JOURNAL OF CHEMICAL INFORMATION AND MODELING

Article

pubs.acs.org/jcim

Harvesting Classification Trees for Drug Discovery

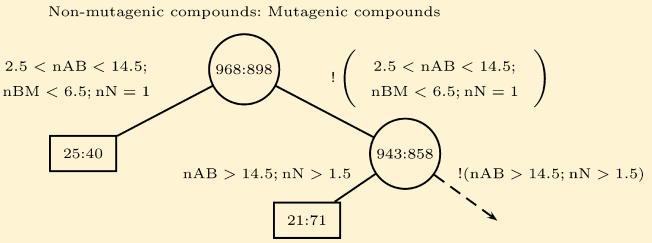
Yan Yuan,*[,†] Hugh A. Chipman,[‡] and William J. Welch[§]

[†]Department of Public Health Sciences, University of Alberta, Edmonton, Alberta T6G 1C9, Canada

[‡]Department of Mathematics and Statistics, Acadia University, Wolfville, Nova Scotia B4P 2R6, Canada

[§]Department of Statistics, University of British Columbia, Vancouver, BC V6T 1Z4, Canada

ABSTRACT: Millions of compounds are available as potential drug candidates. High throughput screening (HTS) is widely used in drug discovery to assay compounds for a particular biological activity. A common approach is to build a classification model using a smaller sample of assay data to predict the activity of unscreened compounds and hence select further compounds for assay. This improves the efficiency of the search by increasing the proportion of hits found among the assayed compounds. In many assays, the biological activity is dichotomized into a binary indicator variable; the explanatory variables are chemical descriptors capturing compound structure. A tree model is interpretable, which is key, since it is of interest to identify diverse chemical classes among the active compounds to serve as leads for drug optimization. Interpretability of a tree is often reduced, however, by the sheer size of the tree model and the number of variables and rules of the terminal nodes. We develop a "tree harvesting" algorithm to filter out redundant "junk" rules from the tree while retaining its predictive accuracy. This simplification can facilitate the process of uncovering key relations between molecular structure and activity and may clarify rules defining multiple activity mechanisms. Using data from the National Cancer Institute, we illustrate that many of the rules used to build a classification tree may be redundant. Unlike tree pruning, tree harvesting allows variables with junk rules to be removed near the top of the tree. The reduction in complexity of the terminal nodes improves the interpretability of the model. The algorithm also aims to reorganize the tree nodes associated with the interesting "active" class into larger, more coherent groups, thus facilitating identification of the mechanisms for activity.



1. INTRODUCTION

High throughput screening (HTS) facilitates the assay of millions of compounds against a biochemical target of interest. It is widely used by the pharmaceutical industry to identify molecules that can be modified to produce new drugs, many of which entered clinical trials in recent years.[1−3]

Exhaustive screening of compounds via HTS is very inefficient, however. First, large chemical collections number in the millions, and the number of theoretically possible drug-like compounds is many orders of magnitude greater. Second, active compounds are very rare. The hit rate, or proportion of active compounds among those screened, is often of the order 1% for a biochemical target, making the screening process extremely inefficient. Third, valuable resources like proteins and chemicals from the inventory are consumed. Fourth, the search is not only for high potencies but also for a variety of chemical structures associated with high potency. Chemists need multiple chemical classes as starting points for modification, because compounds, besides being potent, need to meet constraints on toxicity, side effects, duration of effect, and specificity.[4]

One way to improve the efficiency is to use the sequential screening paradigm.[5,6,3,7] The process starts with screening an initial sample from which a statistical model for the structure−activity relationship is built. On the basis of the first statistical model, the biological activities of unscreened compounds are predicted, in order to get a more focused set of compounds with higher probability of activity for a second round of screening. Several cycles of screening−modeling−screening can be used to improve the hit rate of active compounds.

Among various statistical models that have been investigated for sequential screening, classification/regression trees (also known as recursive partitioning) have enjoyed some successes.[5,8,9] Classification tree algorithms[10,11] are data-adaptive, automatically approximating complex nonlinear relationships, including interaction effects, given enough data. An issue with the tree model is that as the tree grows bigger, the number of variables that define terminal nodes tends to increase, which reduces the interpretability of the tree model.

A major thrust of this article is to remove redundant "junk" rules among those defining the tree. Focusing on the important explanatory variables and their critical ranges aids interpretability, and hopefully, the new rules will be better aligned with the underlying mechanisms affecting the response. As mentioned already, in addition to finding active compounds, HTS aims to identify a diverse set of active compounds with different chemical structures and possibly different activity mechanisms.[12] It is possible that some explanatory variables are important for one mechanism but irrelevant for other mechanisms. Although our proposed "tree harvesting" algorithm is motivated by such complexities in drug discovery data,

Received: January 13, 2012

   dx.doi.org/10.1021/ci3000216 | *J. Chem. Inf. Model.* XXXX, XXX, XXX−XXX

we anticipate that the potential application areas are much wider.

Like tree pruning,[10] the proposed tree harvesting algorithm simplifies a large tree. Unlike pruning, which works from the bottom of the tree up, however, harvesting allows global reorganization of the tree. The tree is simplified by removing junk rules or explanatory variables. Quinlan[13] (Chapter 4) also edited the decision rules of a tree, but the method is quite different. The distinctions are discussed further at the end of section 2, after our algorithm has been introduced. A discussion of other related algorithms is deferred to section 5.

The rest of the article is organized as follows. A simulated example is given in section 2 to describe the tree method and introduce our tree harvesting algorithm. The algorithm is more formally defined in section 3. Section 4 compares trees in terms of their rules before and after harvesting for the simulated example and for two drug-discovery data sets. This section explores a number of important issues, including the ability of harvested trees to manage the trade-off between accuracy and interpretability, insights gained from the use of harvested trees in ensembles, and the interplay between pruning and harvesting. Predictive performance of the harvested trees is also assessed via the latter two data sets. Finally, section 5 concludes with a discussion of the main findings and future work.

2. SIMULATED EXAMPLE

A simple example adapted from a simulation by Lam et al.[14] will be used to briefly review classification trees and outline our harvesting algorithm.

There are three explanatory variables, LogP, MeltPt, and MolWt, as set out in Table 1. Two distinct activity mechanisms are defined by different explanatory variables:

Table 1. Explanatory (Descriptor) Variables for the Simulated Example

variable	description	range
LogP	octanol/water partition coefficient	−2 to 7
MeltPt	melting point	120 to 280 °C
MolWt	molecular weight	200 to 800

A: $3.5 \leq \text{LogP} \leq 4.0$ and

B: $160 \leq \text{MeltPt} \leq 205$ and $400 \leq \text{MolWt} \leq 500$.

Compounds with explanatory variables inside these two regions are active; outside these two regions, there are no active compounds. Assay errors lead to false negatives and false positives in the observed data, however.

Specifically, the training data for 190 compounds are created as follows. First, values are generated at random in the three-dimensional space of explanatory variables in Table 1 such that there are 18 compounds uniformly distributed in region A (mechanism A true actives), 12 compounds uniformly distributed in region B (mechanism B true actives), and 160 compounds uniformly distributed outside the two activity regions (true inactives). These true statuses are shown in Figure 1; because the two active regions are not mutually exclusive, a few compounds belong to both active regions. Second, 3 of the 18 mechanism A compounds and 2 of the 12 mechanism B compounds are incorrectly assayed as negative, and 8 of the 160 inactive compounds are incorrectly assayed as positive. The false negative and false positive rates of 16.7% and 5%, respectively, reflect studies[15,16] where the false negative rate was estimated to exceed the false positive rate.

As well as assay errors, in practice the various activity mechanisms are often unknown and therefore not distinguished in the data; there are just inactive and active compounds. The observed data set to be analyzed looks like Figure 2, where the patterns are less obvious. Though the example is simple it has features which resemble those in actual drug discovery problems: several underlying mechanisms, nonlinear effects, thresholds, imbalance of classes, and noise. In particular, the set of explanatory variables that is useful versus irrelevant depends on the (unknown) mechanism. Uncovering the two active regions in Figure 1 will not be straightforward for widely used classification methods.

Tree models[17] are promising for drug discovery data with junk variables and threshold effects[9] but have limited effectiveness here. A tree model for the simulated example is shown in Figure 3. It is produced by the rpart function[18] in R,[19] which only implements binary splits. The circle at the top of the tree is the root node, containing all 190 observations, of which 157 are class 0 (inactive) and 33 are class 1 (active). Because inactives are in the majority, the node is deemed to be class 0. The root node is split into two descendant nodes using the rule $\text{LogP} < 3.348$ versus $\text{LogP} \geq 3.348$. This rule is chosen from all possible explanatory variables and all possible split boundaries to give the most homogeneous class distributions within each descendant. Homogeneity is defined as the change in the log likelihood based on a probability model of independent Bernoulli random variables. Twenty-two of the 33 class 1 compounds go to the right descendant. Ideally, the two descendant nodes would both be pure—all the inactive-class data going to one node, and all the active-class data going to the other. One split is insufficient, however, and the algorithm iterates, splitting each descendant node. Circles denote internal nodes, which are split further by this process. The rectangles represent terminal nodes, where splitting stops.

We now consider how the 10 terminal nodes relate to the two activity mechanisms. Three of the terminal nodes are classified as class 1 or active. The only active terminal node on the left-hand side of the tree is labeled node 1 in Figure 3. Node 1 is important because it turns out to be the first node processed by the harvesting algorithm. It is defined by the rule set:

$$\text{LogP} < 3.348 \quad (1)$$

$$164.9 \leq \text{MeltPt} < 204.8 \quad (2)$$

$$381.9 \leq \text{MolWt} < 515.3. \quad (3)$$

It contains seven observed actives that are a subset of the true active compounds from mechanism B; the two observed inactives are false negatives, also from the mechanism B region. The other two active terminal nodes are labeled node 2 and node 3 on the right-hand side of the tree in Figure 3. Node 2 has three true active compounds from the mechanism B region mixed with two true inactives, while node 3 has 15 true actives and one false negative, all from the mechanism A region. Between the seven inactive terminal nodes, there are eight observed active compounds, all of which are false positives, and two false negatives from the mechanism A region.

Thus, the tree model does not identify the mechanism B region well: actives from mechanism B are divided between nodes 1 and 2. The first split of the root node is based on LogP with a cutoff at 3.348 (Figure 3), which corresponds roughly to

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

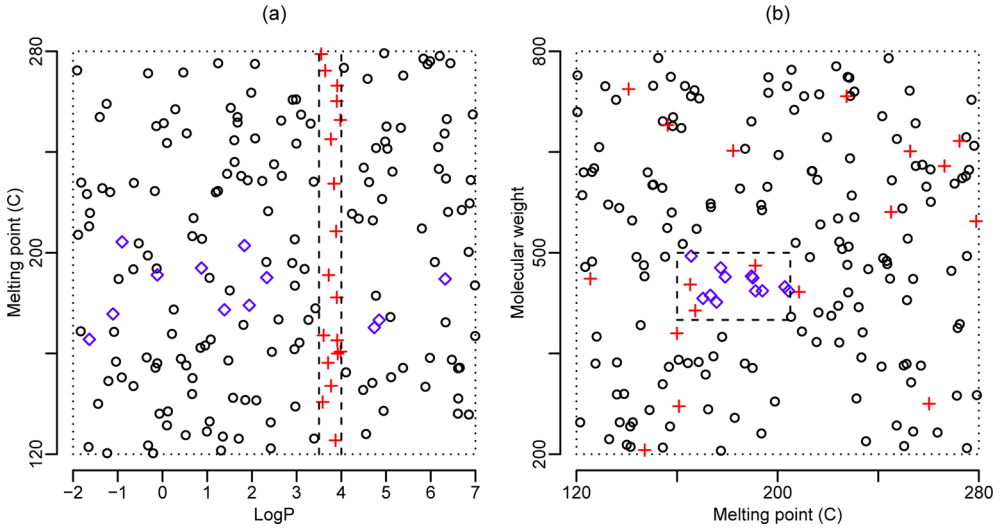


Figure 1. True inactive/active statuses of 190 compounds in the simulated example. Inactive compounds are shown by circles in the explanatory variable space; compounds active via mechanisms A or B are shown by "+" or "◊", respectively. In panels a and b, the dashed lines mark the active regions for LogP (mechanism A) and for MeltPt and MolWt (mechanism B), respectively.

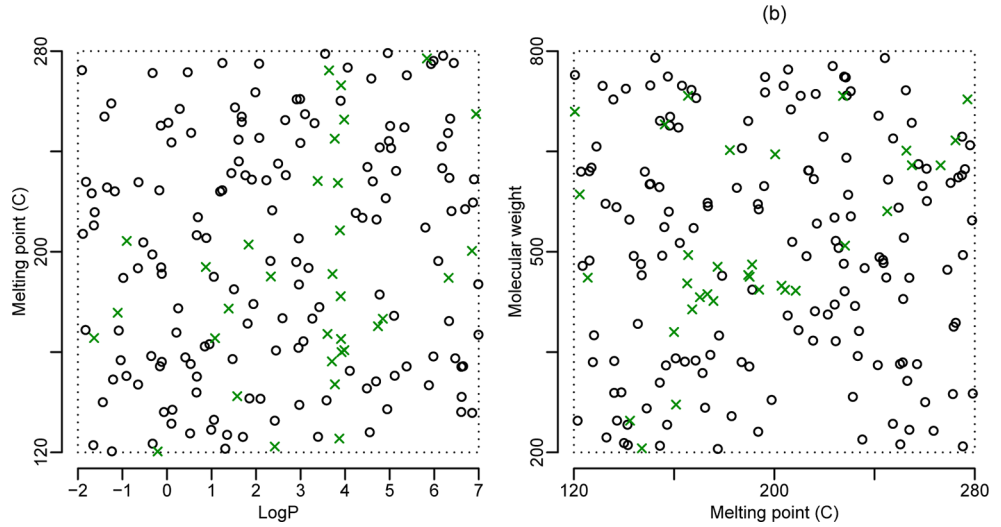


Figure 2. Observed assay results (training data) for the simulated example. Compounds assayed as inactive are shown by circles in the explanatory variable space. The two mechanisms for activity are not distinguished; all compounds assayed as active are shown by "×".

the left boundary for mechanism A. While this succeeds in keeping the active compounds for mechanism A together in the right descendant, the actives from mechanism B are separated between the root node's two descendants. This root node splitting rule is passed down the tree to all terminal nodes. Thus, it is impossible for the actives from mechanism B to be grouped together well. Similarly, if either MeltPt or MolWt had been used to split the root node, the mechanism A actives would have been divided.

The tree harvesting method we propose remedies this problem in the following novel way. For every terminal node, we consider a "simplification" in which junk rules corresponding to one or more variables are discarded. For instance, consider node 1 in Figure 3. If rule 1 is removed from the rule set, leaving only rules 2 and 3, a new, larger set of points in the explanatory variable space is defined. These points become node 1* at the top of Figure 4. The three terminal nodes denoted by dashed rectangles in Figure 4 have fewer data points than before (see Figure 3 for a comparison). In addition to all the compounds in node 1, node 1* has six further active compounds from two terminal nodes on the right side of the tree. This appears to be an improvement. First, the rule set defining node 1* corresponds closely to the MeltPt and MolWt ranges defining mechanism B, and LogP correctly does not appear. Second, the three mechanism B active compounds from the original node 2 are now grouped with those from node 1. Finally, no inactive compounds move to node 1*. The original node 3, which corresponds to mechanism A, also loses three actives to node 1*; these compounds are the three that are seen in Figure 1 to fall in both active regions.

Thus, active compounds, which are of interest here, are gathered together in node 1*, hence the term "harvesting". The name "tree harvesting" has been used previously by Hastie et al.,[20] but in a rather different context. Their method forms a hierarchical *clustering* tree, and harvesting is a postprocessing stepwise algorithm to select cluster averages when building a model for the response of interest. In contrast, our approach

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

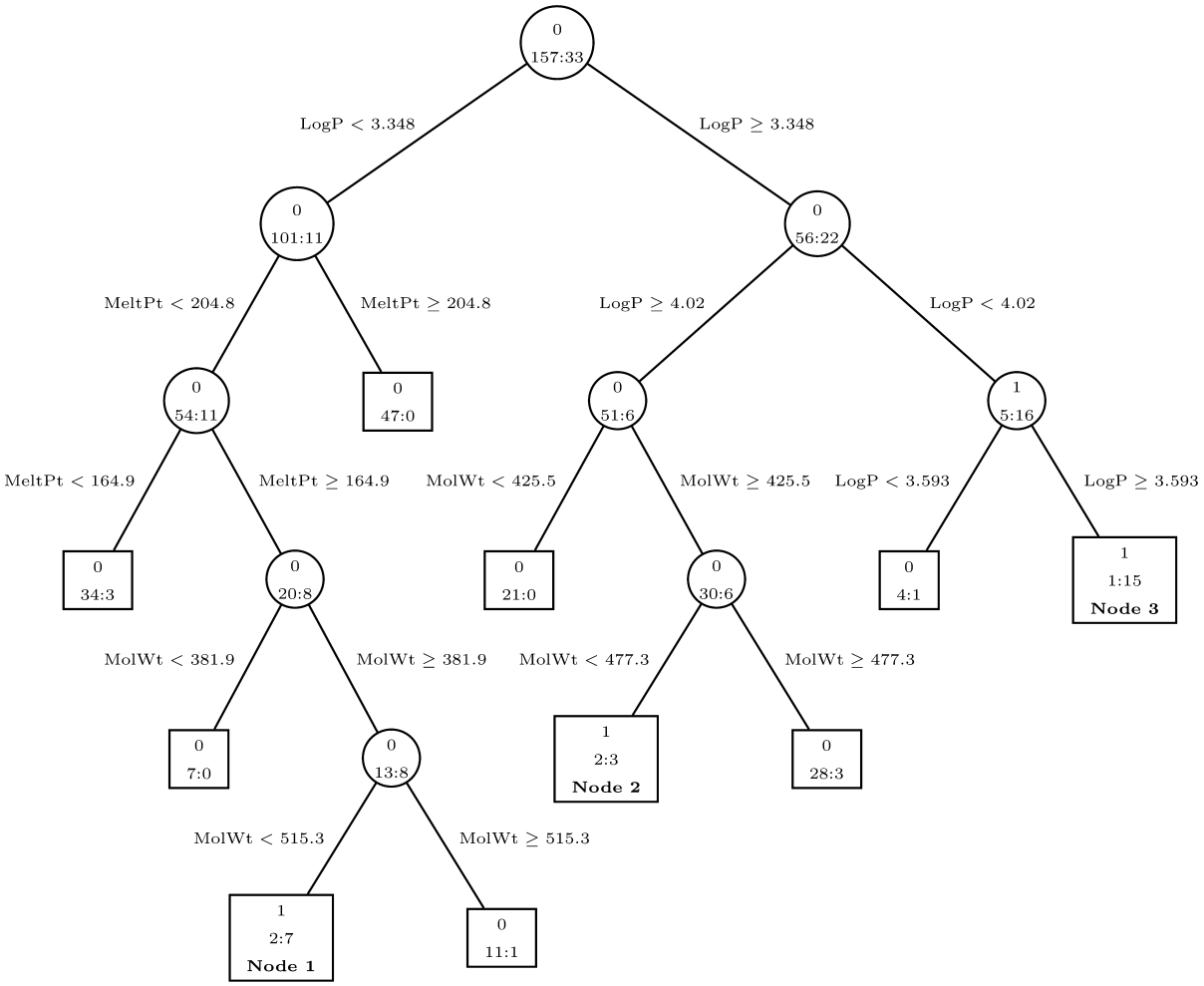


Figure 3. Tree for the simulated example. Nodes that are split (internal nodes), are represented by circles; terminal nodes are represented by rectangles. The predicted class for each node is indicated by a 0 or 1 at the top of the node label; below are the numbers of class 0 (inactive) and class 1 (active) data points. Of the terminal nodes, the three labeled "Node 1", "Node 2", and "Node 3" are predicted to be class 1.

applies to supervised classification trees, and it refines the rules defining the tree architecture.

Alternatively, we could remove one of the other rules in node 1 or a rule from any other terminal node. To assess which choice is best, we use the log likelihood criterion. For example, when rule 1 is removed from node 1, three terminal nodes change. On the basis of the same Bernoulli probability model used to generate the original tree, the change in log likelihood from Figure 3 to Figure 4 is

$$\begin{aligned}\Delta &= l(2, 13) + l(0, 0) + l(2, 0) + l(1, 12) \\ &\quad - [l(2, 7) + l(2, 3) + l(1, 15)] \\ &= -9.42 - (-11.87) = 2.45\end{aligned}$$

where

$$l(a, b) = a \ln(a/(a + b)) + b \ln(b/(a + b)) \quad (4)$$

and a is the number of inactives, b is the number of actives, and $l(0,0) = 0$. Typically in a statistical comparison of models, a small increase in lack of fit is tolerated to simplify a model. In contrast, here the positive value ($\Delta = 2.45$) indicates that this simplification (deletion of a rule) actually yields an improvement in the overall fit of the tree. After removing this rule from node 1, we consider whether either rule 2 or rule 3 in the rule set can be removed. The changes in log likelihood are $\Delta = -31.78$ and -47.42, respectively. Both changes fall below the threshold $\Delta_{crit} = -{}^1/_2\chi^2_{2,0.95} = -2.99$, which indicates an inferior model. Thus, the simplification process for node 1 terminates without deleting either of the remaining two rules. Two degrees of freedom (df) are used for the cutoff since each rule in question has two constraints. The same calculation was performed for every terminal node in the original tree. It turned out that simplifying node 1 by removing only rule 1 gives the largest value of Δ.

Harvesting can be applied to the tree iteratively until none of the nodes remaining in the original tree can be simplified. We will examine the resulting final harvested tree for the simulated example in section 4.1.

In the C4.5 algorithm, Quinlan[13] also considered simplifying rules for terminal nodes. His method and our method are different in several respects. First, his definition of a rule is any condition that leads to an internal or terminal node. For example, there would be five rules for node 1 in Figure 3. In our definition, there is at most one rule for a variable (possibly with both lower and upper bounds). Second, and more importantly, Quinlan's decision to delete a rule was based on a criterion similar to the misclassification rate for a single node. In contrast, harvesting looks at the global change in log likelihood for the entire model.

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

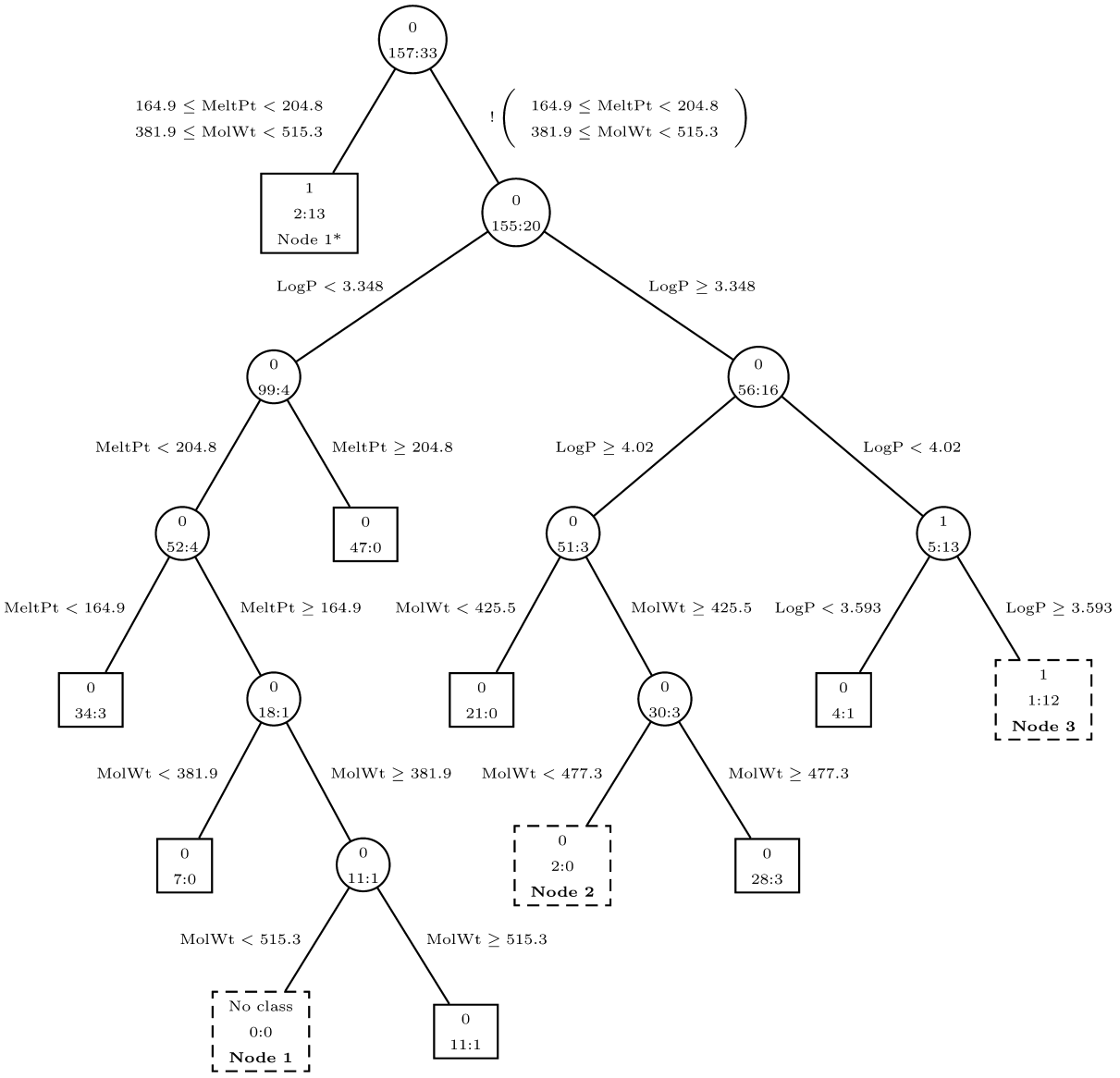


Figure 4. The harvested tree after removing rule 1 from node 1 in Figure 3 to create the terminal node labeled node 1* at the first split. At the first split, "!" denotes the negation of the two rules in parentheses. The three terminal nodes with a dashed border lose data points to node 1*. In particular, node 1 is now empty.

3. HARVESTING ALGORITHM

We now describe the algorithm for the general two-class classification problem with p explanatory variables.

First we define some terms. At any stage in the algorithm, let $\mathcal{N} = \{N_1, ..., N_t\}$ be a subset of terminal nodes from the original tree; we call these unharvested nodes. Let $t = |\mathcal{N}|$ be the number of the unharvested nodes. Similarly, let $\mathcal{H} = \{H_1, ..., H_h\}$ be a set of harvested nodes. The tree harvesting algorithm iteratively removes a node N_i from $\mathcal{N}$ and places a simplified version of it in $\mathcal{H}$ as the new harvested node H_{h+1}.

Node N_i is defined by a set of $k_i = |N_i|$ rules $R_{i1}, ..., R_{ik_i}$, where $k_i \leq p$. A rule refers to both the lower and upper constraints on the variable. For example, suppose there are $p = 3$ explanatory variables; a node could be defined by $k_i = 2$ rules, $x_2 > 2.5$, and $0.1 < x_3 < 14.3$. The distinction between $<$ and $\leq$ is unimportant for tree models, because cutoff points are usually chosen midway between distinct observed values.

A harvested tree is represented by the set of all nodes, $\{\mathcal{H}, \mathcal{N}\}$. The ordering of the nodes is important. When a data point (set of values for the explanatory variables) is passed down the tree, if it satisfies the rules for H_1 then it is assigned to that node. Otherwise, the rules for H_2, H_3, ... are tested, and the point is assigned to the first node where the rules are satisfied. Only if the point does not belong to any of the harvested nodes is it assigned to one of the unharvested nodes.

The log likelihood for a harvested tree,

$$l(\mathcal{H}, \mathcal{N}) = \sum_{j=1}^{h} l(H_j) + \sum_{i=1}^{t} l(N_i)$$

is the sum of contributions over all nodes. The contribution $l(H_j)$ or $l(N_i)$ from any node depends only on the numbers of class 0 and class 1 points, called a and b in (4).

A key step in the algorithm is the creation of a new harvested node H_{h+1} from one of the unharvested nodes N_i, by removing one or more of its rules, $R_{i1}, ..., R_{ik_i}$. As the rules for H_{h+1} are a

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

relaxation of those for N_i, and harvested nodes are considered first in the harvested tree, N_i is depleted of points when harvested, and it no longer makes contribution to the log likelihood and is deleted from $\mathcal{N}$. We note that as a node N_i is harvested, another node N_j remaining in $\mathcal{N}$ may become empty too. This could be due either to specific values of the explanatory variables in the training set or because harvested nodes render some nodes logically empty. Such nodes are retained during the harvesting algorithm, as their simplification could still yield useful harvested nodes.

The harvesting process is divided into two algorithms: `simplify` and `harvest`. The pseudocode for them in Figure 5 uses the set operators ∪ and \ to denote addition or deletion of a node to or from a set of nodes.

```
harvest(𝒩) {
    ℋ ← ∅
    do {
        for i = 1 ... |𝒩| {
            H_i ← simplify(N_i; 𝒩, ℋ, Δ_crit)
            if (H_i ≠ N_i) Δ_i ← l(𝒩 \ N_i, ℋ ∪ H_i) − l(𝒩, ℋ)
            else Δ_i ← −∞
        }
        if (max(Δ_i) ≠ −∞) {
            i* ← argmax_i Δ_i
            𝒩 ← 𝒩 \ N_i*
            ℋ ← ℋ ∪ H_i*
        }
    } while (|𝒩| > 0 and max(Δ_i) ≠ −∞)
    return 𝒩, ℋ
}

simplify(N_i; 𝒩, ℋ, Δ_crit) {
    H_i ← N_i
    do {
        for j = 1 ... |H_i| {
            H_try ← H_i \ R_j
            Δ_j ← l(𝒩 \ N_i, ℋ ∪ H_try) − l(𝒩 \ N_i, ℋ ∪ H_i)
        }
        j* ← argmax_j Δ_j
        if (Δ_j* > Δ_crit) {
            H_i ← H_i \ R_j*
        }
    } while (|H_i| > 0 and Δ_j* > Δ_crit)
    return H_i
}
```

Figure 5. Pseudocode for the harvesting algorithm. Details and notation are provided in section 3.

The `simplify` function operates on a single node, attempting to remove junk rules. This is accomplished by deleting the least important rule, one rule at a time, until all remaining rules are significant. The `harvest` function moves one node at a time from $\mathcal{N}$ to $\mathcal{H}$. It identifies the best node N_{i^*} (with simplified form H_{i^*}) to move. Provided that the rules for the node have actually been simplified ($|H_{i^*}| \neq |N_{i^*}|$), it will remove N_{i^*} from $\mathcal{N}$ and insert the simplified version H_{i^*} in $\mathcal{H}$.

When `simplify` relaxes the constraints on a node N_i, the points belonging to all nodes in $\mathcal{N}$ have to be temporarily updated to enable evaluation of the log-likelihood. Similarly, in the `harvest` step, all points belonging to N_{i^*} are moved to the harvested node H_{i^*}, possibly with points from other remaining nodes in $\mathcal{N}$ (see Figures 3 and 4, for example). Those points will have their node membership updated. Thus, every time a node is harvested from $\mathcal{N}$ to $\mathcal{H}$, the search for the next node to harvest involves trying to simplify rules of each remaining node in $\mathcal{N}$.

In `simplify`, the critical value Δ_{crit} corresponds to $-{}^1/_2\chi^2_{m,0.95}$, half of the 95% upper quantile of a Chi-square distribution with m df, with m = 1 or 2 depending on whether the rule being deleted is 1-sided or 2-sided.

4. RESULTS

In this section, we demonstrate the harvesting algorithm on three data sets and explore various aspects of its performance. Section 4.1 completes the analysis of the simulated example introduced in section 2. In the study of an NCI-AIDS data set (section 4.2), we examine appropriate performance measures for highly unbalanced classes (i.e., rare actives). The example demonstrates that tree harvesting removes many junk rules from the tree, while retaining sufficient structure to give accurate predictions. This ability to retain accuracy while improving interpretability is further demonstrated by placing harvested trees and conventional trees in ensemble models, and noting that the two kinds of ensembles produce similar predictive accuracy. Finally, in this example we demonstrate that harvested trees are more stable than conventional trees. A second application (section 4.3), involving mutagenicity, represents a more equal class distribution. In this example, we delve into a number of different performance measures and study whether harvesting should be applied to a large tree or a pruned tree.

4.1. Simulated Example. Applying the harvesting algorithm to the simulated data in section 2 leads to the harvested tree in Table 2. It has four harvested nodes and four that are unharvested; the latter could not be simplified and retain their rules from the original tree. Two of the original 10 nodes are now empty and not displayed. The choice of harvested node 1, labeled node 1* in Figure 4, was described in section 2. It contains 2 inactive compounds and 13 active compounds in the training data.

Table 2. Harvested Tree for the Simulated Example

	variable range				
harvested node	LogP	MeltPt	MolWt	inactive:active	log likelihood change
1	[−, −]	[164.92, 204.83]	[381.89, 515.29]	2:13	2.46
2	[−, −]	[164.92, 204.83]	[−, 381.89]	11:0	0.25
3	[4.02, −]	[−, −]	[−, −]	48: 3	−1.55
4	[−, 3.35]	[164.92, 204.83]	[−, −]	11:1	0
unharvested node					
A	[3.59, 4.02]	[−, −]	[−, −]	1:12	NA
B	[3.35, 3.59]	[−, −]	[−, −]	3:1	NA
C	[−, 3.35]	[204.83, −]	[−, −]	47:0	NA
D	[−, 3.35]	[−, 165.92]	[−, −]	34:3	NA

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

Harvested node 1 contains all actives from mechanism B and a few actives that fall in both the mechanism A and B regions. The rules 164.92 ≤ MeltPt ≤ 204.83 and 381.89 ≤ MolWt ≤ 515.29 are in good agreement with the actual rules for mechanism B: 160 ≤ MeltPt ≤ 205 and 400 ≤ MolWt ≤ 500. The unharvested node A corresponds to region A and contains 1 inactive and 12 active compounds. The rule 3.59 ≤ LogP ≤ 4.02 is nearly identical to the actual rule for mechanism A: 3.5 ≤ LogP ≤ 4. The harvested tree is therefore largely successful in separating the two mechanisms.

Harvested nodes 2−4 and unharvested nodes B−D are all inactive nodes, each corresponding to part of the inactive region. For example, harvested node 3 contains 48 inactive compounds and three false positives. It corresponds to the inactive area to the right of the LogP active region in Figure 1a.

As well as separating the two mechanisms, the harvested tree has other advantages. First, it contains 8 nodes versus the original tree's 10 nodes. Second, the harvested tree only has 1 or 2 rules per terminal node, whereas the original tree has up to three rules per node on the left side of Figure 3. Third, the harvested tree is more successful in classification overall: The misclassification rates based on the true inactive/active statuses of the compounds are 1.1% and 2.1% for the harvested and original trees, respectively.

4.2. NCI-AIDS Data. This example involves highly unbalanced data from the National Cancer Institute (NCI) on 29 812 compounds assayed for their ability to protect human CEM cells from HIV-1 infection. The database is available from the NCI at http://dtp.nci.nih.gov/docs/aids/aids_data.html; the 1999 release is used in our analysis. The original response comprised three categories: inactive (29 204 compounds), moderately active (393 compounds), and confirmed active (215 compounds). The two active classes were merged to form a single "active" class, somewhat mitigating the class imbalance.

BCUT descriptors[21,12] were calculated by GlaxoSmithKline computational chemists to serve as explanatory variables. The six BCUT numbers we use correspond to the largest and smallest eigenvalues of three Burden connectivity matrices for atomic charge (BCUT 1 and 2), for polarizability (BCUT 3 and 4), and for the ability to form a hydrogen bond (BCUT 5 and 6), respectively.[12] In the construction of the atomic charge matrix B, for example, the diagonal entry B_{ii} is the atomic charge of non-hydrogen atom i in the compound. The off-diagonal entry B_{ij} summarizes the relationship between atoms i and j, such as distance, bond type, and overlap. The other two Burden matrices, for polarizability and the ability to form a hydrogen bond, are constructed in a similar fashion.

4.2.1. Analysis. The rpart package in R was used to build the tree model. Previous work[16] on the same data found that the best out-of-sample predictive performance was achieved by growing very large trees without any pruning. Thus only node size was restricted (at least 10 observations to consider splitting a node and at least 5 observations per terminal node). For completeness, section 4.2.4 also includes the predictive performance of pruned trees on out-of-sample test data.

The original tree has 124 terminal nodes, while the harvested tree has 122. Although harvesting does not dramatically reduce the number of nodes, it does simplify their rules: the original tree has an average of 4.54 rules/node out of a possible maximum of 6 rules/node. The harvested tree has just 2.5 rules/node. Only a small fraction of the 120 or so nodes have high activity levels; they are candidates for closer examination. Smaller rule sets will aid interpretation of these nodes.

As the nodes are harvested, the change in log-likelihood can be monitored. Figure 6 shows that the changes are small,

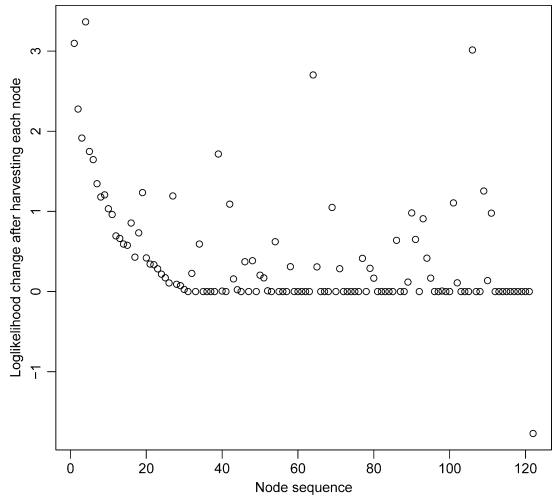


Figure 6. Log likelihood changes for the sequence of harvested nodes.

positive quantities, implying that even though junk rules are being removed, small improvements in fit are gained. The overall log likelihood change is 48.6, indicating purer terminal nodes (activity rate closer to 0 or 1) of the harvested tree relative to before harvesting. A log likelihood change of this magnitude suggests that the harvested tree is substantially improved from the original tree model. A formal test of significance is not carried out, because of difficulty in identification of an appropriate reference distribution.

4.2.2. Measuring Model Performance in Unbalanced Classification. To assess predictive performance, we build the model on training data and examine the performance on separate test data.[22] Four splits of the data into equal-sized training and test sets are used. The splits, originally created by Wang,[16] are made by separate random sampling of active and inactive compounds to ensure a similar balance of active/inactive compounds among data sets. The results on tree size and log-likelihood change described thus far were in fact from one such training data set. Before giving performance results, we briefly discuss appropriate measures of predictive performance in unbalanced classification.

Hit Curve. The hit curve is specifically designed to assess the ability of the model to accurately identify active compounds. We note in passing that misclassification rate is inappropriate for unbalanced problems: a very low misclassification rate can be obtained by a naive classifier that always predicts the majority class.

The hit curve is a graphical measure of performance.[23] In its construction, test compounds are first ranked according to their predicted probability of being active. They are then selected in the ranked order, one at a time, and the cumulative number of actives is recorded. A hit curve displays the number of selected compounds m (m = 1, 2, ..., n) on the horizontal axis against the number of actives, $c_m = \sum_{j=1}^{m} y_{(j)}$, on the vertical axis. Here the 0/1 variable $y_{(j)}$ is the observed response for the jth ranked compound. A hit curve that increases steeply until the actives are exhausted indicates a model effective at prioritizing active compounds, while a line with slope c_n/n corresponds to

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

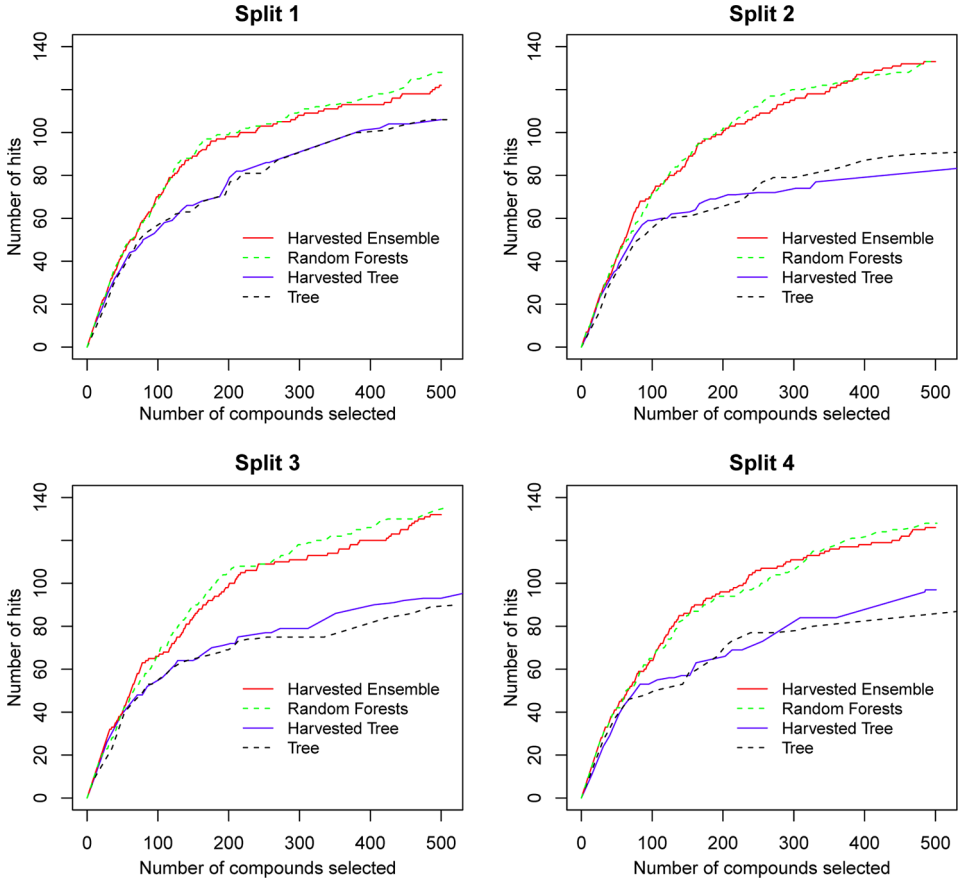


Figure 7. Hit curves up to 500 compounds selected for the test data from four data splits of the NCI-AIDS data. The ranking criteria are p_{lb} for the single tree and its harvested version, $\bar{p}$ for the random forests, and $\bar{p}_H$ for the ensemble of harvested trees.

selecting compounds in random order. Tied predictions are typically handled by selecting groups of tied points rather than one-at-a-time selection.

A hit curve may be summarized numerically by the average hit rate (AHR). Assuming the compounds with tied predictions are in an arbitrary order within the group, the hit rate at the compound with rank m is c_m/m. The AHR averages the hit rate of each *active* compound on the hit curve. For a hit curve up to the M top-ranked compounds, AHR is defined as

$$\mathrm{AHR}_M = \frac{1}{c_M}\sum_{m=1}^{M} y_{(m)}\frac{c_m}{m}$$

It takes value between 0 and 1. A higher AHR corresponds to a steeper rising hit curve and a better ranking model. Typically, we set $M = n$, where n is the size of the test set, and c_M is the total number of actives. Thus, AHR measures the average performance of the entire hit curve. Wang[16] derived a closed form solution for the expected AHR where expectation is taken with respect to permutations of tied predictions in a hit curve.

Ranking the Predictions. It is common to rank the terminal nodes by their estimated probability of being active, $\hat{p}$. For both trees and harvested trees, $\hat{p}$ for each terminal node is the sample proportion of actives in that node. Predictions are then made by passing each new compound down the tree, and assigning it the $\hat{p}$ from the node in which it lands. The main problem with ranking by $\hat{p}$ is that estimation uncertainty is ignored. Consider two terminal nodes, one having 100 compounds with 99 active ($\hat{p}$ = 99/100 = 0.99), the other having only one compound which is active ($\hat{p}$ = 1/1 = 1.00). The $\hat{p}$ criterion will give the second node higher rank, even though its $\hat{p}$ is more variable. Lam et al.[14] encountered this problem in a similar context, and ranked nodes using a lower confidence bound on the probability of activity. Assuming a binomial model for the responses in each terminal node, a node with a inactives and b actives has a 95% lower confidence bound, p_{lb}, which can be calculated as the solution for p in the equation

$$\Pr(B \geq b \mid a, b, p) = 0.05 \qquad (5)$$

where B has a binomial distribution with $a + b$ trials and probability p. With this ranking, p_{lb} is 0.95 for the 1:99 (inactive:active) node and 0.05 for the 0:1 node. We use p_{lb} to produce the hit curves for original and harvested trees.

4.2.3. Random Forest and Ensemble of Harvested Trees. In addition to the single tree model, we compare our method with random forests,[24] an ensemble of trees. A random forest typically consists of many single tree models, say 500, each of which assigns 0 or 1 class membership to a test compound based on the label of the terminal node in which it lands. A prediction of the probability of activity is the proportion of 1's the compound received from the 500 votes, denoted by $\bar{p}$. The compounds are ranked according to $\bar{p}$.

For comparisons with random forests, we use a bootstrap ensemble of 100 harvested trees. A harvested tree is built for each bootstrap sample to give predicted probabilities of activity, $\hat{p}^{(b)}$ (b = 1, ..., 100), for each test compound. Here, $\hat{p}^{(b)}$ could be either from the simple sample proportion of actives in the relevant node or it could be the lower bound in (5). We use the sample proportion for $\hat{p}^{(b)}$ in our study. The final predicted

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

probability, $\bar{p}_H$, for a test compound is the average of $\hat{p}^{(b)}$ across the 100 harvested trees. Compounds are then ranked by $\bar{p}_H$.

We examine random forests and ensembles of harvested trees to assess whether the harvesting process degrades trees in such a way that they are less suitable as base learners in an ensemble. Of course, we expect that ensembles based on trees (e.g., random forests) and on harvested trees will have superior predictive performance in comparison to individual trees or harvested trees. In balance, individual trees and (especially) harvested trees will be more interpretable.

4.2.4. Prediction Performance of Harvested Trees. Figure 7 displays test set hit curves for the top ranking 500 compounds from four models: a single (unharvested) tree, its harvested version, a random forest, and an ensemble of harvested trees. Each method is executed on the four data splits. The ranking criteria are p_{lb} for a single tree and its harvested version, $\bar{p}$ for the random forests, and $\bar{p}_H$ for the ensemble of harvested trees. For all four data splits, the hit curves of the random forest and the ensemble of harvested trees are comparable and both rise much steeper than the hit curves of the single tree models, with or without harvesting. Between the harvested tree and original tree model, the harvested tree performs slightly better. In the top ranking 500 compounds, both ensemble methods find roughly 130 active compounds in each of the four data splits, while the two single tree methods find between 80 and 100 active compounds.

Figure 8 summarizes via AHR the hit curves when all test compounds are selected. The results are consistent with the

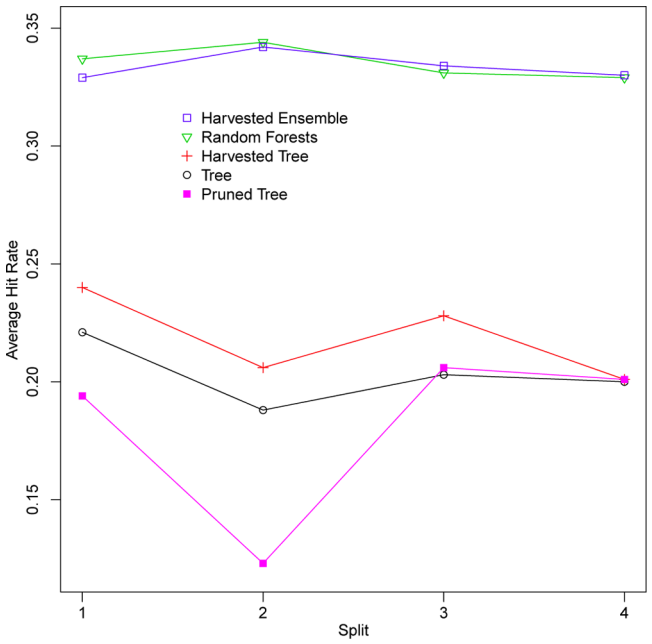


Figure 8. Average hit rate for four sets of test data from four data splits of the NCI-AIDS data. Test compounds are ranked by the p_{lb} criterion for the original single tree, its harvested version, and its pruned version; by $\bar{p}$ for random forests; and by $\bar{p}_H$ for an ensemble of harvested trees.

curves up to 500 compounds in Figure 7. The random forests and the ensembles of harvested trees have similar AHRs in the 0.33−0.34 range across data splits. The AHRs of the single tree and harvested tree are considerably lower than those of the ensemble methods at around 0.2, with the AHRs of the harvested trees slightly better for all splits. For completeness, we also include the AHRs of pruned trees, where the original trees are pruned back using a cross-validated misclassification criterion on the training data. The pruning does not necessarily improve the prediction performance by the hit rate criterion used here, as evidenced by Figure 8 and previous work.[16]

As expected, the ensemble models outperform individual trees. What is interesting about the comparison is that an ensemble of harvested trees is competitive with a random forest of conventional trees. This suggests that harvesting, while simplifying the tree by removing extraneous structure, provides an effective base classifier for an ensemble. The earlier analysis in section 4.2.1 showed that the harvesting process actually increased the likelihood function. Similarly, the comparisons in Figures 7 and 8 point to harvested trees providing stronger base classifiers than the original trees. Breiman[25] argued that increasing the strength of the base classifiers is one factor leading to improved performance of an ensemble of bagged predictors. On the other hand, Breiman also showed that an ensemble produces more gain in performance for unstable classifiers, and we argue in section 4.2.5 that harvested trees are more stable than the original trees. There are complex trade-offs in constructing an ensemble and the use of harvested trees as the base classifiers provides no gain or loss in predictive performance here relative to random forests.

Thus, we are not recommending harvested trees for an ensemble. While they may be competitive with random forests for prediction, they are much slower to construct. The advantage of harvested trees is their interpretability.

4.2.5. Stability of Harvested Trees. One criticism of tree models is their instability: small changes in the data can result in drastically different trees.[26] Here we show that the harvested trees are less sensitive than the original trees to disturbances of the training data across the four data splits.

Stability is measured here in terms of the rules of the active nodes, because identification of different activity mechanisms is of interest. For any data split, the rule sets defining the top 10 active nodes (ranked by p_{lb}) of the harvested tree (similarly original tree) are considered. Any variable could appear from 0 to 10 times in these 10 nodes, and we look at the stability of these variable ranges across the 4 splits.

Figure 9 compares the active regions of the original and harvested trees for two of the BCUT variables, x_2 and x_4. The gray portion of each line segment is the apparent rule, which is simply the rule given by the algorithm. The black portion is the effective range, which is the actual data range for training compounds belonging to that node. For x_2, it is seen that rules are much more stable across the four splits for the harvested trees (bottom panel), compared with the original trees (top panel). Two active regions with sharper boundaries are apparent in the harvested trees. In contrast, the active regions (the gray segments) identified by the original trees cover the entire range in each split. The plots for x_4 on the right convey a similar message. While the apparent rules from the original trees consist of a mix of tight ranges and wide ranges on x_4, the harvested trees keep most of the tight ranges, which are more meaningful, and remove many of the wide ranges.

Similar plots (not shown) for the other explanatory variables also show more stability for the rules of the harvested nodes.

Across data splits, any pair of training data sets have only 50% of the data in common on average, thus the perturbation of the training data is large. Nonetheless, patterns emerge here for the harvested active nodes.

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

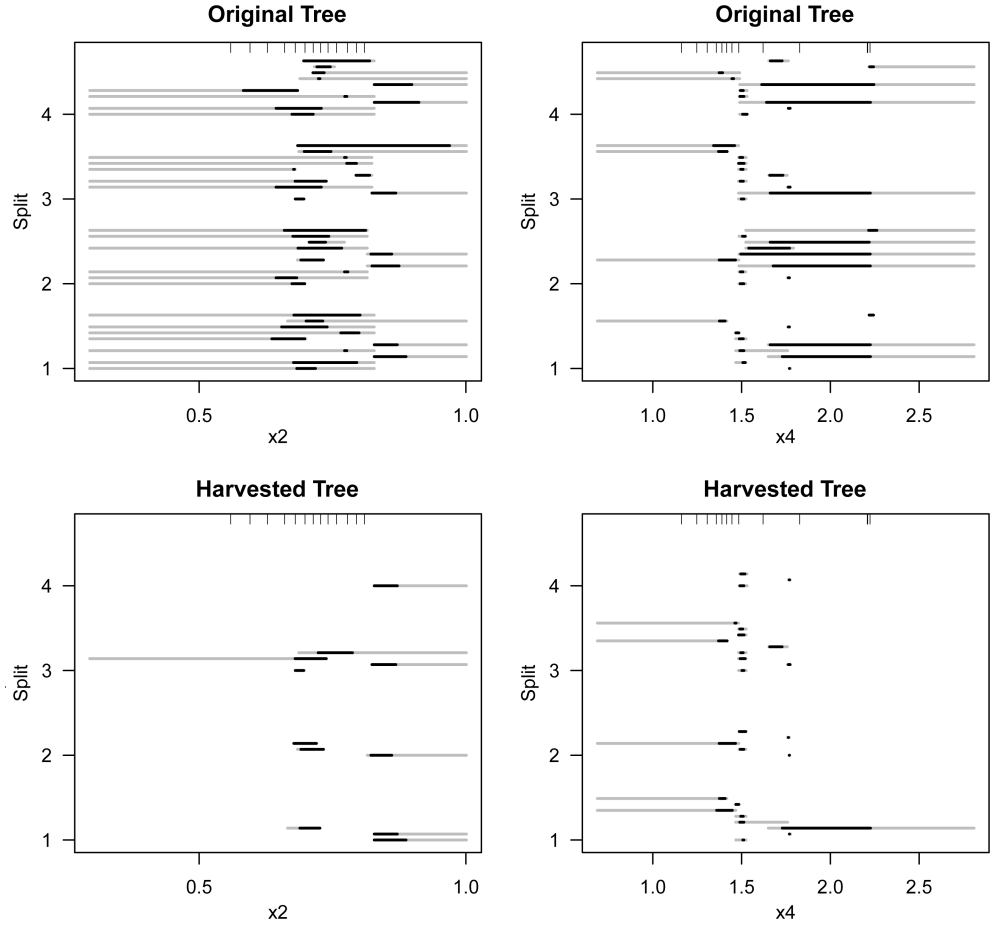


Figure 9. Active x_2 and x_4 regions identified by the 10 top active nodes of the original and harvested trees across four splits of the NCI-AIDS data. Each gray segment represents a rule that defines one node. The black segment reflects the "effective" range, the actual range of the training data in the node. Not all nodes have rules on each variable, especially for the harvested nodes. In each split, there are between 1 and 9 segments for the harvested trees (median 3.5) and between 9 and 10 segments for the original trees (median 10). At the top of the plot, short vertical lines indicate the 2.5, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 95, and 97.5 percentiles of each variable.

Table 3. Results of Classification Tree, Pruning, and Harvesting for Four Different Train/Test Splits of the Mutagenicity Data Set, where 75% and 25% of the Data Was Used for Training and Testing, Respectively[a]

		train/test				
		abc/d	*acd/b*	*abd/c*	*bcd/a*	average
misclassification rate	original	0.246	0.240	0.277	0.279	0.261
	harvested	0.241	0.234	0.270	0.288	0.258
	pruned	0.212	0.240	0.262	0.289	0.251
	harv pruned	0.220	0.238	0.255	0.275	0.247
number of nodes	original	71	67	65	67	67
	harvested	71	66	64	63	66
	pruned	25	22	25	19	23
	harv pruned	24	21	24	19	22
number of variables used	original	29	21	26	29	26
	harvested	29	21	26	29	26
	pruned	19	17	15	14	16
	harv pruned	18	16	14	13	15
number of rules per node	original	8.68	7.58	7.11	7.91	7.82
	harvested	2.99	2.90	2.78	3.10	2.94
	pruned	6.38	5.27	5.28	5.42	5.59
	harv pruned	2.08	2.38	2.75	2.16	2.34

[a]"Harv pruned" indicates that the harvesting algorithm is applied to the pruned tree.

4.3. Mutagenicity Data. Here the 0/1 binary response is nonmutagenic/mutagenic. Since mutagenic compounds tend to increase the frequency or extent of DNA mutation, it is cost-effective to identify mutagens in the early phase of drug

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

development process. We analyze a data set of 1866 compounds (968 nonmutagenic, 898 mutagenic) obtained from GlaxoSmithKline. The constitutional descriptor set used for the explanatory variables is generated using Dragon software.[27] It consists of 47 variables such as molecular weight, volume, number of oxygen atoms, etc.

The class balance of the mutagenicity data leads to some different techniques for model-building and performance assessment. The rpart tree was pruned using cross-validated misclassification rate, and the harvesting algorithm was applied to the pruned tree. Due to the balanced nature of the data, predictive performance is measured by the misclassification rate.

4.3.1. Performance of Harvested Tree. Table 3 shows some characteristics of the pruned tree and the harvested pruned tree, as well as those of the original tree and the harvested original tree. The data were randomly partitioned into four subsets (labeled a–d); three form the training data (75%) and the other is the test data (25%). Results for all four possible training/test splits are reported. Within the training set, 3-fold cross-validation (each time leaving out 25% of the full data) was used to select tree size using pruning. In all instances, the same four subsets are used as building blocks for the train/test splits and for cross-validation within the training set.

In all cases, the harvesting algorithm results in (i) comparable test set misclassification rates, (ii) the same number of nodes or slightly fewer, (iii) use of the same number of variables or one fewer, and (iv) removal of over half the rules as junk rules, when compared with the tree model it is applied to. These results are similar to the NCI-AIDS example, with harvesting yielding fewer, more interpretable rules, and no deterioration in out-of-sample predictions.

The pruned tree constructed using all the data has 22 terminal nodes and uses 15 of the variables. Table 4 shows the first nine harvested nodes and lists the variables with junk rules removed during harvesting. The table indicates considerable simplification of the original tree.

Table 4. First Nine Harvested Nodes for the Mutagenicity Data

node	mutagenic/ total	label	rules	deleted variables
1	40/65	1	2.5 < nAB < 14.5; nBM < 6.5; nN = 1	AMW nR03
2	71/92	1	nAB > 14.5; nN > 1.5	nCIR
3	53/64	1	nAB < 14.5; nR03 > 0.5	AMW nN
4	15/126	0	nR06 > 0.5; nAT < 15.5	nAB nN nO Sp
5	20/60	0	2.5 < nAB < 14.5; Mv < 0.635; nN = 1	AMW nBM nR03
6	96/125	1	nAB < 14.5; AMW > 6.245; 0.5 < nX < 3.5	nN nR03
7	9/47	0	nBnz = 0; nO = 0; nN > 1.5	nAB Sp
8	93/132	1	nN > 1.5; nBnz >0.5; Sp < 20.33	nR03 nX
9	35/131	0	nAB < 14.5; nN < 1.5; AMW > 6.245; MW < 221.3	nR03 nX

The node harvested first was constrained by five conditions originally:

$$2.5 < \text{nAB (number of aromatic bonds)} < 14.5 \quad (6)$$

$$\text{nN (number of Nitrogen atoms)} = 1 \quad (7)$$

$$\text{nBM (number of multiple bonds)} < 6.5 \quad (8)$$

$$\text{nR03 (number of 3-membered rings)} = 0 \quad (9)$$

$$\text{AMW (average molecular weight)} < 6.245. \quad (10)$$

It contains 49 compounds, and 32 of them are mutagenic. When harvested, conditions 9 and 10 on nR03 and AMW were dropped. The resulting harvested node contains 65 compounds, of which 40 are mutagenic. Since there are three rules for this harvested node, the interpretation becomes simpler. Similar complexity reduction occurs in the second harvested node, while increasing the size of the node from 45 to 92 compounds and reducing the node's misclassification rate from 27% to 23%.

Similar to the NCI-AIDS analysis, the harvesting process yields a sequence of mostly small increases to the log-likelihood. The improvement is largest for the first six nodes, then near zero for the remaining nodes. The overall log likelihood increase is 21.62, indicating a substantial improvement from the original tree model. These results are for harvesting a tree grown from the full data set, although the patterns are similar for the four train/test splits as above.

5. DISCUSSION

The key contributions of the tree harvesting algorithm are (1) it offers an innovative approach to deal with multiple mechanisms and (2) it improves the interpretability of the model by creating simpler rules in the harvested nodes of the reorganized tree. The current algorithm also improves the support of the identified rules by increasing the sizes of nodes. Besides, our algorithm demonstrates modest improvements in prediction performance when the harvested tree is compared to the original tree model. Our focus has been classification trees, but as the methodology is based on the general concept of likelihood it would extend to regression trees for a continuous response variable.

A number of other approaches also aim to simplify rules. For example, Quinlan's C4.5 algorithm[13] has an optional step, c4.5rules, which allows the deletion of rules from the rule sets that identify terminal nodes. Our method is different: We use a likelihood-based criterion for removing rule(s), consider deletion of the entire rule for a variable, and include specialized tools for the analysis of unbalanced classification problems. Our method and the method of Boulesteix and Tutz[28] have similar objectives. However, their hypothesis testing is local to a rule set, whereas we look at the likelihood for the whole tree. Lausen et al.[29] used a *p*-value based method that would avoid irrelevant rules at the bottom of the tree but it does not consider rules at the top of the tree. Friedman and Fisher's PRIM algorithm[30] seeks to identify rules of the same form as our tree harvesting algorithm. A critical difference is that PRIM is a top-down algorithm that constructs the final nodes "from scratch", while our approach modifies an existing tree. One advantage of our approach is that it can be used as a complement to a tree-based analysis, producing rules that are more interpretable than the original tree. Friedman and Popescu[31] also consider the generation of rule ensembles, but allow different rules to additively combine to generate a final prediction.

In comparisons with ensemble methods, we focused on "model averaging" ensembles such as random forests and bootstrap averages of harvested trees. Although we see no advantage in substituting a harvested tree for a conventional

dx.doi.org/10.1021/ci3000216 | J. Chem. Inf. Model. XXXX, XXX, XXX–XXX

tree in such ensembles, it is encouraging that the harvesting algorithm does not degrade predictive performance in this context. We did not study the use of harvested trees as weak learners in boosting type ensembles. The individual trees used in boosting algorithms are much simpler (and individually weaker) models than the harvested trees developed here. The fact that the harvesting algorithm seems to retain and perhaps improve predictive accuracy suggests that harvested trees would be too sophisticated a model to use as a weak learner.

The current harvesting algorithm treats the original rules coarsely, considering the deletion of all rules involving a variable at once. Finer control might be possible, such as adjusting the individual constraints on variables rather than just deleting them. For example, in the simulated example, the true constraint on MeltPt for mechanism B is [160, 205], while the tree identifies the constraint as [164.9, 204.8]. By adjusting (and possibly deleting) the individual bounds in this rule, it may be possible to better uncover the actual structure and perhaps reduce the number of terminal nodes substantially.

The criterion to harvest nodes is based on the log-likelihood, and harvesting continues provided that the change in log-likelihood is larger than a negative cutoff. Lacking an appropriate reference distribution, the current cutoffs are based on the 95th percentiles of a χ^2 distribution with 1 or 2 df, depending on whether 1 or 2 constraints make up the rule being deleted. While a proper reference distribution for this cutoff would be desirable, its identification is complicated by the greedy, sequential nature of the harvesting algorithm.

AUTHOR INFORMATION

Corresponding Author

*E-mail: yyuan@ualberta.ca (Y.Y.).

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

Research supported by MITACS and NSERC, Canada. We are grateful to the reviewers for their insightful comments and suggestions. We also thank Stan Young for providing data and commenting on this research.

REFERENCES

(1) Fox, S.; Farr-Jones, S.; Sopchak, L.; Boggs, A.; Nicely, H. W.; Khoury, R.; Biros, M. High-Throughput Screening: Update on Practices and Success. *J. Biomol. Screening* **2006**, *11*, 864−869.

(2) Macarron, R.; Banks, M. N.; Bojanic, D.; Burns, D. J.; Cirovic, D. A.; Garyantes, T.; Green, D. V. S.; Hertzberg, R. P.; Janzen, W. P.; Paslay, J. W.; Schopfer, U.; Sittampalam, G. S. Impact of High-Throughput Screening in Biomedical Research. *Nat. Rev. Drug Discovery* **2011**, *10*, 188−195.

(3) Mayr, L. M.; Bojanic, D. Novel Trends in High-Throughput Screening. *Curr. Opin. Pharmacol.* **2009**, *9*, 580−588.

(4) Bradley, E. K.; Beroza, P.; Penzotti, J .E.; Grootenhuis, P. D. J.; Spellmeyer, D. C.; Miller, J. L. A Rapid Computational Method for Lead Evolution: Description and Application to α_1-Adrenergic Antagonists. *J. Med. Chem.* **2000**, *43*, 2770−2774.

(5) Abt, M.; Lim, Y.; Sacks, J.; Xie, M.; Young, S. S. A Sequential Approach for Identifying Lead Compounds in Large Chemical Databases. *Stat. Sci.* **2001**, *16*, 154−168.

(6) Engels, M. F. M.; Venkatarangan, P. Smart Screening: Approaches to Efficient HTS. *Curr. Opin. Drug Discovery Dev.* **2001**, *4*, 275−283.

(7) Young, S. S.; Lam, R. L. H.; Welch, W. J. Initial Compound Selection for Sequential Screening. *Curr. Opin. Drug Discovery Dev.* **2002**, *5*, 422−427.

(8) van Rhee, A. M.; Stocker, J.; Printzenhoff, D.; Creech, C.; Wagoner, P. K.; Spear, K. L. Retrospective Analysis of an Experimental High-Throughput Screening Data Set by Recursive Partitioning. *J. Comb. Chem.* **2001**, *3*, 267−277.

(9) Young, S. S.; Hawkins, D. M. Using Recursive Partitioning to Analyze a Large SAR Data Set. *SAR QSAR Environ. Res.* **1998**, *8*, 183−193.

(10) Breiman, L.; Friedman, J. H.; Olshen, R. A.; Stone, C. J. *Classification and Regression Trees*; Wadsworth: Monterey, CA, 1984.

(11) Hawkins, D. M.; Kass, G. V. Automatic Interaction Detection. In *Topics in Applied Multivariate Analysis*; Hawkins, D. M., ed.; Cambridge University Press: Cambridge, UK, 1982.

(12) Pearlman, R. S.; Smith, K. M. Metric Validation and the Receptor-Relevant Subspace Concept. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 28−35.

(13) Quinlan, J. R. *C4.5: Programs for Machine Learning*; Morgan Kaufmann: San Mateo, CA, 1993.

(14) Lam, R. L. H.; Welch, W. J.; Young, S. S. *Cell-Based Analysis of High Throughput Screening Data for Drug Discovery*; Research Report RR-02-02; Business and Industrial Statistics Research Group; University of Waterloo, Canada, 2002.

(15) Coma, I.; Clark, L.; Diez, E.; Harper, G.; Herranz, J.; Hofmann, G.; Lennon, M.; Richmond, N.; Valmaseda, M.; Macarron, R. Process Validation and Screen Reproducibility in High-Throughput Screening. *J. Biomol. Screening* **2009**, *14*, 66−76.

(16) Wang, Y. *Statistical Methods for High Throughput Screening Drug Discovery Data*. PhD thesis, University of Waterloo, Canada, 2005.

(17) Venables, W. N.; Ripley, B. D. *Modern Applied Statistics with S-Plus*, third ed.; Springer: New York, NY, 1999; pp 303−327.

(18) Therneau, T. M.; Atkinson, B. Rpart: Recursive Partitioning. R package version 3.1−53. http://CRAN.R-project.org/package=rpart (accessed June 15, 2011).

(19) R Development Core Team R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2011, http://www.R-project.org (accessed June 15, 2011).

(20) Hastie, T.; Tibshirani, R.; Botstein, D.; Brown, P. Supervised Harvesting of Expression Trees. *Genome Biol.* **2001**, *2*, 1−12.

(21) Burden, F. R. Molecular Identification Number for Substructure Searches. *J. Chem. Inf. Comput. Sci.* **1989**, *29*, 225−227.

(22) Hawkins, D. M. The Problem of Overfitting. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1−12.

(23) Zhu, M.; Su, W.; Chipman, H. A. LAGO: A Computationally Efficient Approach for Statistical Detection. *Technometrics* **2006**, *48*, 193−205.

(24) Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5−32.

(25) Breiman, L. Bagging Predictors. *Mach. Learn.* **1996**, *24*, 123−140.

(26) Breiman, L. Arcing Classifiers (with discussion and a rejoinder by the author). *Ann. Stat.* **1998**, *26*, 801−849.

(27) Todeschini, R.; Consonni, V. *Handbook of Molecular Descriptors*; Wiley-VCH: Weinheim, Germany, 2000.

(28) Boulesteix, A.-L.; Tutz, G. Identification of Interaction Patterns and Classification With Applications to Microarray Data. *Comput. Stat. Data An.* **2006**, *50*, 783−802.

(29) Lausen, B.; Hothorn, T.; Bretz, F.; Schumacher, M. Assessment of Optimal Selected Prognostic Factors. *Biometrical J.* **2004**, *46*, 364−374.

(30) Friedman, J. H.; Fisher, N. I. Bump Hunting in High-Dimensional Data. *Stat. Comp.* **1999**, *9*, 123−143.

(31) Friedman, J. H.; Popescu, B. E. Predictive Learning via Rule Ensembles. *Ann. Appl. Stat.* **2008**, *2*, 916−954.