Pruning of Random Forests: a diversity-based heuristic measure to simplify a random forest ensemble

TALEB ZOUGGAR SOUAD[1]
ADLA ABDELKADER[2]

Oran 2 University
Department of Economics
Algeria
Oran 1 University
Department of Computer Science
Algeria
[1]souad.taleb@gmail.com
[2]adla.abdelkader@univ-oran1.dz

Abstract. Random forests are among the most successful ensemble methods. They are fast, noise-resistant and do not suffer from over-learning. Moreover, they offer possibilities of explanation and visualization. In this paper, we propose to simplify a set of random forests using an entropy function that measures the diversity of trees in the forest. The function is used in two types of paths: a Sequential Forward Selection (SFS) path and a path based on genetic algorithms (GA). The proposed methods are applied to datasets of the UCI Repository. The results are encouraging and provide ensembles of smaller sizes with performances that are similar to or even,in some cases,exceed the performances of the initial forest. Moreover, the comparison between the two methods shows that in most cases SFS provides reduced ensemble compared to GA, but the latter gives better success rates in the majority of cases.

Keywords: Classification, CART Trees, Random Forests, Selection, Diversity, Accuracy, Entropy, Fitness, Forward Selection, Genetic Algorithms.

(Received May 1st, 2019 / Accepted June 1st, 2019)

1 Introduction

Random forests [14] use bagging [13] to generate CART trees [16]. Bagging allows random selection of a subset of training data (bootstrap) for generating each tree in the forest. Bootstraps are built using random draws with delivery to the original learning set.

A random selection of variables (or Random feature selection) is added to the bagging. This selection allows choosing a subset of variables for the partition at each node; a fixed number of K characteristics is chosen randomly and from which are chosen those which optimize the partitioning.

The goal of combining two principles of randomization is to make models (trees) built more independent of each other. This independence will increase the ensemble performance. The approach is very efficient in biochips, signals, images and curves. Moreover, it is very simple to implement and generates a low computation cost compared to the performances obtained.

A large number of trees forming the forest have also the effect to reduce the variability of the global predictor. In his paper Breiman [14] has shown that beyond a certain number of trees the error in generalization tends to its maximum, which shows that a large number of trees in a forest does not make it more efficient. In this direction, several studies try to limit the number of trees in a random forest by trying to find the optimal ensemble of the forest. This process is called "Random Forest

Pruning".

The pruning of a random forest is an additional step which aims to reduce the number of constitutive trees. This allows saving the storage space and reducing the prediction time while aggregating or combining all the generated trees. In a regression case, aggregating the predictions of q predictors consists in averaging them: given q models, each of them provides a response y_l and,then the final prediction is $\frac{1}{q}\sum_1^q y_l$. In the case of classification, aggregation consists in making a majority vote among the classes provided by each predictor.

In this paper we propose to prune a random forest using a heuristic measure and two paths: Sequential Forward Selection (SFS) and a course based on genetic algorithms (GA). The measure maximizes the chances of choosing the trees that disagree with the ranking of an instance. The two search strategies allow browsing the tree space using different paths.

The experiments are simulated on benchmark datasets of the UCI Repository [9] with a comparative study carried out between the initial ensemble constituted of all the trees and the ensembles obtained after simplification based on two criteria: performance and size of the ensemble obtained.

The rest of the paper is organized as follows: In section 2, we give some preliminaries on random forests and sequential forward selection. Section 3 presents related work that led to the discovery of forests as well as the areas of application in which the approach was applied. In section 4 we describe our proposal for random forest pruning. Section 5 outlines the proposed measure using a diversity based function as well as the two search strategies associated with their algorithms are detailed. In section 6 we present some experimental results. Finally in the last section we conclude and give some future work.

2 Preliminaries

2.1 Random Forests

The following definition of a random forest is given in [14]: Let $\{h(.,\Theta_1),...,h(.,\Theta_q)\}$ be a collection of predictors using trees, where $(\Theta_1,...,\Theta_1)$ is a sequence of random variables, independent of the learning sample Ω_n. The predictor of random forests is obtained by aggregating this collection of predictors.

Random forests RI (Random Input) are an implementation of random forests [15] which the corresponding general algorithm is as follows:
Input:
Ω_n: A learning sample comprising n examples and p variables,

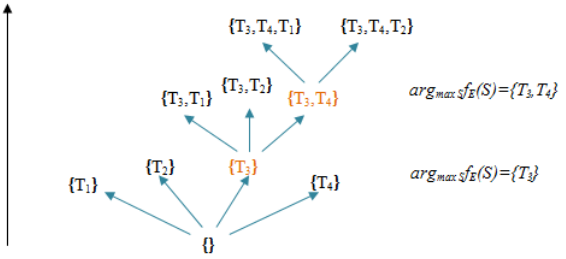

Figure 1: SFS path for an ensemble composed of 4 trees

m: a subset of variables to choose at each step;
Output:
T = Number of trees to build;
Begin
To build each tree:
Create a sample Ω_B of Ω_n
Create a non-pruned CART tree;
At each node of the tree, choose randomly m variables from which the test variable will be chosen;
End.

2.2 Sequential Forward Selection

A Sequential Forward Selection (SFS) path is used to find a sub-optimal solution, because the sequential process used by this method makes each iteration dependent on the previous one and, therefore, all the possible solutions are not explored. However, this path method has the advantage of being simple and fast. Its principle is simple: the process starts from an empty set and add progressively the models that maximize the evaluation function (the entropy function here). The models are added if they belong to the neighborhood of the current ensemble of trees. The process stops when there are no trees to add or there is no improvement in the evaluation function. The neighborhood of the sub-ensemble $SUB = \{T_2\}$ is the sub-ensemble augmented with one tree among the remaining trees neighborhood $SUB = \{\{T_2,T_1\},\{T_2,T_3\},\{T_2,T_4\}\}$.
The following algorithm shows the steps to perform a forward selection:
Algorithm SFS ;
Input :
F : Ensemble of initial classifiers ;
T_2 : A classifier from F ;
f : Evaluatuin Function to optimize ;
Eval : Pruning Sample ;
Output :
SUB : Sub-ensemble of F ;
Begin

SUB :=Φ ;
S :=T_j ;
While F $\neq \Phi$ Do
SUB :=SUB U $argmax_f(S, Eval)$;
S :=Neighborhood(SUB) ;
F :=F-SUB ;
End While ;
End.

3 Related Work

Several decision tree ensemble methods have emerged. They have been successfully applied to various applications. Early work addressing issues related to the synthesis results of multiple trees [33][34] shows that a large improvement in accuracy can be achieved by using the same training sample to generate a combination of binary decision trees (generated by selection criteria for different variables) and combining them using the Dempster and Shafer model [18] [12]. The approach is applied in the field of character recognition.

The proposition of [29] consists to generate multiple trees by changing the learning parameters. The generation of decision tree committees by stochastic selection of attributes has also been proposed in [23] [7] [1].

Tin Kam Ho [25] proposes to create a set of decision trees (decision forest) by randomly selecting a subensemble of variables to construct each tree. The random draw of variables to cut a node had also been used by [8] in image recognition problems for random feature selection or random trees. They introduce a disturbance in the choices of the internal partitions, by preselecting randomly at each node, a sub-ensemble of variables to choose the optimal partition.

Based on the work presented in [8] [14] introduces Random Forest (RF). Since their appearance, forests have been used in a wide variety of fields of application, particularly in the medical field.

In [27], the authors propose to classify faces using random forests. The main purpose of this work is to further reduce the error of facial image classification. Classification error has been significantly reduced compared to popular classifiers including SVM methods.

In a comparative study, [3] uses SVM and random forests to detect spams from private and public emails from a large community of Internet users collected over several years.

Network Intrusion Detection Systems (NIDSs) are used in network security. NIDS are rule-based systems for which performance depends on these sets. However, with the large volume of network traffic, the coding of rules by security experts becomes difficult and time-consuming. For intrusion detection, [4] apply random forests. They deal with the problem of unbalanced intrusions, features selection and optimization of random forest parameters. Experimental results on KDD'99 datasets show that the proposed approach gives better performance than the best results of KDD'99.

[30] use two types of random forests (one for the binary classification and the other for the regression) on a real sample of 1,000,000 customers from a data warehouse of a major European company of financial services. They note an improvement in estimation and validation compared to linear and logistic regression models.

[19] uses random forests for classification in ecology, which are large-scale data with complex and nonlinear interactions as well as a lot of missing data.

The study carried out by [17] tries to check the robustness of classification methods using random forests on agronomic data. These data are characterized by interactions often complex as well as samples of modest size.
[26] proposes to use random forests for supervised and unsupervised categorization of emails and filtering of spam emails. He shows that random forests are more suitable for these tasks and operate quickly on large bases.
The pruning methods of random forests can be classified into two categories: static and dynamic. Static methods generate a fix number of trees then select the ones that will be part of the random forest, while dynamic methods generate trees that will be directly included in the forest using a certain criterion.

For the static approach, [2] proposes to use a direct and non-parametric comparison test. The McNemar test [32] allows deciding whether to include a tree in an ensemble or not. The process systematically determines a minimum number of models to combine for a given database. Knowing the minimum size of the classifier ensemble that gives the best accuracy allows saving time and storage space especially for large data sizes and real-time applications.

In order to reduce the number of trees in the forest while maintaining its precision, [11] proposed methods of tree selection after the construction of the forest. The authors show that better sub-ensembles of decision trees can be obtained by using the sub-optimal methods of selecting SFS (Sequential Forward Selection) and SBS (Sequential Backward Selection) classifiers where adding or removing models is based on the performance measure.

[40] introduce a pruning algorithm based on margin optimization that can reduce the size and increase the performance of a random forest ensemble. The pro-

posed algorithm takes into account the distribution of the forest margin on the learning ensemble. To this end, four different metrics based on the margin distribution are used to evaluate the generalization capacity of sub-ensembles and the importance of individual classifiers. Once the forest is built, the trees are ordered according to the margin metrics. Finally, ensembles with decreasing sizes are constructed by recursively removing the least important trees one at a time.

[31] propose to prune a random forest (RF) for limited sources prediction. Initially an RF random forest is constructed then pruned to optimize the cost and accuracy of the expected features. The forest pruning program encompasses linear constraints that favor the reuse of features. The total uni-modularity of the constraints is set to prove that the corresponding LP relaxation solves the original program. Connections to combinatorial optimization are fianally exploited and an efficient primal-dual algorithm adaptable to large scale data is developed.

[22] use statistical analyzes of basic classifiers to ensemble pruning without compromising the classification accuracy. Learning the statistics of the entire forest in addition to the information available in the dataset can reveal the optimal thresholds that should be used to prune an ensemble model.

[6] propose an ensemble selection technique that provides a small size and a great accuracy. They use a genetic algorithm for which the initial population is composed of individual trees with high performance to improve the result of the algorithm.

[39] propose a new forest simplification strategy by assessing the importance of tree branches against the complete ensemble. This importance is evaluated considering the ensemble performance as well as the diversity of the elements composing the whole ensemble. The proposed metric is used to evaluate how well the ensemble accuracy can be improved when a branch is pruned.

For the dynamic approach, which consists in generating trees gradually satisfying a certain criterion, several works have also been proposed, namely [38] which proposes the development of a method which automatically determines the number of trees to include in a forest during the generation process. The method is based on the use of an online adjustment procedure and is evaluated using conventional random forests and its variants as ensemble methods. Initially the ensemble contains ten trees. At each iteration, a new tree is added and tested if it allows a better fit. To select the best fit, eight polynomials are used. The end of the iterative process is based on predefined thresholds for the adjusted value and accuracy.

[10] propose to add trees independently. A tree is added based on the evaluation of the current sub-forest using adaptive approach. A tree is initially generated, then, to generate the next tree, the weights of the individuals of the learning sample are modified. These weights are incremented for misclassified instances and decremented for those that are well ranked. The trees generated are thus dependent on each other.

[5] develops three heuristics to improve learning by random forest. The first is to use disjointed data partitions to learn basic trees, then to reduce the depth of trees without using repetitive variables, and finally to select reduced sub-ensembles of attributes for cutting at each node of each tree.

4 The Measure for Random Forest Pruning

We propose a static method of random forest pruning which consists in generating the whole forest in what we call an overproduce phase. Then we will eliminate trees that negatively influence performance see Figure 2.

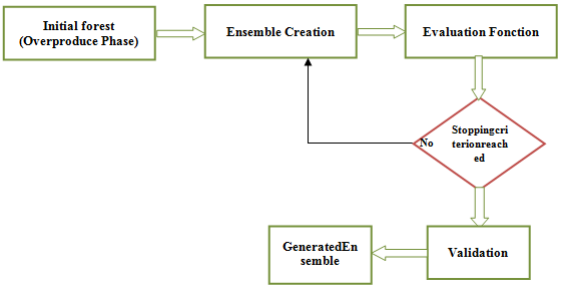

Figure 2: Generating steps of an ensemble from an initial random forest

The forest ensemble is generated. A first ensemble containing a single tree chosen randomly is created. A diversity-based evaluation function is calculated: if its value is improved, we continue to add trees to the ensemble otherwise we stop and choose the current ensemble.

5 Diversity Based Function

The key idea in this approach is to generate only trees that have maximum diversity (they are less correlated with each other). This is based on the principle that the error of generalization of the random forest reduces while diversity increases among the trees.

Let Ω_V be a sample of individuals with their labels (classes), $|\Omega_V|$=n, $|\Omega_V| = \{v_1, ..., v_n\}$. Each individual v_j is described by m variables denoted $x_{1j}, ..., x_{mj}$. Let C_i be a classifier belonging to the classifiers ensemble $\{C_1, ..., C_i, ..., C_T\}$ represented by a n-dimensional binary vector $y_i = (y_{1i}, ..., y_{ni})^T$ such that $y_{ji} = 1$ if the classifier C_i recognize the individual v_j and 0 otherwise.

5.1 Entropy Function

The entropy function f_E measures the diversity within an ensemble (forest) [28]. Given an individual $x_j \in \Omega_V$, if half of the classifiers T/2 doesn?t misclassify x_j then the other half T-T/2 misclassifies it necessarily and vice versa. In this case, we speak of maximum diversity.
We note $nc(x_j)$ the number of classifiers of T which correctly classify x_j, $nc(x_j) = \sum_1^T y_{ji}$. The entropy measure f_E is written as:

$$\frac{1}{n} * \sum_{j=1}^{n} \frac{1}{T - \frac{T}{2}} min\{nc(x_j), T - nc(x_j)\} \quad (1)$$

$f_E \in [0, 1]$ where 1 indicates a very large diversity and 0 a lack of diversity. Thus, the goal is to maximize the f_E function.

5.2 Genetic Algorithm (GA)

Genetic algorithms are a preferred technique for selection because they are inspired by natural selection. They generate individuals that optimize an evaluation function also called fitness function.

A genetic algorithm is defined by [20]:
- Individual also called chromosome or sequence represents a potential solution of the problem. In our case, a solution of the problem corresponds to a binary string of size T (corresponds to the number of trees composing the forest). A chromosome is noted $ch = (val_1 val_2 ... val_T)$ where val_i= 1 if the tree is present in the selected chromosome and 0 otherwise;
- Population corresponds to all the chromosomes representing all possibilities of 1 and 0 in a binary chain of size T;
- Environment represents the search space $|ER| = 2^T$.
- Fitness function corresponds to the function $ff_E = f_E$ (f_E is the diversity function defined above). The goal is to minimize the value of ff_E.
For example, given a forest composed of 4 trees $C = \{T_1, T_2, T_3, T_4\}$, the chromosome $ch_1 = (1010)$ corresponds to the fact that the trees T_1 and T_3 are chosen in the ensemble. Classification vectors on Ω_V correspond to the two trees. If $|\Omega_V|$=2 then the classification vectors $(10)^t$ and $(01)^t$ are associated respectively with T_1 and T_3. Calculate the ff_E fitness function for chromosome ch_1 is equivalent to calculate f_E.
n=2=$|\Omega_V|$, T=2 (the classifiers to which correspond the value 1 for the chromosomes T1 and T3), x_1 and x_2 are the individuals of Ω_V classified respectively $(10)^t$ and $(01)^t$ by T_1 and T_3.
$nc(x_1)$=1 (the number of trees that correctly classify instances x_1).
$nc(x_2)$= 1 (the number of trees that correctly classify instances x_2).

$$f_E = \frac{1}{2}(\frac{1}{2 - \frac{2}{2}} * min(1, 2-1) + \frac{1}{2 - \frac{2}{2}} * min(1, 2-1)) = 1 \quad (2)$$

and $ff_E = f_E$=1, ff_E takes its minimum which is equal to 0 when the trees are consonant and its maximum 1 when they are discrepant; Hence $ff_E \in [01]$.
We propose the AG_{fE} algorithm that uses a genetic algorithm-based path and an entropy-based fitness function. It is described by the following pseudo code:
Algorithm AG_{fE};
Input :
$C = (C_1, ..., C_i, ..., C_T)$: a forest composed of T CART trees ;
$y_i = (y_{1i}, ..., y_{ni})^T$: a classification vector associated with C_i on Ω_V ;
Ω_V: a validation or selection sample;
ch_i: chromosome i of the search space;
$ff_E(ch, \Omega_V)$: fitness function ;
Output:
Ch_sol : Solution chromosome;
Begin
Generate a population of bits of size T;
Evolve the population where the fitness of a chromosomch i is calculate by $ff_E(ch_i, \Omega_V)$;
ch_sol :=$argmax_{ch_i}(ff_E(ch_i, \Omega V))$;
End.

6 Experiments and Results

In this section, we describe information about the datasets used to carry out our experiments. We test 7 benchmarcks datasets taken from the UCI Repository [9] as depicted in Table 1. The value of the parameter k is fixed to $\sqrt{m}$ [24] (m represents the number of descriptors).

The datasets are split into two samples: a sample for learning and pruning denoted Ω_L (80% of the initial sample size), and a test sample to compute the perfor-

Table 1: Datasets description and the used values of the parameter k

$Dataset$	$Numb.Inst$	$Features$	$Classes$	k
$Gamma$	1920	10	2	$\sqrt{10}$=3
$Letter$	2000	16	26	$\sqrt{16}$=4
$Pendigits$	10992	16	10	$\sqrt{16}$=4
$Segment$	2310	19	7	$\sqrt{19}$=4
$Spambase$	4610	57	2	$\sqrt{57}$=7
$Vehicle$	946	18	4	$\sqrt{18}$=4
$Waveform$	5000	40	3	$\sqrt{40}$=6

mance in generalization or the rate of success (20% remaining).

An initial set of 300 trees is generated first composing the Random Forest (RF), then the pruning methods are applied to RF ensemble in order to eliminate irrelevant trees based on the f_E function.

The comparison is made between the proposed methods AG_{fE}, FS_{fE}, the SFS_A method [11] and the initial RF against two criteria: performance or success rate in generalization using the test sample Ω_T and the size of the sub-ensemble obtained after pruning. From table 2 we

Table 2: Success Rates of ensembles obtained by FS_{fE}, AG_{fE}, SFS_A and RF for all the datasets

	FS_{fE}	AG_{fE}	SFS_A	RF
$Gamma$	88.56%	**90.72%**	88.93%	87.81%
$Letter$	96.98%	**97.01%**	96.93%	95.91%
$Pendigits$	99.42%	99.40%	**99.59%**	98.99%
$Segment$	99.24%	**99.54%**	99.34%	97.51%
$Spambase$	**96.88%**	**96.88%**	96.67%	94.78%
$Vehicle$	85.62%	85.68%	**85.71%**	73.21%
$Waveform$	89.92%	**90.56%**	89.84%	86.00%
$AverageSR$	0.9380	0.9426	0.939	0.906

note that the FS_{fE}, AG_{fE} and SFS_B methods have a better accuracy compared to the initial RF forest. On average, over all 7 datasets FS_{fE}, AG_{fE} and SFS_A improve the accuracy of the forest with3.2% and 3.6% and 3.3% respectively. FS_{fE} do better than SFS_B in 3 cases with a minimum improvement of 0.05% and a maximum improvement of 0.21%. $AG_{fE}$is better than SFS_A in 5 cases; i.e. more than 70% of cases with a minimum improvement of 0.08% and a maximum improvement of 1.79%.

The results of the comparative study based on the size of the ensembles obtained by the three methods of pruning are given in Table 3: Of the 7 datasets, the FS_{fE} method allows obtaining ensembles sub-ensembles of reduced size compared to SFS_A for 5 benchmarcks; the reductions varying between between 3 and 14 trees. For the remaining benchmarks, the two methods obtain the same reduction in one, and for the last one is in favor of SFS_A; the reduction being of 22 trees. For the AG_{fE} method, we note that a path using genetic al-

Table 3: Ensembles sizes obtained by FSfE, $AG_{fE}$and $SFS_A$for all datasets

	FS_{fE}	AG_{fE}	SFS_A
$Gamma$	**65**	88	79
$Letter$	**98**	120	**98**
$Pendigits$	54	72	**32**
$Segment$	**12**	30	15
$Spambase$	**26**	29	31
$Vehicle$	**17**	20	25
$Waveform$	72	**70**	86
$AverageSZ$	49.14	61.29	52.29

gorithms explores more possibilities; therefore we get ensembles larger than those generated by the two other methods except for the datasets Spambase and Waveform; the reductions being of 2 and 16 trees compared to SFS_A.

FS_{fE} gives the best average size over all the datasets with average reductions of 12.14 and 3.14 against AG_{fE} and SFS_A respectively. Fnally, compared with the initial RF ensemble, the three pruning methods are better in 100% of the cases with average reductions of 83.62%, 79.57%, and 82.57% for FS_{fE}, AG_{fE}, and SFS_A for the 7 datasets.

To better analyze the data, we use the comparison approach proposed by [21] which consists to assign a rank to each compared method.

For the success rate, AG_{fE} is first with an average

Table 4: Ranking FS_{fE}, AG_{fE} and SFS_A based on success rate obtained for all datasets

	FS_{fE}	AG_{fE}	SFS_A
$Gamma$	3	1	2
$Letter$	2	1	3
$Pendigits$	2	3	1
$Segment$	3	1	2
$Spambase$	1	1	3
$Vehicle$	3	2	1
$Waveform$	2	1	3
$AverageRK$	2.29	1.43	2.14

rank of 1.43, followed by SFS_A with 2.14, and FS_{fE} with 2.29 lastly. For the ensembles size, FS_{fE} is first

Table 5: Ranking FS_{fE}, AG_{fE} and SFS_A based on size obtained for all datasets

	FS_{fE}	AG_{fE}	SFS_A
$Gamma$	1	3	2
$Letter$	1	3	1
$Pendigits$	2	3	1
$Segment$	1	3	2
$Spambase$	1	2	3
$Vehicle$	1	2	3
$Waveform$	2	1	3
$AverageRK$	1.29	2.43	2.14

with 1.29, followed by SFS_A with an average rank of

2.14, and finally AG_{fE} with 2.43.
The search time is an important parameter to use for the comparison. We noticed during the experiments that the AG_{fE} method takes much more time than the FS_{fE} method for research because genetic algorithms explore more possibilities.

7 Conclusion

The disadvantage of random forest methods lies in the loss of intellegibility of the model provided, composed of a large number of distinct trees and therefore more difficult to synthesize and submit to human expertise. Selection methods can address this problem of loss of intelligibility by simplifying the forest structure and enabling storage space savings and response time for time-constrained application domains.

In this paper, we used a diversity-based heuristic measure to simplify a random forest ensemble. The measurement employs two search strategies: SFS path (Sequential Forward Selection) and genetic algorithms based path. These search strategies are compared to show which one explores better the search space.

The experimental results show the measure effectiveness to search ensembles of reduced size and performances equal or sometimes exceeding those of the initial forest ensemble as well as the SFS_A method proposed in [11] using accuracy as measurement. Furthermore, the two methods allow obtaining ensembles of different sizes generally smaller for a SFS path with generally reduced performances compared to that obtained with genetic algorithms based path.

As future work, in order to improve the performances, we propose to search the optimal ensembles of trees using for example optimal methods such as Branch and Bound method [35] or near optimal methods such as Genetic Algorithms. We will also apply the measure (based on diversity and accuracy simultanously) in bagging ensembles [36][37] for forest pruning. Finally, we plan to experiment our approach in the field of digestive diseases detection.

References

[1] *Stochastic attribute selection committees*, 1998.

[2] *Limiting the number of trees in random forests*, 2001.

[3] *Exploring Support Vector Machines and Random Forests for spam detection*, 2004.

[4] *A hybrid network intrusion detection technique using random forests*, 2006.

[5] *Heuristic Based Improvements for Effective Random Forest Classifier*, 2012.

[6] Adnan, M. and Islam, M. Optimizing the number of trees in a decision forest to discover a subforest with high ensemble accuracy using a genetic algorithm. *Knowledge-Based Systems*, 110:86–97, 2016.

[7] Ali, K. Learning probabilistic relational concept descriptions. *Phd Thesis, Dept of Info and Computer Science, Univ. of California, Irvine.*, 1996.

[8] Amit, Y. and Geman, D. Shape quantization and recognition with randomized trees. *Neural Computation*, 9:1545–1588, 1997.

[9] Asuncion, A. and Newman, D. Uci machine learning repository. 2007.

[10] Bernard, S., Adam, S., and Heutte, L. Dynamic random forests. *Pattern Recognition Letters*, 33(12):1580–1586, 2012.

[11] Bernard, S., Heutte, L., and Adam, S. On the selection of decision trees in random forest. *IJCNN*, pages 302–307, 2009.

[12] Bogler, P. Shafer-dempster reasoning with applications to multisensor target identification. *1EEE Trans. Sys. Man. Cyb.SMC*, 17:968–977, 1987.

[13] Breiman, L. Bagging predictors. *Machine Learning*, 2:123–140, 1996.

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

[15] Breiman, L. and Cutler, A. Random forests. 2005.

[16] Breiman, L., Friedman, J., Olshen, R., and Stone, C. *Classification And Regression Trees*. 1984.

[17] Brostaux, Y. *Etude du classement par forets aléatoire déchantillons perturbés á forte structure dâinteraction*. PhD thesis, 2005.

[18] Buchanan, B. and Shortliffe, E. Rule based expert systems. *Addison-Wesley, Reading, Massachusetts*, pages 288–291, 1984.

[19] Cutler, D., Edwards, T. J., Beard, K., Cutler, A., Hess, K., Gibson, J., and Lawler, J. Random forests for classification in ecology. *Ecology*, 88:2783–2792, 2007.

[20] Davis, L. Handbook of genetic algorithms. *Van Nostrand Reinhold,New York*, 1991.

[21] Demsar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[22] Dheenadayalan, K., Srinivasaraghavan, G., and Muralidhara, V. Machine learning and data mining in pattern recognition. pages 516–529, 2016.

[23] Dietterich, T. and Kong, E. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. *Technical Report, Dept of Computer Science, Oregon State University, Covallis, Oregon*, 1995.

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

[25] Ho, T. Random decision forests. *Proc. Third Intâl Conf. Document Analysis and Recognition*, pages 278–282, 1995.

[26] Koprinska, I., Poon, J., Clark, J., and Chan, J. Learning to classify e-mail. *Information Sciences*, 177(10):2167–2187, 2007.

[27] Kouzani, A., Nahavandi, S., and Khoshmanesh, K. Face classification by a random forest. *Tencon, IEEE Region 10 Conference*, pages 1–4, 2007.

[28] Kuncheva, L. and Whitaker, C. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51:181–207, 2003.

[29] Kwok, S. and Carter, C. Multiple decision trees. *Uncertainty in Artificial Intelligence 4, ed. Shachter, R., Levitt, T., Kanal, L., and Lemmer, J., North-Holland*, pages 327–335, 1990.

[30] LariviÃ¨re, B. and Poel, D. Predicting customer retention and profitability by using random forests and regression forests techniques. *Expert Systems with Applications*, 29:472–484, 2005.

[31] Nan, F., Wang, J., and Saligrama, V. Pruning random forests for prediction on a budget. *.In NIPS*, 2016.

[32] Salzberg, S. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and knowledge discovery*, 1:317–327, 1997.

[33] Shlien, S. Multiple binary decision tree classifiers. *Pattern Recognition*, pages 757–763, 1990.

[34] Shlien, S. Nonparametric classification using matched binary decision trees. *Pattern RecognitionLett.*, 13:83–87, 1992.

[35] Somol, P., Pudil, P., and Kittler, J. Fast branch and bound algorithms for optimal feature selection. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 26,7:900–912, 2004.

[36] Taleb Zouggar, S. and Adla, A. A new function for ensemble pruning. *Dargam et al. (Eds): ICDSST 2018, LNBIP 313, Springer International Publishing AG*, pages 181–190, 2018.

[37] Taleb Zouggar, S. and Adla, A. A diversity-accuracy measure for homogenous ensemble selection. *International Journal of Interactive Multimedia and Artificial Intelligence*, 5(5):63–70, 2019.

[38] Tripoliti, E., Fotiadis, D., and Manis, G. Dynamic construction of random forests: Evaluation using biomedical engineering problems. *IEEE Society*, 2010.

[39] X., J., C., W., and H., G. Forest pruning based on branch importance. *Computational Intelligence and Neuroscience*, 2017.

[40] Yang, F., Lu, W., Luo, L., and Li, T. Margin optimization based pruning for random forest. *Neurocomputing*, 94:54–63, 2012.