–11]. In most of those works, it has been shown that RFs are particularly competitive with one of the most efficient learning principles, i.e. boosting [5, 7, 10]. However, the mechanisms that explain the good performance of this type of classifier ensemble are not clearly identified and one has to admit that it is still a complex

task for the practitioner to take full benefits of the potential of those methods. For example considering the reference RF method called Forest-RI, introduced by Breiman in [5] (cf. section 2), an important hyperparameter has been identified : the number K of features randomly selected at each node during the tree induction process. Yet, in those research works that have experimented this method, the value of K is arbitrarily or empirically set, and sometimes without any theoretical nor experimental justification.

In this paper we propose an alternative to those arbitrary settings of K. Indeed, it appears that this hyperparameter setting is crucial for accuracy in RF induced with Forest-RI algorithm [3]. However, the choice of the value of K is usually made either by a greedy search that tests every possible value to choose the optimal one, either by choosing *a priori* one of the three arbitrary values commonly used in the literature. With this work we show that none of those three values is always better than the others and that none of them consequently represent a good setting. We thus propose a new RF algorithm, called Forest-RK, based on Forest-RI but for which the setting of K does not play a crucial role anymore for growing accurate RF classifiers. We show that this new method is at least as statistically accurate as the "classic" Forest-RI algorithm with default settings.

The paper is thus organized as follows: in the following section, we detail the Forest-RI algorithm used in our experiments; in section 3, we describe our experimental protocol, the datasets used, and compare the results obtained with different settings of the hyperparameter K. We finally draw some conclusions and future works in the last section.

2 The Forest-RI algorithm

One can see RFs as a family of methods, made of different decision tree ensemble induction algorithms, such as the Breiman Forest-RI method often cited as the reference algorithm in the literature [5]. In this algorithm the Bagging principle is used with another randomization technique called Random Feature Selection. The training step consists in building an ensemble of decision trees, each one trained from a bootstrap sample of the original training set — i.e. applying the Bagging principle — and with a decision tree induction method called Random Tree. In this induction algorithm, a feature subset is randomly drawn for each node, from which the best splitting criterion is then selected. Thus, the Forest-RI method grows a decision tree using the following process :

- Let N be the size of the original training set. N instances are randomly drawn with replacement, to form the bootstrap sample, which is then used to build a tree.
- Let M be the dimensionality of the original feature space, and K a preliminary fixed parameter so that $K \in [1, M]$. For each node of the tree, a subset of K features is randomly drawn without replacement, among which the best split is then selected.
- The tree is thus built to reach its maximum size. No pruning is performed.

This process is thus led by an important hyperparameter: the number K of randomly selected features in the splitting process. Whereas this parameter has already shown to be critical for RF performance, no research work has been specifically devoted to study its setting and its real influence on performance, and only a few have empirically dealt with this issue.

In [8] for example, Guerts et al. have proposed a new method of RF induction, called Extras-Trees for Extremely Randomized Tree Ensemble, that modifies the Forest-RI algorithm to accentuate the randomization. Here the Random Feature Selection is still used but modified so that the best splitting criterion selection is one step further randomized. The authors have designed their experimental protocol to study the influence of K on performance. Even if this method is partly different from the Forest-RI algorithm, this work allows to draw some intuitions on the RF behavior according to K. It highlights for example that their default setting $K = \sqrt{M}$, where M stands for the dimensionality of the original feature space, is most of times closed to the optimal setting, at least for the Extras-Trees method and on several representative datasets. Another example is Breiman's work on performance according to this K parameter [5]. In these experiments, a large number of RF has been grown on three databases of the UCI repository, for which the test set error rate has been monitored. Actually only one of those three experiments was really concerned by the Forest-RI algorithm, since the two others have been run with a different induction algorithm that uses feature combinations in the splitting criterion, instead of single features. Hence, even if Breiman draws some tendencies [5], those experiments do not allow to conclude on RF behavior according to the setting of K . We also noticed that in his Forest-RI experiments, Breiman has decided to use two values of $K : 1$ and $log_2(M)+1$. While the first value is intuitively interesting since it corresponds to a decision tree induction that selects in a fully random manner the splitting criterion among features for each node, the second one seems to be more arbitrary or at least is not justified.

Finally, implementation and experimentation of the Forest-RI algorithm require to fix the value of the hyperparameter K but as we have shown, there actually does not exist any theoretical rule that can be used to fix it. As mentioned previously, only arbitrary default values are proposed in the literature and nothing guarantees that these values are close to the optimal setting, as we will show in section section 3.3. We thus propose an alternative to those settings, with a new push-button RF induction algorithm for which K is not an hyperparameter anymore. We describe in the following section our new algorithm and compare it with the "classic" Forest-RI algorithm, through extensive experiments on several datasets and with predefined values of K.

3 Investigating the influence of K on Forest-RI performance

The purpose of this set of experiments is to compare accuracies of Forest-RI algorithm with default settings of K on the one hand, to our new push-button

RF algorithm in which K is not an hyperparameter anymore, on the other hand. In this new algorithm, called Forest-RK, K still exists since Random Feature Selection is still used, but is randomly chosen for each splitting node. Instead of fixing the value of K so that it is identical for all the decision trees, a new value of K is randomly chosen at each node of the trees, and used for this current node splitting only. The new Forest-RK decision tree induction procedure can be summarized as below:

- Let N be the size of the original training set. N instances are randomly drawn with replacement, to form the bootstrap sample, which is then used to build a tree.
- Let M be the dimensionality of the original feature space. Randomly set a number $K \in [1, M]$ for each node of the tree, so that a subset of K features is randomly drawn without replacement, among which the best split is then selected.
- The tree is thus built to reach its maximum size. No pruning is performed.

As shown in the above algorithm, the main difference between Forest-RI and Forest-RK lies in that K is randomly chosen for each node of the tree leading therefore to more diverse trees in the Forest-RK than in the Forest-RI. Two ideas have led us to this new algorithm: i) it avoids the greedy iterative test of every possible value of K to find the best one, while being at least as accurate as the traditional Forest-RI algorithm parametrized with default values; ii) it is an alternative to default settings of K, since as shown in section 3.3, those values are arbitrary and are not always the best choice. Thus the problem is now to determine which of the two algorithms, Forest-RI or Forest-RK, will produce more accurate classifiers when trained and tested on the same datasets. To answer to this question, *i.e.* assessing which of the two algorithms performs better, we lean on the comparison of five approximate statistical tests, compared in [13]. In this paper, it is recommended to use McNemar's test [14], for which it is shown that it better suits to experimental protocols like ours. The McNemar's test is firstly used here to determine whether or not two sets of predictions differ significantly. That is to say, under the null hypothesis H_0, the two algorithms should have the same error rate. Given two algorithms A and B producing two classifiers h_A and h_B, the contingency table is constructed, so that it gives the four values : $n_{00} = \#$ samples misclassified by both h_A and h_B; $n_{01} = \#$ samples misclassified by h_A but not by h_B; $n_{10} = \#$ samples misclassified by h_B but not by h_A and $n_{11} = \#$ samples misclassified by neither h_A nor h_B. McNemar's test is then based on a χ^2 test for goodness-of-fit that compares the distribution of counts expected under the null hypothesis to the observed counts. It thus states that the statistics X^2 (equation 1) can be considered as following a χ^2 distribution with 1 degree of freedom.

$$X^2 = \frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \sim \chi^2_{1,0.05} = 3.841459 \tag{1}$$

Consequently H_0 is rejected, i.e. one of the two algorithms is considered to be "better" than the other, if X^2 is greater than $\chi^2_{1,0.05} = 3.841459$. Finally, when

applied on a specific testing set, three answers can be obtained through the McNemar test :

- H_0 is rejected and $n_{01} > n_{10}$: Algorithm B produces significantly more accurate classifiers than algorithm A.
- H_0 is rejected and $n_{01} < n_{10}$: Algorithm A produces significantly more accurate classifiers than algorithm B.
- H_0 is accepted: The two algorithms do not produce classifiers significantly different in term of accuracy.

With such a procedure, we are able to assess if Forest-RK statistically outperforms or not the Forest-RI algorithm with default K parameter settings. We first describe in the following subsection the datasets used. We then detail our experimental protocol to compare the two algorithms and discuss the obtained results.

3.1 Datasets

The 10 datasets that have been used for these experimentations are described in table 1: The first 8 datasets have been selected from the UCI repository [12]; the two last, Twonorm and Ringnorm, are synthetic datasets designed by Breiman [15]. Those datasets have firstly been selected because they are representative of typical machine learning issues in terms of number of classes, of features and of samples. They have also been chosen because they do not contain any missing values and because the features are all numerical. All those datasets are not preliminary divided into training and testing subsets. Thus for our experiments we have decided to randomly split each original dataset, with two thirds of the instances used for training, and the other third for testing. In order to make sure that our results do not depend on this arbitrary splitting this process has been repeated 10 times with various splittings.

Table 1. Datasets description

Dataset	Size	Features	Classes	Dataset	Size	Features	Classes
Diabetes	768	8	2	Spambase	4610	57	2
Gamma	19020	10	2	Vehicle	946	18	4
Letter	20000	16	26	Waveform	5000	40	3
Pendigits	10992	16	10	Ringnorm	7400	20	2
Segment	2310	19	7	Twonorm	7400	20	2

3.2 Experimental protocol

In this section our experimental protocol is described. It performs comparative tests on the datasets detailed in table 1. For all the experiments described

Algorithm 1 Experimental Protocol

Input: N the number of instances in the original dataset.
Input: M the number of features in the original dataset.
for $i = 1$ *to* 10 **do**
Randomly draw without replacement $\frac{2}{3} \times N$ of the original dataset instances to form a training subset Tr_i. The remaining instances form the test subset Ts_i, and the couple (Tr_i, Ts_i) is denoted T_i.
Grow three Random Forests, on the training set Tr_i, noted h_1, $h_{\sqrt{M}}$ and $h_{log_2(M)+1}$, according to Forest-RI algorithm with K respectively equal to 1, $\sqrt{M}$ and $log_2(M) + 1$.
for $j = 1$ *to* 50 **do**
Grow a Random Forest, noted h_{R_j}, according to Forest-RK algorithm, on the training set Tr_i.
Apply McNemar test on classifier pairs (h_1, h_{R_j}), $(h_{\sqrt{M}}, h_{R_j})$ and $(h_{log_2(M)+1}, h_{R_j})$, with the testing set Ts_i. Store the results.
end for
end for

in this section, the number of trees grown in the forests has been set to 100. This choice is based on a previous experimental work presented in [3], in which we have shown that it is a reasonable value to grow an accurate RF, and considering that we do not seek to reach intrinsic optimal performance. First, each dataset has been randomly split into training and testing subsets, as explained in the previous section. This splitting procedure has been repeated 10 times so that 10 different training sets and testing sets are thus available, each set containing two thirds and one third of the original dataset respectively. We denote by $T_i = (Tr_i, Ts_i)$ such a split, with $i \in [1, 10]$ and where Tr_i and Ts_i stand respectively for the training part and the test part. Then, for each T_i, the Forest-RI algorithm has been run with the three following default values of K: $K = 1$, $K = \sqrt{M}$ and $K = log_2(M) + 1$; and Forest-RK has been run 50 times. By this way, $10 \times 50 \times 3 = 1500$ comparisons between Forest-RI and Forest-RK have been performed for each dataset. Algorithm 1 summarizes the whole experimental protocol applied to each dataset. This procedure outputs for each dataset 3 tables of 500 McNemar test outputs, i.e. values $\in \{-1, 0, 1\}$, corresponding to the three possible cases enumerated previously. Those results are presented and discussed in the next section.

3.3 Results

Table 2 presents a synthesis of our results obtained by the experimental protocol detailed in Algorithm 1. As mentioned above, 3 tables of 500 comparison results are firstly obtained for each dataset. For those 3 tables, the number of occurrences of the three possible cases have been counted and detailed in table 2. The first observation that can be made from this table is that, when all the results are summed for each of the three McNemar possible answers, the second case for which the two algorithms have shown to be statistically equivalent, is strongly in

the majority. Thus, considering all the comparisons performed between Forest-RI and Forest-RK, the two algorithms are as accurate as each other. Then looking at each cell of the table the same observation can be made : for each case — except for two of them, i.e. Letter/$K = 1$ and Ringnorm/$K = 1$ — the two algorithms have shown to be equivalent most of the times. Hence, the McNemar test indicates that the two algorithms are "globally" equivalent; however, as the first and third values in each cell of the table are not always strictly null, one can say that one of the two algorithms is sometimes better than the other. Thus, let us consider only cases for which the McNemar test indicates that one algorithm outperforms the other, by comparing the first and the third values in each cell of the table: we can notice that in 19 of the 30 duels, Forest-RI outperforms Forest-RK most of times, and in 9 of the 30 duels it is the contrary. But concerning the Forest-RI algorithm, when looking at each dataset, it appears to be more interesting to use $K = 1$ for 3 datasets (Spambase,Ringnorm,Twonorm), to use $K = \sqrt{M}$ for 4 datasets (Pendigits,Gamma,Letter,Segment) and to use $K = log_2(M)+1$ for 3 datasets (Diabetes, Vehicle, Waveform). Finally, choosing the best value of K in Forest-RI algorithm still remains an unsolved problem since it depends on the intrinsic characteristics of the tested dataset. Consequently the Forest-RK algorithm is a good alternative to Forest-RI for producing accurate Random Forest classifiers since it provides a means to avoid the selection of the optimal value of K while providing the same good performance in average than those obtained with the best value of K.

Table 2. McNemar test results. In each cell the first number corresponds to cases for which Forest-RK outperforms Forest-RI; the second number to cases for which neither of the two algorithms outperforms the other; and the third number to cases for which Forest-RI outperforms Forest-RK. The number in brackets represents the corresponding value of K.

Dataset	$K = 1$	$K = \sqrt{M}$	$K = log_2(M) + 1$
Diabetes	4/**482**/14	0/**490**/10 (3)	12/**471**/17 (4)
Gamma	3/**495**/2	8/**473**/19 (3)	6/**488**/6 (4)
Letter	**425**/75/0	1/**350**/149 (4)	0/**374**/126 (5)
Pendigits	38/**467**/0	4/**491**/5 (4)	49/**451**/0 (5)
Segment	142/**358**/0	7/**490**/3 (4)	6/**494**/0 (5)
Spambase	1/**408**/91	0/**416**/84 (8)	0/**452**/48 (7)
Vehicle	5/**495**/0	6/**491**/3 (4)	5/**490**/5 (5)
Waveform	50/**450**/0	1/**447**/52 (6)	0/**440**/60 (6)
Ringnorm	0/3/**497**	0/**414**/86 (4)	1/**479**/20 (5)
Twonorm	0/**296**/204	1/**485**/14 (4)	1/**483**/16 (5)
Sums	627/**3034**/839	20/**4074**/406	74/**4134**/292

4 Conclusions

Investigations on RF parametrization have been presented in this paper, that have focused on the number K of features randomly selected at each node during the tree induction process. A new push-button algorithm has been presented for which the setting of K is not a crucial issue anymore. Experimental comparisons with the reference algorithm Forest-RI, using the McNemar statistical test of significance, have shown that this new algorithm produces classifiers that are statistically as accurate as the Forest-RI inducted RF with default settings of K usually found in the literature. Since the setting of K for the Forest-RI algorithm is still an unsolved issue, it appears that Forest-RK is a good alternative for producing accurate classifiers. However, some issues still remain when focusing on hyperparameter K : Is there any other value than those proposed in the literature, that can make Forest-RI produce more accurate classifiers? How can the optimal setting be determined? Answering to those questions would furthermore be interesting for adapting this new algorithm so that it would be able to change the setting of K to a value known to be useful to produce a more accurate classifier.

References

1. Breiman, L.: Bagging predictors. Machine Learning **24**(2) (1996) 123–140
2. Ho, T.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(8) (1998) 832–844
3. Bernard, S., Heutte, L., Adam, S.: Using random forests for handwritten digit recognition. International Conference on Document Analysis and Recognition (2007) 1043–1047
4. Boinee, P., Angelis, A.D., Foresti, G.: Meta random forests. International Journal of Computationnal Intelligence **2**(3) (2005) 138–147
5. Breiman, L.: Random forests. Machine Learning **45**(1) (2001) 5–32
6. Breiman, L.: Consistency of random forests and other averaging classifiers. Technical Report (2004)
7. Cutler, A., Zhao, G.: Pert - perfect random tree ensembles. Computing Science and Statistics **33** (2001)
8. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning **36**(1) (2006) 3–42
9. Latinne, P., Debeir, O., Decaestecker, C.: Limiting the number of trees in random forests. 2nd International Workshop on Multiple Classifier Systems (2001) 178–187
10. Rodriguez, J., Kuncheva, L., Alonso, C.: Rotation forest : A new classifier ensemble method. IEEE Transactions on Pattern Analysis and Machine Intelligence **28**(10) (2006)
11. Robnik-Sikonja, M.: Improving random forests. European Conference on Machine Learning, LNAI 3210, Springer, Berlin (2004) 359–370
12. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
13. Dietterich, T.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation **10** (1998) 1895–1923
14. Everitt, B.: The Analysis of Contingency Tables. Chapman and Hall, London (1977)
15. Breiman, L.: Arcing classifiers. The Annals of Statistics **26**(3) (1998) 801–849