LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR STATISTIK

Malte Nalenz and Thomas Augustin

# Cultivated Random Forests: Robust Decision Tree Learning through Tree Structured Ensembles

Technical Report Number 240, 2021
Department of Statistics
University of Munich

http://www.statistik.uni-muenchen.de

# Cultivated Random Forests: Robust Decision Tree Learning through Tree Structured Ensembles

Technical Report

Malte Nalenz
Department of Statistics
Ludwig-Maximilians-University
Munich

Thomas Augustin
Department of Statistics
Ludwig-Maximilians-University
Munich

**Abstract**

We propose a robust decision tree induction method that mitigates the problems of instability and poor generalization on unseen data. In the spirit of model imprecision and robust statistics, we generalize decision trees by replacing internal nodes with two types of ensemble modules that pool together a set of decisions into a soft decision: (1) option modules consisting of all reasonable variable choices at each step of the induction process, (2) robust split modules including all elements of a neighbourhood of an optimal split-point as reasonable alternative split-points. We call the resulting set of trees *cultivated random forest* as it corresponds to an ensemble of trees which is centered around a single tree structure, alleviating the loss of interpretability of traditional ensemble methods. The explicit modelling of non-probabilistic uncertainty about the tree structure also provides an estimate of the reliability of predictions, allowing to abstain from predictions when the uncertainty is too high. On a variety of benchmark datasets, we show that our method is often competitive with random forests, while being structurally substantially simpler and easier to interpret.

## 1 Introduction

Decision trees are one of the most common prediction methods. Their popularity mostly stems from their interpretability and methodological simplicity. Decision tree successively partition the covariate space into smaller subspaces that are purer with respect to the target values $y$. Most practical algorithms, such as CART [6] and C45 [27], use a greedy procedure that chooses the covariate and split-point with the largest gain in purity at each step. Decision trees are adaptive to arbitrary underlying functions and can perform well in several domains. A major downside of decision trees is their instability with respect to small perturbations of the training data, see already [4]. Slight changes in the training set can lead to entirely different tree structures, raising suspicion about the validity of their implied interpretations as well as their generalizibility to unseen data.
The instability can be traced back to the all-in decision at each node [8]. Adding or removing observations might lead to the choice of a different splitting point or even different variable to split on. Through the recursive structure, all decisions depend on the

previous ones. Thus small changes in the top layer of the tree can lead to dramatically different subtrees. Decisions can only partially be reversed post-hoc through pruning away dubious subtrees, making individual choices very influential.
Ensemble methods such as bagging [4], random forests [5] and boosting [16] solve the instability and generalizibility issues at the cost of giving up the interpretational simplicity: Instead of a single tree model, a sequence of trees is generated, each built on alterations of the original data. The final prediction is then the a combination of these individually weak decision trees.
In this article, we take a conceptually different approach. Instead of trying to find an optimal single tree or generating an ensemble of multiple decision trees, we model the uncertainty about the tree structure directly. To this end we introduce ensemble modules that pool a set of decisions into a soft decision. Ensemble modules capture the uncertainty about both the variable to use and the choice of an exact splitting position. The resulting model, that we call *cultivated random forest* (CRF), corresponds to an ensemble of trees, carrying over the desired stability and generalizability of ensemble methods. However, through a notion of neighbourhood interpretability is preserved. In many domains such as the clinical, the ability to inspect what a prediction is based upon is crucial in order to reveal spurious or nonsensical relationships [9], potential gender and racial biases [10] and give practitioners the option to intervene with the decision system in a guided way. This is also important to build acceptance from practioners. Additionally, through the framework of model imprecision, CRF is able to give an estimate of the reliability of predictions, that can be used to abstain from a predictions, if the uncertainty is too high. This is especially important, when the decision system is integrated in a larger work-flow and also alternative means of decision exist such as domain experts. With this, CRF offers a good trade-off between high accuracy, interpretability and accountability.
In section 2 basic notations and principles of decision tree learning are recalled. In section 3 we introduce ensemble modules and the resulting CRF model. Benchmark results on several binary classification datasets are shown in section 4, and section 5 concludes.

# 2 Decision tree learning

**Decision Trees.** Decision trees use a graph of decision rules to map a $p$-dimensional covariate vector $\mathbf{x}_i = (x_{i,1}, ..., x_{i,p})$ to a decision about the target value $y_i$. In the following all covariates are assumed to be numeric and the target to be binary thus $y \in \{0, 1\}$. Trees consist of a root node, a set of internal nodes and a set of leaf nodes. Starting from the root node where the whole dataset $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ consisting of $N$ covariate vectors is present, subsets of observations are moved to its childnodes based on decision rules, recursively partitioning $\mathcal{X}$ into smaller rectangles. Here only univariate binary decisions of the form $d(\mathbf{x}, t_0, j) = I(x_j \leq t_0)$ are considered, thus each decision leads to exactly two childnodes. For ease of notation $d(\mathbf{x}, t_0, j)$ will be in the following oversimplified as $d(\mathbf{x})$ and assumed that $t_0$ and $j$ are attached.
In this article, we also utilize the idea of fractional observerations [27, 29]: if a decision can not be made with certainty, observations are split up into fractions and moved to both childnodes. Each observation is attached a value $w_{i,l} \in [0, 1]$ that represents the fraction of the $i$'th observations that is present in the $l$'th node. Once a leafnode is reached, a decision is made based on its attached prediction value $\hat{y}$ for the target variable $y$, typically either the majority class or class distribution. A leaf node can be written as a

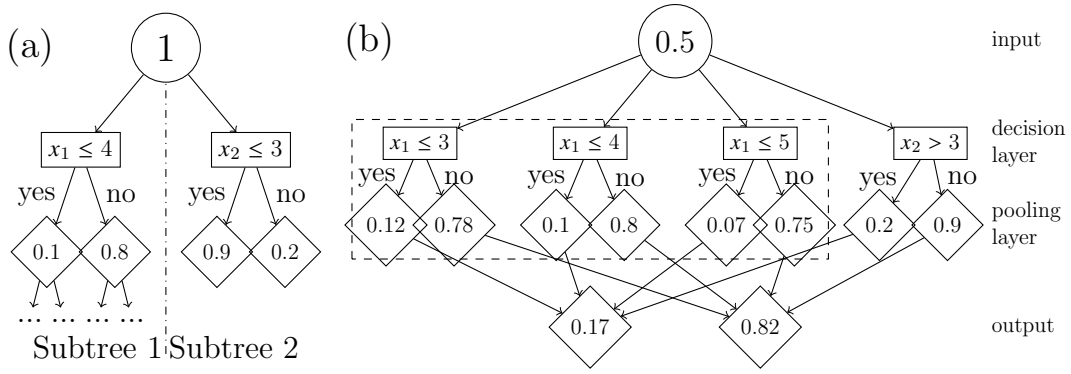


Figure 1: Option Tree vs. Ensemble module. Decisions are shown as rectangles, implied current leaf nodes as diamonds. The numbers inside the diamonds reflect the target distribution $p(y = 1|\mathbf{x})$. (a) Option Tree. Different subtrees follow from each option split (b) Ensemble module, consisting of a set of decisions followed by pooling of the resulting leafnodes with similar target distribution. The decision nodes inside the dashed rectangle can be summarized as a single robust split module.

product of decision rules by $\mathcal{L}(\mathbf{x}, \mathcal{D}_\mathcal{L}) = \prod_{d \in \mathcal{D}_\mathcal{L}} d(\mathbf{x})$, where $\mathcal{D}_\mathcal{L} = \{d_1, ..., d_l\}$ is the path of $l$ decisions, that have to be fulfilled [1] to reach this node [17].

**Tree induction.** Given a training sample where we are given $\mathcal{X}$ and the labels $y$, the goal is to build a decision tree that is able to classify unseen samples. Many different tree inducers have been proposed, for an overview we refer to [30]. Usually, at each step during training the decision that minimizes some measure of impurity in the implied childnodes is chosen and applied to the data points, partitioning the training data. In this article, the Gini impurity is used as in CART [6]. This recursive process is repeated until no split reduces impurity further or a stopping criterion is reached.

**Decision tree instability.** Instability in decision tree learning is a well known problem [20, 4]. At each node a single decision is required, while there can be considerable uncertainty about the correct choice. This dilemma leads to a high degree of instability. In this article, we focus on:

*Variable uncertainty:* At each node, a binary decision tree needs to decide on exactly one covariate for further partitioning. If the implied purity of several covariates is similar, this all-in approach neglects the uncertainty about our choice.

*Splitting point uncertainty:* Given the covariate to split on, a cut-point $t_0$ needs to be chosen. If the impurity surface is flat, a sharp decision is not justified. This uncertainty translates to a lack of smoothness that is found in decision trees [20].

Other sources of uncertainty include paramter uncertainty, such as the correct maximum tree depth, and the choice of the best subtree. This is typically addressed through pruning techniques. Good overviews can be found in [23, 30].

**Ensemble methods and option trees.** Random forests address the aforementioned stability issues with the bagging of randomized trees. In the standard version each tree is build independently on a bootstrap sample of $\mathcal{X}$ using only a subset of covariates. The predictions of this sequence of trees are combined through averaging or voting. Ensemble

[1] If the path to this leaf node goes to the left side at a node $j$ we take $d_j(\mathbf{x})$ into $\mathcal{D}_\mathcal{L}$ while we take $1 - d_j(\mathbf{x})$ if it goes to the right. More details can be found in the supplementary materials.

methods work well in practice because they reduce several problems of single decision trees: they introduce smoother decision boundaries, mitigate the variable selection uncertainty and lead to better generalization performance.
Another approach to address the variable uncertainty is to use multiple decisions at each node if the implied purity is similar. This was first introduced by Buntime [8] as option trees. In Fig.1 (a), an option tree is shown. At each node several decisions are allowed. The resulting subtrees are subsequently grown and evaluated separately and the final prediction is an aggregate over all subtrees. The option tree approach was combined in [15] with boosting into the alternating decision tree (ADT) model. ADTs were shown to posses decent predictive performance and relatively small model sizes. An interesting property is that option trees and ADT can be seen as structurally sparse representation of an ensemble. As a part of the structure is shared by all subtrees, a whole ensemble of trees can be described by a single tree structure [14].

# 3 Cultivated Random Forest

In this work, ensemble learning is viewed from the point of robust statistics and model imprecision. Instead of a single model, we are looking at a set of models that correspond to a set of different choices in the model construction process. Typically, in robust statistical models, these are distributional assumptions or priors in the Bayesian setting. In the context of decision trees, model imprecision was applied to the probability distributions in the leafnodes to robustify entropy based splits [22, 2]. Here we instead use this framework to express our uncertainty about the structure of the tree itself and capture the uncertainty involved with the choices made during the tree induction process. To this end, we generalize the decision tree model by replacing internal nodes with *ensemble modules* $\mathcal{M} = \{d_1, ..., d_h\}$ that consist of the set of $h$ decisions that are reasonable at a given step of the induction process. To preserve the binary tree structure, the decisions are then pooled and observations split up into fractions. Usually the left child node is the True part of the decision rule. For ensemble modules we require the decisions to be directed as in [32]. In binary classification we define the right childnode to have the higher implied target probability $p(y = 1|x)$. For multinomial and other target distributions, more sophisticated merging algorithms are required, as in [26][31]. For the case that all decisions are weighted equally and we use the average as pooling function, the fraction going to the left childnode is given by $\psi(\mathbf{x}, \mathcal{M}) = |\mathcal{M}|^{-1} \sum_{d \in \mathcal{M}} d(\mathbf{x})$. If only a fraction of an observation is present in this leaf, we simply take fractions of this fraction.
The whole process is shown in Fig. 1 (b). A set of decisions is considered inside the ensemble module and then pooled into two child nodes. Note that the decision $x_2 > 3$ is directed, such that the right childnode has the higher target probability $p(y = 1|x)$. The name ensemble module stems from the insight that the fraction of an observation to be present in a given leaf node can be written by replacing the $l$ binary decisions in the path to a leafnode with $l$ ensemble modules $\mathcal{M}_{\mathcal{L}} = \{\mathcal{M}_1, ..., \mathcal{M}_l\}$, and pooling after each ensemble module, as

$$\mathcal{L}(\mathbf{x}, \mathcal{M}_{\mathcal{L}}) = \prod_{\mathcal{M} \in \mathcal{M}_{\mathcal{L}}} \left( \frac{1}{|\mathcal{M}|} \sum_{d \in \mathcal{M}} d(\mathbf{x}) \right) = \frac{1}{|D_\times|} \sum_{D_{\mathcal{L}} \in D_\times} \left( \prod_{d \in D_{\mathcal{L}}} d(\mathbf{x}) \right) = \frac{1}{|D_\times|} \sum_{D_{\mathcal{L}} \in D_\times} \mathcal{L}(\mathbf{x}, D_{\mathcal{L}}) \tag{1}$$

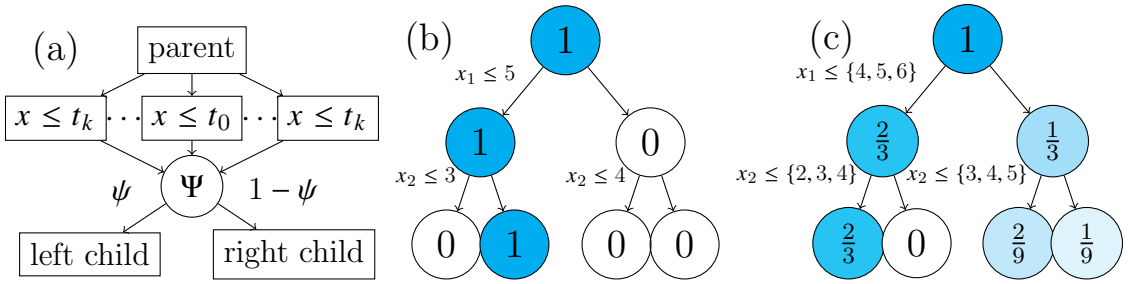


Figure 2: (a): Compact representation of a robust split module. (b,c) Example Classification of $(x_1, x_2) = (5, 4)$ assuming $\phi = 1$. (b) standard binary tree, (c) tree with robust split module. Nodes are drawn as circles. Numbers and colouring represent $w_{i,l}$, the fractions of the example observation that reach each node.

with $\mathcal{D}_\times = \{\mathcal{M}_1 \times ... \times \mathcal{M}_l\}$. So in fact, by pooling the decisions at each ensemble module, the fractional observations in each leafnode can be written as average over an ensemble of trees that is spanned by the Cartesian product of the ensemble modules. The derivation can be found in the supplementary materials. By that CRF is a structurally sparse representation of an ensemble of size $|\mathcal{D}_\times|$. The main difference to random forests is that the trees in the ensemble are not grown independently, but instead are chosen as all reasonable choices along the induction process. Individual trees are therefore not constructed as weak learners that are decorrelated in order to improve the final classifiers generalizibility. On the contrary, the trees share the largest part of the tree structure with other trees in the ensemble but deviate on average just in a few decisions. We expect that generalization performance will be lower in certain domains where the decorrelation aspect of random forest is important to capture all underlying mechanisms present in the data. But we argue that in many applications this aspect is overcompensated by the simplification and resulting interpretability. We introduce two types of ensemble modules, that will be defined formally in the next sections:

- *Option modules* that consist of all reasonable variable choices at the each respective step.
- *Robust split modules* that given the best split-point with respect to impurity for each covariate also consist of reasonable alternative thresholds.

Both module types of ensemble modules can be combined. This is shown in Fig.1 (b) by replacing the single decision that is part of the option module $x_1 \leq 4$ with a robust split module.

**Robust split module.** [7] show that an ensemble of decision trees using bagging without replacement can be described as a neighborhood around the optimal split-point. Imagine the simple example where we are given one numeric covariate $x$ with associated labels $y$. The underlying true function is $y = I(x > t_0)$, however $x$ is observed with noise. Depending on the degree of noise and the sample size, when using bootstrap samples of the original data, the decisions will be distributed around the true split point $t_0$. In this simple example the ensemble can be summarized as $P(t_o)$, where $P$ is a unknown distribution function. Therefore, it should be possible to describe large parts of a bagged tree ensemble through the neighbourhood of splits.
The splits found in our induction process are unlikely to be optimal, so the theoretical results from [7], just discussed, can be only understood as a heuristic. As no prior

knowledge about the split's distribution is available we choose a non-parametric approach and use the closest points in the covariate space to construct a neighbourhood $\mathcal{T}(t_0)$ around $t_0$. Let $(x_{(i)}, w_{(i)})$ be the $i$'th *ordered* covariate value and its fraction present in the current node to split, then we define the robust split module as

$$\mathcal{T}(t_0) = \{t_{-j} = x_{(i-j)} \leq \ldots \leq t_{-1} = x_{(i-1)} \leq t_0 = x_{(i)} \leq t_1 = x_{(i+1)} \leq \ldots \leq t_m = x_{(i+m)}\}$$

with

$$j = (\arg\max_{\tilde{j}} \sum_{q=i-\tilde{j}}^{i-1} w_{(q)} < k) + 1 \qquad ; \qquad m = (\arg\max_{\tilde{j}} \sum_{q=i+1}^{i+\tilde{j}} w_{(q)} < k) + 1.$$

Intuitively, on both sides we take $k$ "full" observations into the set. This expresses our assumption that we can not be too sure about the exact position of the split and should also consider all slightly different splits as equally likely, mimicking the behaviour of bagging. As split-points are constituted by observations, fractional observations directly imply fractional split-points. Let $\phi(t)$ denote the weight that can be interpreted as the "representation strength" defined by the point mass of the data points in the current leaf for a cut-point $t$, given by the recursive function

$$\phi(t) = \begin{cases} \sum_{i=1}^{N} w_{(i)} I(x_{(i)} = t), & \text{if } t \notin \{t_{-j}, t_m\} \\ k - \sum_{l=1}^{j-1} \phi_(t_{-l}), & \text{if } t = t_{-j} \\ k - \sum_{l=1}^{m-1} \phi_(t_l), & \text{if } t = t_m. \end{cases} \tag{2}$$

In (2) an exception is made for the boarder cases $t_{-j}$ and $t_m$. As those often can not be included fully they are simply asssigned the remaining of $k$ on this side. This neighbourhood is constructed such that $\sum_{t \in \mathcal{T}(t_0)} \phi(t) \leq 2k + \phi(t_0)$ in each robust split module. For an observation reaching a robust split module $l$, the fraction that is moved to the left side is given by the gating function

$$\psi(\mathbf{x}, \mathcal{T}(t_0)) = \frac{1}{\sum_{t \in \mathcal{T}(t_0)} \phi(t)} \sum_{t \in \mathcal{T}(t_0)} \phi(t) I(x \leq t).$$

The fraction present in the left childnode is then $w_{(i),left} = w_{(i)} \psi(x_{(i)}, \mathcal{T}(t_0))$ and the fraction in the right childnode $w_{(i),right} = w_{(i)}(1 - \psi(x_{(i)}, \mathcal{T}(t_0)))$. This directly implies that for each observation the sum of the fraction over all (current) leaf nodes equals 1 at each moment in training and prediction. Cases close to the decision boundary will be present in both childnodes for further training. This is shown in Fig.2 for an example data point. Instead of being present in only one node, the data point is present in three current leafnodes. This reflects our uncertainty as the observation is close to the decision boundaries and slightly different model choices would have let to different decision.
During induction, the exact position of $t_0$ will therefore not influence the tree structure substantially, leading to more stable structures, as we withdraw from making a definite decision at this point. Importantly, as $\mathcal{T}$ is centered around $t_0$, the interpretation of the robust split module is similar to a common binary decision. When looking at a node instead of the sharp interpretation 'if $x_j \leq t_0$' we can interpret each robust split module as 'if $x_j$ is less than around $t_0$' ($x_j \lessapprox t_0$). This offers a nice trade-off between smoothness and interpretability. The parameter $k$ controls the degree of smoothness that

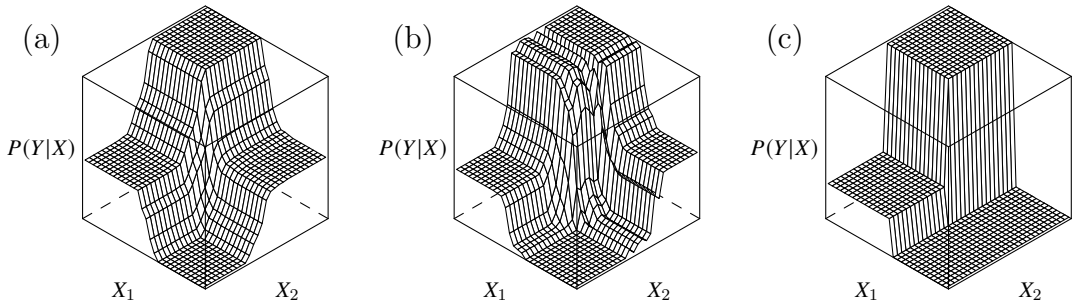


Figure 3: Decision Surface for the simulated dataset of (a) The first split using CRF, (b) full random forest model (c) CART tree with depth 2. Details can be found in the supplementary materials.

we enforce on the model. In the extreme case of $k = N$ the gating function $\psi$ equals the univariate empirical cumulative distribution function. Our choice is motivated by the results in [7], who show that split-points using bagging without replacement lie within a $N^{1/3}$ neighborhood around the optimal $t_0$. Due to the suboptimality of found splits in practice, we found a slightly bigger neighbourhood of $k = (\sum_{i=1}^{N} w_{(i)})^{1/2}$ to work well. Note that $k$ is not optimized in the tree induction.

**Option module.** To address the variable uncertainty, we introduce option modules, similar to option trees and ADT. Let $h_j$ denote the weighted impurity using covariate $j$ for splitting and $h_{min}$ the minimal impurity value found in the current step. Then, for a given threshold $\eta_l$, for the next split all covariates $x_j$ with $\{j|h_j \leq h_{min} + \eta_l\}$ are taken into the set as reasonable options. As the decisions made in the top layers of the tree are the most influential on the tree structure, the parameter $\eta_l$ is set to diminish with increasing depth, here by $\eta_l = \eta_0/{s_l}^3$, where $s_l$ is the tree depth in node $l$ and $\eta_0$ is a pre-specified parameter. Thus in the top layers more covariates are taken into option modules, while in the bottom layers extra covariates are only added if the decision is very tight.[2] Option modules can easily be combined with robust split module using robust split modules instead of single splits for each covariate. $h_j$ can then be set as average impurity of all elements in $\mathcal{T}(t_0)$. This approach shares similarity with the idea of inner ensembles, where bootstrap samples are used to decide on the best next *single* split [1, 21]. $\phi(\mathcal{T}(t_0))$ is then normalized to sum up to one, to give each covariate the same weight. The combination of the two types of modules is shown in Fig.3 on simulated data, where the two classes are drawn from 2-dimensional mixture of normals. In this example, CRF approximates the behaviour of random forest quite well with respect to the decision surface and the smoothness that is introduced. The smoothness stems from the robust split modules. Also both covariates are used in the first split, as the decision is tight. In contrast to multivariate split methods, such as oblique trees [25], the decisions are not jointly optimized. This can be seen as a

[2]We also tested using Hoeffding bounds to select alternative covariates [12, 24].The Hoeffding bound is inversely proportional to $n$ and in our experiments too few covariates were considered in the top levels of the tree and too many in the bottom levels. However a more theoretically motivated choice of $\eta$ that expresses the trade-off between reflecting all choices and the reduced interepretability would still be desirable.

form of regularization, as the decisions are required to be individually predictive. The decision surface for CART in Fig.3 (c) shows the lack of smoothness in standard decision trees, making it impossible to capture the underlying relationship in this simulation.

**Predictions.** For a leafnode the attached prediction value is set as the mean of $\mathbf{y}$ weighted by its fractions present in this node $\hat{y}_j = (\sum_{i=1}^{N} w_{i,l})^{-1} \sum_{i=1}^{N} w_{i,l} y_i$. At prediction time, test cases will have non-zero weights in several leafnodes. The output from our model for a given observation $i$ and the set of $m$ leafnodes is the set $\check{\mathbf{y}} = \{(\hat{y}_1, w_{i,1}), ..., (\hat{y}_m, w_{i,m})\}$. For obtaining a real valued point estimate, we can simply use the weighted average over the set $\check{\mathbf{y}}$ with $p(y|x_i) = \sum_{j=1}^{m} w_{i,j} \hat{y}_j$. Note that the size of $\check{\mathbf{y}}$ is the number of leafnodes, not the number of trees that are represented by our ensemble. For interpretation, this allows to look at the leaf nodes with the highest fractions and have a symbolic description, what the prediction is based on. It can also be informative to look at the spread of $\check{\mathbf{y}}$ as a measure for reliability of this prediction. A natural measure is the variance of the fractional predictions. The reliability reflects the degree of conflict between the decisions in the ensemble modules. If an observation is often close to the decision boundary, or if decisions inside option modules are contradictory, the prediction is found unreliable. Consider the example where half the fractions of an observations fall into leafs with prediction value 0 and the other half into leafs with prediction value 1. Both the variance and the final prediction for this observation will be 0.5. This prediction shows a high degree of uncertainty in two layers: about the predicted value $p(y|x) = 0.5$ and given $p(y|x)$ the stochastic uncertainty about the outcome. On the other hand, if an observation always falls into leafs with predicted values of 0.5, we are quite certain that we should predict 0.5 and the uncertainty concerns only the outcome. A nice property for set-valued predictions is the option to abstain from a prediction if the uncertainty is too high [11]. This is important in practice, when a high cost is associated with a wrong prediction. For example in clinical applications it might be better to remeasure covariates in case of potential measurement error or consider a further test altogether if the model prediction is unreliable for a given patient. Also expert knowledge should be taken into account for uncertain predictions, if available.

# 4 Experiments

To test the predictive capabilities of our proposed method, we carried out 10-fold cross validation on two sets of data sets:

**Gene expression data.** The goal is to predict a binary disease outcome, based on the genetic expression profile. These data sets are characterised by an extreme $p >> n$ situation with thousands of covariates and small sample sizes.

**Binary classification benchmark datasets.** The datasets are taken from the UCI repository [13]. Data are coming from various domains, including small and medium sized data sets with varying number of covariates, and varying degrees of target imbalance.

**Evaluation settings.** All datasets together with seeds (that were generated randomly for the experiments) and a data description are available in the supplementary material to enhance an easy reproducibility. As some of the datasets are imbalanced, we chose the area under the ROC curve (AUC) as evaluation metric. We compare CRF against random forest as the ensemble benchmark and to CART as the baseline for an interpretable model. All methods were run with standard settings from the R-libraries randomForest and rpart respectively [28]. We test 4 different versions of CRF:

| | Dataset | random forest | CRF-full | CRF-split | CRF-option | CRF-shallow | CART |
|---|---|---|---|---|---|---|---|
| Gene Expression | Colon [3] | 0.850 | **0.887** | **0.874** | 0.808 | **0.887** | 0.859 |
| | DLBC [18] | 0.951 | **0.955** | 0.901 | 0.78 | **0.955** | 0.721 |
| | Leukaemia [19] | **1** | **1** | **1** | **1** | **1** | 0.838 |
| | Prostate [18] | 0.955 | **0.968** | 0.938 | 0.847 | **0.968** | 0.851 |
| Benchmark Data | Australian Credit | 0.932 | **0.936** | 0.921 | **0.928** | **0.937** | 0.903 |
| | Banknote | 1.000 | **0.998** | **0.998** | **0.999** | **0.995** | 0.966 |
| | Blood Transfusion | 0.685 | **0.750** | **0.731** | **0.691** | **0.745** | 0.729 |
| | Climate Model | 0.930 | **0.940** | **0.920** | 0.909 | **0.936** | 0.771 |
| | Diabetes | 0.828 | **0.831** | 0.814 | 0.800 | **0.830** | 0.797 |
| | EEG-Eye-State | 0.985 | 0.935 | 0.940 | 0.934 | 0.794 | 0.724 |
| | Haberman | 0.682 | **0.724** | **0.696** | **0.681** | **0.723** | 0.626 |
| | Indian Liver | 0.752 | **0.749** | 0.724 | 0.709 | **0.749** | 0.667 |
| | Ionosphere | 0.982 | 0.957 | 0.940 | 0.960 | 0.949 | 0.905 |
| | Magic | 0.937 | 0.920 | 0.901 | 0.904 | 0.887 | 0.808 |
| | Parkinsons | 0.980 | 0.950 | 0.949 | 0.932 | 0.953 | 0.890 |
| | QSAR Biodeg | 0.933 | **0.923** | 0.900 | 0.878 | 0.906 | 0.838 |
| | Spambase | 0.986 | **0.980** | 0.971 | 0.973 | 0.962 | 0.894 |
| | SPECTF | 0.850 | 0.837 | 0.764 | 0.724 | **0.841** | 0.721 |
| | Steel Plates | 0.992 | **1.000** | **1.000** | **1.000** | **0.987** | 1.000 |
| | Vertebral | 0.995 | 0.958 | 0.956 | 0.966 | 0.947 | 0.927 |
| | Wisconsin Breast | 0.992 | **0.992** | **0.987** | **0.991** | **0.992** | 0.948 |
| | Wilt | 0.987 | **0.987** | **0.989** | **0.984** | 0.958 | 0.960 |

Table 1: Average AUC using 10-fold CV over 22 Datasets. Results are marked in bold where versions of CRF performs comparably or better than random forest.

1. *CRF-full* with both types of ensemble modules with a max-depth of 14.
2. *CRF-split* using only neighborhood moduleswith a max-depth of 14.
3. *CRF-option* using only option modules with a max-depth of 14.
4. *CRF-shallow* with a max-depth of 6 leading to a maximum of 126 nodes/ensemble modules and both types of ensemble modules.

Each version uses a minimum node size for splitting of 6, a data depenendet parameter $k = \sqrt{n_l}$, where $n_l$ is the sum of weights in node $l$ and $\gamma_0 = 0.3$. All algorithms could be tuned, so we believe it to be a fair comparison to run them with standard settings, especially as random forests are known to be quite robust with respect to the parameter choices. More details about the different implementations can be found in the supplementary materials.

**Predictive performance.** Table 1 shows the AUC of the competing methods using the above setting. All versions of CRF outperform CART on all tested datasets. On 16 out of the 22 tested Datasets CRF-full performs comparably or better than Random Forest, if one is willing to trade off 0.01 in AUC. In 9 Datasets CRF-full shows slightly better performance. Performance is especially good in small data sets, where the uncertainty in the tree induction process is high. Noteworthy is the good performance on the gene expression datasets. CRF is able to account for the extremely high uncertainty due to the small samples and has s generalization performance on par with random forest, whereas CART clearly struggles. Also on difficult and perhaps noisy data sets such as Blood Transfusion and Haberman with overall low AUC values, CRF shows strong performance. Here the decorrelated trees in random forests might become too weak, leading to suboptimal ensemble performance. On 5 of the datasets CRF performs significantly

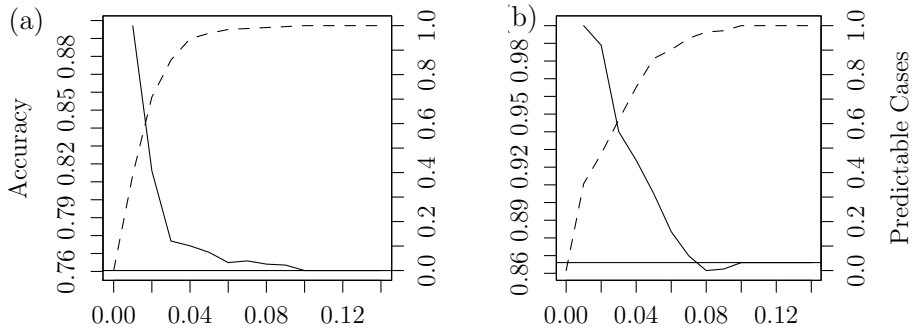


Figure 4: Accuracy, given the option to abstain from a vote, with varying thresholds the prediction spread $\sigma(p(y|\mathbf{x}))$. The dashed line shows the proportion that can be reliably predicted and the full line the accuracy for the predicted cases. (a) Diabetes, (b) Parkinsons.

worse than random forest. A likely explanation is that for these datasets the optimal decision surface is truly multimodal, which CRF in its current state is unable to capture well.

**Influence of treedepth.** CRF-shallow performs almost identical to CRF-full on most datasets. However on some datasets performance drops significantly, implying that deeper trees are necessary in some of the datasets. This result is also interesting, as it suggests that over-fitting is not a huge problem in CRF. Note that no form of pruning is applied on the deeper trees.

**Robust split modules.** Using only the robust split module deteriorates the AUC on most datasets, but still outperforms CART on all tested datasets. On 9 datasets the performance is similar to random forest. Note that CRF-split offers almost the same easy interpretability as a standard decision tree.

**Option modules.** CRF-option still improves on standard CART in most datasets, but the lack of smoothness provided by the robust split modules decreases performance significantly. It is also notable that on some datasets the decrease is quite big, especially the genetic datasets with small sample sizes. The uncertainty about the split-point is here the highest and neglecting it results in worse generalization performance due to instability. Also the setting of $\eta$ and choice of $\eta_0$ might be suboptimal on these datasets.

**Abstaining from predictions.** Fig.4 shows the accuracy if the classifier is given the option to abstain if the prediction spread $\sigma(p(y|\mathbf{x})) > t$ for different thresholds $t$. On the Diatebes and Parkinsons data, abstaining from about 40% of the predictions gives an substantial increase of 8% in accuracy. For the abstained predictions other means of decisions (such as expert opinions) might be better suited, or data recollected in case of possible meassurement error. Together with the interpretability aspect, this makes our method especially well suited for medical and clinical applications.

To summarise, in this section we showed, that CRF can perform remarkably well on many datasets despite restricting the ensemble to a neighborhood around a single tree.

# 5 Conclusion

We introduced a new framework for the construction of tree ensembles: instead of growing a sequence of trees, we construct a set of trees corresponding to reasonable choices in the construction process. The resulting cultivated random forests (CRF) are structurally simpler than regular random forests, which can be beneficial both for diagnosing their validity as well as memory efficiency. The empirical results suggest that CRF is competitive to random forests on many of the tested data sets. An further advantage is that CRF gives an estimate of the reliability of each prediction that can be used to abstain from predictions. This might also be interesting in further research, when building ensembles of randomized CRF, where weak learners can abstain from predictions. We believe that the framework of model imprecision in decision tree learning is well worth exploring further, as it is flexible and the CRF can be generated "on the fly". Next steps include more data adaptive ways to construct neighbourhoods, as well as exploring weighting schemes when pooling covariates in option modules. Also suitable pruning methods should be investigated, as it is likely to limit the complexity further and might lead to an increase in accuracy. A shortcoming of our method, that we will adress in the future, is the implicit assumption of uni-modal impurity surfaces in the covariates. While being beneficial for interpretation this might harm predictitve performance for those data sets, where this assumption does not hold.

# 6 Broader Impact

We believe that our framework of tree structured ensemble learning makes a step towards much needed transparency in machine learning. As machine learning emerges in more and more areas of daily life it affects large parts of society directly. Black-box models may be used to predict insurance claims, calculate credibility scores for credit applicants or in prosecution of potential criminals, with potentially negative consequences for individuals. In our opinion the possibility to give a reasoning behind a prediction should be a minimal requirement in these areas. The same is true in clinical applications, where statistical models might decide on the optimal treatment and consequences of error may be fatal. Here practitioners should need to have the possibility to have insight in the reasoning behind a prediction, in order to challenge its validity. Also the ability to estimate its own reliability becomes more and more important in order to build trust in the predictions made by machine learning models. Here our approach is only a first step and more sophisticated ways should be explored to estimate a model's reliability.

# References


[1] Houman Abbasian, Chris Drummond, Nathalie Japkowicz, and Stan Matwin. Inner ensembles: Using ensemble methods inside the learning algorithm. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 33–48. Springer, 2013.

[2] Joaquin Abellan and Serafin Moral. Maximum of entropy for credal sets. *International journal of uncertainty, fuzziness and knowledge-based systems*, 11(05):587–597, 2003.

[3] Uri Alon, Naama Barkai, Daniel A Notterman, Kurt Gish, Suzanne Ybarra, Daniel Mack, and Arnold J Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the national academy of sciences*, 96(12):6745–6750, 1999.

[4] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

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

[6] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[7] Peter Bühlmann and Bin Yu. Analyzing bagging. *The annals of statistics*, 30(4):927–961, 2002.

[8] Wray Buntine. Learning classification trees. *Statistics and computing*, 2(2):63–73, 1992.

[9] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

[10] Danton S Char, Nigam H Shah, and David Magnus. Implementing machine learning in health care - addressing ethical challenges. *The new england journal of medicine*, 378(11):981, 2018.

[11] Giorgio Corani, Joacquin Abellán, Andres Masegosa, Serafin Moral, and Marco Zaffalon. Classification. In Thomas Augustin, Frank PA Coolen, Gert De Cooman, and Matthias CM Troffaes, editors, *Introduction to imprecise probabilities*, pages 916–954. Wiley, 2014.

[12] Pedro Domingos and Geoff Hulten. Mining high-speed data streams. In *Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining*, pages 71–80, 2000.

[13] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[14] Eibe Frank, Michael Mayo, and Stefan Kramer. Alternating model trees. In *Proceedings of the 30th annual ACM symposium on applied computing*, pages 871–878, 2015.

[15] Yoav Freund and Llew Mason. The alternating decision tree learning algorithm. In *International conference of machine learning*, volume 99, pages 124–133, 1999.

[16] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.

[17] Jerome H Friedman and Bogdan E Popescu. Predictive learning via rule ensembles. *The annals of applied statistics*, 2(3):916–954, 2008.

[18] Enrico Glaab, Jaume Bacardit, Jonathan M Garibaldi, and Natalio Krasnogor. Using rule-based machine learning for candidate disease gene prioritization and sample classification of cancer gene expression data. *PLOS one*, 7(7), 2012.

[19] Todd R Golub, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Coller, Mignon L Loh, James R Downing, Mark A Caligiuri, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[20] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009.

[21] Igor Ibarguren, Jesús M Pérez, Javier Muguerza, Ibai Gurrutxaga, and Olatz Arbelaitz. Coverage-based resampling: Building robust consolidated decision trees. *Knowledge-based systems*, 79:51–67, 2015.

[22] Carlos J Mantas and Joaquín Abellán. Credal-C4.5: Decision tree based on imprecise probabilities to classify noisy data. *Expert systems with applications*, 41(10):4625–4637, 2014.

[23] John Mingers. An empirical comparison of pruning methods for decision tree induction. *Machine learning*, 4(2):227–243, 1989.

[24] Zahra Mirzamomen and Mohammad Reza Kangavari. A framework to induce more stable decision trees for pattern classification. *Pattern analysis and applications*, 20(4):991–1004, 2017.

[25] Sreerama K Murthy, Simon Kasif, and Steven Salzberg. A system for induction of oblique decision trees. *Journal of artificial intelligence research*, 2:1–32, 1994.

[26] John C Platt, Nello Cristianini, and John Shawe-Taylor. Large margin dags for multiclass classification. In *Advances in neural information processing systems*, pages 547–553, 2000.

[27] J Ross Quinlan. *C4. 5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[28] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019.

[29] Brian D. Ripley. *Pattern recognition and neural networks*. Cambridge University Press, 1996.

[30] Lior Rokach and Oded Maimon. Top-down induction of decision trees classifiers-a survey. *IEEE Transactions on systems, man, and cybernetics, part C (applications and reviews)*, 35(4):476–487, 2005.

[31] Jamie Shotton, Toby Sharp, Pushmeet Kohli, Sebastian Nowozin, John Winn, and Antonio Criminisi. Decision jungles: Compact and rich models for classification. In *Advances in neural information processing systems*, pages 234–242, 2013.

[32] Albrecht Zimmermann. Ensemble-trees: Leveraging ensemble power inside decision trees. In *International conference on discovery science*, pages 76–87. Springer, 2008.

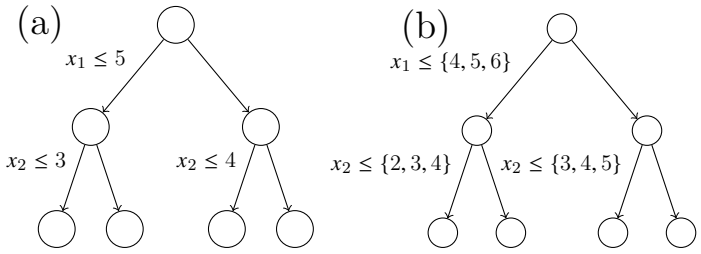


Figure 5: Example for the paths to leafnodes

# Supplement A - Details of Cultivated Random Forest

A leaf node can be written as a product of decision rules by $\mathcal{L}(\mathbf{x}, \mathcal{D}_\mathcal{L}) = \prod_{d \in \mathcal{D}_\mathcal{L}} d(\mathbf{x})$, where $\mathcal{D}_\mathcal{L} = \{d_1, ..., d_l\}$ is the path of $l$ decisions, that have to be fulfilled. For the 4 leafnodes shown in Fig. 1 (a) the corresponding paths $\mathcal{D}_\mathcal{L}$ are:

1. $\{d(\mathbf{x}, 5, 1), d(\mathbf{x}, 3, 2)\}$
2. $\{d(\mathbf{x}, 5, 1), 1 - d(\mathbf{x}, 3, 2)\}$. Note $1 - d(\mathbf{x}, 3, 2)$ corresponds to the decision $I(x_2 > 3)$.
3. $\{1 - d(\mathbf{x}, 5, 1), d(\mathbf{x}, 4, 2)\}$
4. $\{1 - d(\mathbf{x}, 5, 1), 1 - d(\mathbf{x}, 4, 2)\}$

If instead ensemble modules are used as shown in Fig. 1 (b) the corresponding paths $\mathcal{M}_\mathcal{L}$ are:

1. $\{\psi(x, \mathcal{M}_1), \psi(\mathbf{x}, \mathcal{M}_2)\}$
2. $\{\psi(x, \mathcal{M}_1), 1 - \psi(\mathbf{x}, \mathcal{M}_2)\}$
3. $\{1 - \psi(x, \mathcal{M}_1), \psi(\mathbf{x}, \mathcal{M}_3)\}$
4. $\{1 - \psi(x, \mathcal{M}_1), 1 - \psi(\mathbf{x}, \mathcal{M}_3)\}$

with $\mathcal{M}_1 = \{d(\mathbf{x}, 4, 1), d(\mathbf{x}, 5, 1), d(\mathbf{x}, 6, 1)\}$, $\mathcal{M}_2 = \{d(\mathbf{x}, 2, 2), d(\mathbf{x}, 3, 2), d(\mathbf{x}, 4, 2)\}$ and $\mathcal{M}_3 = \{d(\mathbf{x}, 3, 2), d(\mathbf{x}, 4, 2), d(\mathbf{x}, 5, 2)\}$. Then we can write the fraction present in a given

leafnode as:

$$
\begin{aligned}
\mathcal{L}(\mathbf{x}, \mathcal{M}_{\mathcal{L}}) &= \prod_{\mathcal{M} \in \mathcal{M}_{\mathcal{L}}} \left( \frac{1}{|\mathcal{M}|} \sum_{d \in \mathcal{M}} d(\mathbf{x}) \right) \\
&= \frac{1}{\prod_{\mathcal{M} \in \mathcal{D}_{\mathcal{L}}} |\mathcal{M}|} \left( \sum_{d_1 \in \mathcal{M}_1} d_1(\mathbf{x}) \cdot \sum_{d_2 \in \mathcal{M}_2} d_2(\mathbf{x}) \cdot \ldots \cdot \sum_{d_l \in \mathcal{M}_l} d_l(\mathbf{x}) \right) \\
&= \frac{1}{|D_\times|} \sum_{d_1 \in \mathcal{M}_1, \ldots, d_l \in \mathcal{M}_l} d_1(\mathbf{x}) \cdot \ldots \cdot d_l(\mathbf{x}) \\
&= \frac{1}{|D_\times|} \sum_{D \in D_\times} \left( \prod_{d \in D} d(\mathbf{x}) \right) \\
&= \frac{1}{|D_\times|} \sum_{D_{\mathcal{L}} \in D_\times} \mathcal{L}(\mathbf{x}, D_{\mathcal{L}})
\end{aligned}
$$

with $\mathcal{D}_\times = \{\mathcal{M}_1 \times \ldots \times \mathcal{M}_l\}$.

# Supplement B - Simulation

The data used for the illustrative example in Figure 3 of the main paper was generated as following: for $i = 1, \cdots, 300$:

$y_i \sim \mathcal{B}(0.5)$

$x_i | y_i = 1 \sim \mathcal{N}_2\left( \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} 0.75^2 & 0 \\ 0 & 0.75^2 \end{pmatrix} \right)$

$x_i | y_i = 0 \sim \mathcal{N}_2\left( \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 0.75^2 & 0 \\ 0 & 0.75^2 \end{pmatrix} \right)$

The simulation is used to highlight the smoothness of the decision boundaries of different classifiers, when the true underlying relationship requires smooth boundaries.

# Supplement C - Benchmark Experiments

## Data

The UCI data sets were selected under following criteria:

- Public availability.
- Binary classification.
- Mostly numeric or categorical features with low cardinality.

The last criteria has the following reasoning. One of the main improvements of CRF compared to CART is the robust split module, that requires numerical attributes. Hence categorical attributes are transformed to dummy variables, while random forests and CART have different ways built in to handle categorical attributes with larger cardinality. However the CRF framework can straightforwardly be extended to more naturally handling

of categorical features (for example using the approach from CART), which will be one of the next steps in further research. So to achieve a good comparison of the main improvements of CRF we focus on datasets with mostly numeric covariates.
For the genetic datasets fully preprocessed datasets were chosen, so that no further preprocessing needed to be applied.

| Dataset | Covariates | N | Class 1 | Class 2 |
|---|---|---|---|---|
| Australian | 14 | 690 | 383 | 307 |
| Banknote | 4 | 1372 | 762 | 610 |
| Blood Transfusion | 4 | 748 | 570 | 178 |
| Climate Model | 20 | 540 | 46 | 494 |
| Diabetes | 8 | 768 | 500 | 268 |
| EEG-Eye-State | 14 | 14980 | 8257 | 6732 |
| Haberman | 3 | 306 | 225 | 81 |
| Indian Liver | 10 | 583 | 416 | 167 |
| Ionosphere | 34 | 351 | 126 | 225 |
| Magic | 10 | 19020 | 12332 | 6688 |
| Parkinsons | 22 | 195 | 48 | 147 |
| QSAR Biodeg | 42 | 1055 | 699 | 356 |
| Spambase | 57 | 4601 | 2788 | 1813 |
| SPECTF | 44 | 267 | 55 | 212 |
| Steel Plates | 33 | 1941 | 1268 | 673 |
| Vertebral | 6 | 620 | 420 | 200 |
| Wilt | 5 | 4839 | 4578 | 261 |
| Wisconsin Breast Cancer | 30 | 569 | 357 | 212 |

Table 2: Characteristics of the 18 binary classification benchmark data sets from the UCI.

| Dataset | Covariates | N | Class 1 | Class 2 |
|---|---|---|---|---|
| Colon | 2000 | 62 | 40 | 22 |
| DLBC | 2647 | 77 | 19 | 58 |
| Leukaemia | 7128 | 72 | 47 | 25 |
| Prostate | 2135 | 102 | 50 | 52 |

Table 3: Characteristics of the 4 gene expression data sets.

Table 2 shows characteristics of the UCI datasets and Table 3 for the genetic datasets. The datasets cover a wide range of domains as well as varying size and number of covariates. Also different situations of class imbalance are covered. The folder of data sets used is attached. The target variable can be found in the last column of each data set.

## Implementations

We used R (v. 3.5.3) to perform the benchmarking. To this end 10-fold crossvalidation was performed, using the same seed for data splitting for each method.

**Random Forest** Random forest was fit using the ***randomForest*** (v. 4.6-14) package from CRAN. We ran the algorithm with standard settings, which corresponds to 500 trees that are grown until purity and $p/3$ covariates, where $p$ is the number of covariates in total, tried at each node.

**CART** For fitting the CART algorithm, we used the ***rpart*** (v. 4.1-15) package from CRAN. The standard settings are a minimum number of observations for splitting of 20 and a maximum depth of 30. To prevent overfitting a pre-pruning mechanism is build in: a split is not made if the impurity meassure is reduced by less than 0.01. We also tried cost-complexity pruning implemented in rpart, however the results got worse on average, compared to prepruning.

**Cultivated Random Forest** We implemented a prototype version in R. All code to run the experiments together with a full implementation of CRF will be published online at the time of publication, as described above.