TREE ENSEMBLES WITH RULE STRUCTURED HORSESHOE REGULARIZATION

BY MALTE NALENZ AND MATTIAS VILLANI

Linköping University

We propose a new Bayesian model for flexible nonlinear regression and classification using tree ensembles. The model is based on the RuleFit approach in Friedman and Popescu [*Ann. Appl. Stat.* **2** (2008) 916–954] where rules from decision trees and linear terms are used in a L1-regularized regression. We modify RuleFit by replacing the L1-regularization by a horseshoe prior, which is well known to give aggressive shrinkage of noise predictors while leaving the important signal essentially untouched. This is especially important when a large number of rules are used as predictors as many of them only contribute noise. Our horseshoe prior has an additional hierarchical layer that applies more shrinkage a priori to rules with a large number of splits, and to rules that are only satisfied by a few observations. The aggressive noise shrinkage of our prior also makes it possible to complement the rules from boosting in RuleFit with an additional set of trees from Random Forest, which brings a desirable diversity to the ensemble. We sample from the posterior distribution using a very efficient and easily implemented Gibbs sampler. The new model is shown to outperform state-of-the-art methods like RuleFit, BART and Random Forest on 16 datasets. The model and its interpretation is demonstrated on the well known Boston housing data, and on gene expression data for cancer classification. The posterior sampling, prediction and graphical tools for interpreting the model results are implemented in a publicly available R package.

1. Introduction. Learning and prediction when the mapping between input and outputs is potentially nonlinear and observed in noise remains a major challenge. Given a set of N training observations $(\mathbf{x}, y)_i, i = 1, \ldots, N$, we are interested in learning or approximating an unknown function f observed in additive Gaussian noise

$$y = f(\mathbf{x}) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2),$$

and to use the model for prediction. A popular approach is to use a learning ensemble [Breiman (1996, 2001), Freund and Schapire (1996), Friedman (2001)]

$$f(\mathbf{x}) = \sum_{l=1}^{m} \alpha_l f_l(\mathbf{x}),$$

where $f_l(\mathbf{x})$ is a basis function (also called a weak learner in the machine learning literature) for a subset of the predictors. A variety of basis functions f_l have been proposed in the last decades, and we will here focus on decision rules. Decision rules are defined by simple if-else statements and therefore highly interpretable by humans. Finding a set of optimal rules is NP hard [Friedman and Popescu (2008)], and most practical algorithms therefore use a greedy learning procedure. Among the most powerful are divide and conquer approaches [Cohen (1995), Fürnkranz (1999)] and boosting [Schapire (1999), Dembczyński, Kotłowski and Słowiński (2010)].

A new way to learn decision rules is introduced in Friedman and Popescu (2008) in their RuleFit approach. RuleFit is estimated by a two-step procedure. The *rule generation* step extracts decision rules from an ensemble of trees trained with gradient boosting. The second *regularizaton* step learns the weights α_l for the generated rules via L1-regularized (Lasso) regression, along with weights on linear terms included in the model. This is similar to stacking [Wolpert (1992), Breiman (1996)], with the important difference that the members of the ensemble are not learned decision trees or other predictors, but individual rules extracted from trees. As argued in Friedman and Popescu (2008), this makes RuleFit a more interpretable model and, we argue below, has important consequences for the regularization part. RuleFit has been successfully applied in particle physics, in medical informatics and in life sciences. Our paper makes the following contributions to improve and enhance RuleFit.

First, we replace the L1-regularization [Tibshirani (1996)] in RuleFit by a generalized horseshoe regularization prior [Carvalho, Polson and Scott (2010)] tailored specifically to covariates from a rule generation step. L1-regularization is computationally attractive, but has the well-known drawback of also shrinking the effect of the important covariates. This is especially problematic here since the number of rules from the rule generation step can be very large while potentially only a small subset is necessary to explain the variation in the response. Another consequence of the overshrinkage effect of the L1-regularization is that it is hard to choose an optimal number of rules; increasing the number of rules affects the shrinkage properties of the Lasso. This makes it very hard to determine the number of rules a priori, and one has to resort to cross-validation, thereby mitigating the computational advantage of the Lasso. A horseshoe prior is especially attractive for rule learning since it shrinks uninformative predictors aggressively while leaving important ones essentially untouched. Inspired by the prior distribution on the tree depth in Bayesian Additive Regression Trees (BART) [Chipman, George and McCulloch (2010)], we design a generalized horseshoe prior that shrinks overly complicated and specific rules more heavily, thereby mitigating problems with overfitting. This is diametrically opposed to RuleFit, and to BART and boosting, which all combine a myriad of rules into a collective where single rules only play a very small part.

Second, we complement the tree ensemble from gradient boosting [Friedman (2001)] in RuleFit with an additional set of trees generated with Random Forest.

The error-correcting nature of boosting makes the rules highly dependent on each other. Trees from Random Forest [Breiman (2001)] are much more random and adding them to rules from boosting therefore brings a beneficial diversity to the tree ensemble. Note that it is usually not straightforward to combine individual trees from different ensemble strategies in a model; our combination of RuleFit and horseshoe regularization is an ideal setting for mixing ensembles since RuleFit makes it easy to combine ensembles, and the horseshoe prior can handle a large number of noise rules without overfitting.

Third, an advantage of our approach compared to many other flexible regression and classification models is that predictions from our model are based on a relatively small set of interpretable decision rules. The possibility to include linear terms also simplifies interpretation since it avoids a common problem with decision trees that linear relationships need to be approximated with a large number of rules. To further aid in the interpretation of the model and its predictions, we also propose graphical tools for analyzing the model output. We also experiment with post-processing methods for additional pruning of rules to simplify the interpretation even further using the method in Hahn and Carvalho (2015).

We call the resulting two-step procedure with mixed rule generation followed by generalized rule structured horseshoe regularization the *HorseRule* model. We show that HorseRule's ability to keep the important rules and aggressively removing unimportant noise rules leads to both great predictive performance and high interpretability.

The structure of the paper is as follows. Section 2 describes the decision rule generation method in HorseRule. Section 3 presents the horseshoe regularization prior and the MCMC algorithm for posterior inference. Section 4 illustrates aspects of the approach on simulated data and evaluates and compares the predictive performance of HorseRule to several main competing methods on a wide variety of real datasets. Section 5 concludes.

2. Decision rule generation. This section describes the *rule generation step* of HorseRule, which complements the rules from gradient boosting in Friedman and Popescu (2008) with rules from Random Forest with completely different properties.

2.1. *Decision rules.* Let S_k denote the set of possible values of the covariate x_k and let $s_{k,m} \subseteq S_k$ denote a specific subset. A decision rule can then be written as

$$(2.1) \qquad r_m(\boldsymbol{x}) = \prod_{k \in Q_m} I(x_k \in s_{k,m}),$$

where $I(x)$ is the indicator function and Q_m is the set of variables used in defining the mth rule. A decision rule $r_m \in \{0, 1\}$ takes the value 1 if all of its $|Q_m|$ conditions are fulfilled and 0 otherwise. For orderable covariates $s_{k,m}$ will be an
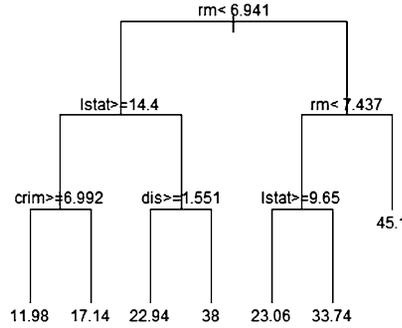
FIG. 1.   *Decision tree for the Boston housing data.*

interval or a disjoint union of intervals, while for categorical covariates $s_{k,m}$ are explicitly enumerated. There is a long tradition in machine learning to use decision rules as weak learners. Most algorithms learn decision rules directly from data, such as in Cohen (1995), Dembczyński, Kotłowski and Słowiński (2010). RuleFit exploits the fact that decision trees can be seen as a set of decision rules. In a first step a tree ensemble is generated, which is then decomposed into its defining decision rules. Several efficient (greedy) algorithmic implementations are available for constructing the tree ensembles. The generated rules typically correspond to interesting subspaces with great predictive power. Each node in a decision tree is defined by a decision rule. Figure 1 shows an example tree for the Boston housing dataset and Table 1 its corresponding decision rules. We briefly describe this dataset here since it will be used as a running example throughout the paper. The Boston housing data consists of $N = 506$ observations which are city areas in Boston and $p = 13$ covariates are recorded. These variables include ecological

TABLE 1
Corresponding rules, defining the decision tree

Rules	Conditions
r_1	$RM \geq 6.94$
r_2	$RM < 6.94$
r_3	$RM < 6.94$ & $LSTAT < 14.4$
r_4	$RM < 6.94$ & $LSTAT \geq 14.4$
r_5	$RM < 6.94$ & $LSTAT < 14.4$ & $CRIM < 6.9$
r_6	$RM < 6.94$ & $LSTAT < 14.4$ & $CRIM \geq 6.9$
r_7	$RM \geq 6.94$ & $LSTAT < 14.4$ & $DIS < 1.5$
r_8	$RM \geq 6.94$ & $LSTAT < 14.4$ & $DIS \geq 1.5$
r_9	$6.94 \leq RM < 7.45$
r_{10}	$6.94 \leq RM < 7.45$
r_{11}	$6.94 \leq RM < 7.45$ & $LSTAT < 9.7$
r_{12}	$6.94 \leq RM < 7.45$ & $LSTAT \geq 9.7$

measures of nitrogen oxides (NOX), particulate concentrations (PART) and proximity to the Charles River (CHAS), the socio-economic variables proportion of black population (B), property tax rate (TAX), proportion of lower status population (LSTAT), crime rate (CRIM), pupil teacher ratio (PTRATIO), proportion of old buildings (AGE), the average number of rooms (RM), area proportion zoned with large lots (ZN), the weighted distance to the employment centers (DIS) and an index of accessibility to key infrastructure (RAD). The dependent variable is the median housing value in the area.

Using equation (2.1) for example, r_{11} can be expressed as

$$r_{11}(\boldsymbol{x}) = \prod_{k \in Q_{11}} I(x_k \in s_{k,11}) = I(6.94 \leq RM < 7.45)I(LSTAT < 9.7).$$

This rule is true for areas with relatively large houses with between 6.94 and 7.45 rooms and less than 9.7% lower status population. The mth tree consists of $2(u_m - 1)$ rules, where u_m denotes the number of terminal nodes. Therefore $\sum_{m=1}^{M} 2(u_m - 1)$ rules can be extracted from a tree ensemble of size M.

2.2. *Collinearity structure of trees.*   The generated rules will be combined in a linear model and collinearity is a concern. For example, the two first child nodes in each tree are perfectly negative correlated. Furthermore, each parent node is perfectly collinear with its two child nodes, as it is their union. One common way to deal with the collinearity problem is to include the terminal nodes only. This approach also reduces the number of rules and therefore simplifies computations. We have nevertheless chosen to consider all possible rules including also nonterminal ones, but to randomly select one of the two child nodes at each split. The reason for also including nonterminal nodes is three-fold. First, even though each parent node in a tree can be reconstructed as a linear combination of terminal nodes, when using regularization this equivalence no longer holds. Second, our complexity penalizing prior in Section 3.3 is partly based on the number of splits to measure the complexity of a rule, and will therefore shrink the several complex child nodes needed to approximate a simpler parent node. Third, the interpretation of the model is substantially simplified if the model can select a simple parent node instead of many complex child nodes.

2.3. *Generating an informative and diverse rule ensemble.*   Any tree method can be used to generate decision rules. Motivated by the experiments in Friedman and Popescu (2003), Rulefit uses gradient boosting for rule generation [Friedman and Popescu (2008)]. Gradient boosting [Friedman (2001)] fits each tree iteratively on the pseudo residuals of the current ensemble in an attempt to correct mistakes made by the previous ensemble. This procedure introduces a lot of dependence between the members of the ensemble, and many of the produced rules tend to be informative only when combined to an ensemble. It might therefore not be possible to remove a lot of the decision rules without destroying this dependency structure.

Random Forest on the other hand generates trees independently from all previous trees [Breiman (2001)]. Each tree tries to find the individually best partitioning, given a random subset of observations and covariates. Random Forest will often generate rules with very similar splits, and the random selection of covariates forces it to often generate decision rules based on uninformative predictors. Random Forest will therefore produce more redundant and uninformative rules compared to gradient boosting, but the generated rules with strong predictive power are not as dependent on the rest of the ensemble.

Since the rules from boosting and Random Forest are very different in nature, it makes sense to use both types of rules to exploit both methods' advantages. This naturally leads to a larger number of candidate rules, but the generalized horseshoe shrinkage proposed in Section 3.2 and 3.3 can very effectively handle redundant rules. Traditional model combination methods usually use weighting schemes on the output of different ensemble methods [Rokach (2010)]. In contrast we combine the extracted rules from the individual trees. To the best of our knowledge this combination of individual weak learners from different ensemble methods is novel and fits nicely in the RuleFit framework with horseshoe regularization, as explained in the Introduction.

The tuning parameters used in the tree generation determine the resulting decision rules. The most impactful is the tree-depth, controlling the complexity of the resulting rules. We follow Friedman and Popescu (2008) with setting the depth of tree m to

$$(2.2) \qquad td_m = 2 + \lfloor \varphi \rfloor,$$

where $\lfloor x \rfloor$ is the largest integer less or equal than x and φ is a random variable following the exponential distribution with mean $L - 2$. Setting $L = 2$ will produce only tree stumps consisting of one split. With this indirect specification the forest is composed of trees of varying depth, which allows the model to be more adaptive to the data and makes the choice of a suitable tree depth less important. We use this approach for both boosted and random forest trees.

Another important parameter is the minimum number of observations in a node $n_{\min}$. A too small $n_{\min}$ gives very specific rules and the model is likely to capture spurious relationships. Using $n_{\min} = N^{\frac{1}{3}}$ as a default setting has worked well in our experiments, but if prior information about reasonable sizes of subgroups in the data is available the parameter can be adjusted accordingly. Another choice is to determine $n_{\min}$ by cross validation.

In the following, all other tuning parameters, for example, the shrinkage parameter in gradient boosting or the number of splitting covariates in the Random Forest, are set to their recommended standard choices implemented in the R-packages *randomForest* and *gbm*.

3. Ensembles and rule based horseshoe regularization. This section discusses the *regularization step* of HorseRule and present a new horseshoe shrinkage prior tailored specifically for covariates in the form of decision rules.

3.1. *The ensemble.* Once a suitable set of decision rules is generated, they can be combined in a linear regression model of the form

$$y = \alpha_0 + \sum_{l=1}^{m} \alpha_l r_l(\mathbf{x}) + \varepsilon.$$

As $r_i(\mathbf{x}) \in \{0, 1\}$ they already have the form of dummy variables and can be directly included in the regression model. A simple but important extension is to also include linear terms

$$(3.1) \qquad y = \alpha_0 + \sum_{j=1}^{p} \beta_j x_j + \sum_{l=1}^{m} \alpha_l r_l(\mathbf{x}) + \varepsilon.$$

This extension addresses the difficulty of rule and tree based methods to approximate linear effects. Splines, polynomials, time effects, spatial effects or random effects are straightforward extensions of equation (3.1).

Friedman and Popescu (2008) do not standardize the decision rules, which puts a higher penalty on decision rules with a smaller scale. To avoid this behavior, we scale the predictors to have zero mean and unit variance.

3.2. *Bayesian regularization through the horseshoe prior.* A large set of candidate decision rules is usually necessary to have a high enough chance of finding good decision rules. The model in (3.1) will therefore always be high dimensional and often $p + m > n$. Many of the rules will be uninformative and correlated with each other. Regularization is therefore a necessity.

RuleFit uses L1-regularized estimates, which corresponds to an a posterior mode estimator under a double exponential prior in a Bayesian framework [Tibshirani (1996)]. As discussed in the Introduction, the global shrinkage effect of L1-regularization can be problematic for rule covariates. L1-regularization is well known to lead to both shrinkage and variable selection. There now exist implementations of RuleFit that use the elastic net instead of L1-Regularization, which can lead to improved predictive performance [Zou and Hastie (2005)], however elastic net still only uses one global shrinkage parameter.

Another common Bayesian variable selection approach is based on the spike-and-slab prior [George and McCulloch (1993), Smith and Kohn (1996)]

$$(3.2) \qquad \beta_j \sim w \cdot N(\beta_j; 0, \lambda^2) + (1 - w) \cdot \delta_0,$$

where δ_0 is the Dirac point mass function, $N(\beta_j; 0, \lambda^2)$ is the normal density with zero mean and variance λ^2, and w is the prior inclusion probability of predictor x_j. Discrete mixture priors enjoy attractive theoretical properties, but need to explore a model space of size $2^{(p+m)}$, which can be problematic when either p or m are large. The horseshoe prior by Carvalho, Polson and Scott (2009, 2010) mimics the behavior of the spike-and-slab but is computationally more attractive. The

regression model with the original horseshoe prior for linear regression is of the form

$$(3.3) \qquad\qquad y|\mathbf{X}, \boldsymbol{\beta}, \sigma^2 \sim \mathcal{N}_n(\mathbf{X}\boldsymbol{\beta}, \sigma^2 \mathbf{I}_n),$$

$$(3.4) \qquad\qquad \beta_j|\lambda_j, \tau^2, \sigma^2 \sim \mathcal{N}(0, \lambda_j \tau^2 \sigma^2),$$

$$(3.5) \qquad\qquad \sigma^2 \sim \sigma^{-2} d\sigma^2,$$

$$(3.6) \qquad\qquad \lambda_j \sim \mathcal{C}^+(0, 1),$$

$$(3.7) \qquad\qquad \tau \sim \mathcal{C}^+(0, 1),$$

where $\mathcal{C}^+(0, 1)$ denotes the standard half-Cauchy distribution. We use horseshoe priors on both linear [the β's in equation (3.1)] and rule terms [the α's in equation (3.1)]. The horseshoe shrinkage for β_j is determined by a local shrinkage parameter $\lambda_j > 0$ and a global shrinkage parameter $\tau > 0$. This is important since it allows aggressive shrinking of noise covariates through small values of τ, while allowing individual signals to have large coefficients through large λ_j. Carvalho, Polson and Scott (2010) show that the horseshoe is better at recovering signals than the Lasso, and the models obtained from the horseshoe are shown to be almost indistinguishable from the ones obtained by a well defined spike-and-slab prior.

3.3. *Horseshoe regularization with rule structure.* The original horseshoe assigns the same prior distribution to all regression coefficients, regardless of the rule's complexity (number of splits in the tree) and the specificity (number of data points that fulfill the rule). Similar to the tree structure prior in BART, we therefore modify the horseshoe prior to express the prior belief that rules with high length (many conditions) are less likely to reflect a true mechanism. In addition, we also add the prior information that very specific rules that are satisfied by only a few data points are also improbable a priori. The rule support $s(r_l) \in (0, 1)$ is given by $s(r_j) = N^{-1} \sum_{i=1}^N r_j(\mathbf{x}_i)$. Note that a support of 95% can also be interpreted as 5%. Therefore we express the specificity of a rule through $\min(1 - s(r_j), s(r_j))$ instead. These two sources of prior information are incorporated by extending the prior on λ_j to

$$\lambda_j \sim \mathcal{C}^+(0, A_j),$$

with

$$(3.8) \qquad\qquad A_j = \frac{(2 \cdot \min(1 - s(r_j), s(r_j)))^\mu}{(l(r_j))^\eta},$$

where $l(r_j)$ denotes the length of rule j defined as its number of conditions. With increasing number of conditions the prior shrinkage becomes stronger, as well as with increasing specificity. The hyperparameter μ controls the strength of our

belief to prefer general rules that cover a lot of observations and η determines how strongly we prefer simple rules. The response y should be scaled when using the rule structure prior since the scale of β depends on the scale of y.

The rule structure for A_j in equation (3.8) is designed such that $A_j = 1$ for rules with support 0.5 and length 1, as the ideal. Since $\lim_{\mu \to 0, \eta \to 0} A_j = 1$, our rule structure prior approaches the standard horseshoe prior for small μ and η. The rule structure prior gives a gentle push towards simple and general rules, but its Cauchy tails put considerable probability mass on nonzero values even for very small A_j; the data can therefore overwhelm the prior and keep a complex and specific rule if needed.

A model with many complex specific rules may drive out linear terms from the model, thereby creating an unnecessarily complicated model. We therefore use a standard prior with $A = 1$ for linear terms, and set the parameters μ and η to values larger than 0, which has the effect of giving linear effects a higher chance of being chosen a priori. When p is small it may also be sensible to use no shrinkage at all on the linear effects, and this is also allowed in our Gibbs sampling algorithm in Section 3.4. The hyperparameters μ and η can be chosen guided by theoretical knowledge about what kind of rules and linear effects are reasonable for a problem by hand, or determined via cross validation. As a default choice $(\mu, \eta) = (1, 2)$ worked well in our experiments, penalizing rule complexity heavily and low rule support moderately.

3.4. *Posterior inference via Gibbs sampling.* Posterior samples can be obtained via Gibbs sampling. Sampling from the above hierarchy is expensive, as the full conditionals of λ_j and τ do not follow standard distributions and slice sampling has to be used. Makalic and Schmidt (2016) propose an alternative Horseshoe hierarchy that exploits the following mixture representation of a half-Cauchy distributed random variable $X \sim \mathcal{C}^+(0, \Psi)$:

$$(3.9) \qquad X^2 | \psi \sim \mathcal{IG}\left(\frac{1}{2}, \frac{1}{\psi}\right),$$

$$(3.10) \qquad \psi \sim \mathcal{IG}\left(\frac{1}{2}, \frac{1}{\Psi^2}\right),$$

which leads to conjugate conditional posterior distributions. The sampling scheme in Makalic and Schmidt (2016) samples iteratively from the following set of full conditional posteriors:

$$\boldsymbol{\beta}|\cdot \sim \mathcal{N}_p(\mathbf{A}^{-1}\mathbf{X}^T\mathbf{y}, \sigma^2\mathbf{A}^{-1}),$$

$$\sigma^2|\cdot \sim \mathcal{IG}\left(\frac{n+p}{2}, \frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})^T(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})}{2} + \frac{\boldsymbol{\beta}^T\boldsymbol{\Lambda}_*^{-1}\boldsymbol{\beta}}{2}\right),$$

$$\lambda_j^2|\cdot \sim \mathcal{IG}\left(1, \frac{1}{v_j} + \frac{\beta_j^2}{2\tau^2\sigma^2}\right),$$

$$\tau^2|\cdot \sim \mathcal{IG}\left(\frac{p+1}{2}, \frac{1}{\rho} + \frac{1}{2\sigma^2}\sum_{j=1}^{p}\frac{\beta_j{}^2}{\lambda_j{}^2}\right),$$

$$\nu_j|\cdot \sim \mathcal{IG}\left(1, \frac{1}{A^2} + \frac{1}{\lambda_j^2}\right),$$

$$\rho|\cdot \sim \mathcal{IG}\left(1, 1 + \frac{1}{\tau^2}\right),$$

with $\mathbf{A} = (\mathbf{X}^T\mathbf{X} + \mathbf{\Lambda}_*{}^{-1})$, $\mathbf{\Lambda}_* = \tau^2\mathbf{\Lambda}$, $\mathbf{\Lambda} = \mathrm{diag}(\lambda_1{}^2, \ldots, \lambda_p{}^2)$.

3.5. *Computational considerations.* The computational complexity of HorseRule can be mainly composed in rule generation and weight learning. The computational cost will thereby always be higher than using boosting or Random Forest alone. This speed disadvantage is partly mitigated by the fact that the HorseRule performs well also without cross-validation.

We have used the R implementations gbm and randomForest here. These algorithms do not scale well for large N and p and become a bottleneck for $N > 10,000$. This can be easily remedied by migrating the rule generation step to Xtreme Gradient Boosting (XGBoost) [Chen and Guestrin (2016)] or lightGBM [Ke et al. (2017)] that are magnitudes faster for big datasets.

Compared to Bayesian tree learning procedures such as BART or the recently proposed Dirichlet Adaptive Regression Trees (DART) [Linero (2018)], no Metropolis–Hastings steps are necessary to learn the tree structure in HorseRule; HorseRule uses only Gibbs sampling on a regularized linear model with rule covariates, which scales linearly with the number of observations [Makalic and Schmidt (2016)]. Sampling 1000 draws from the posterior distribution in the HorseRule model for the Boston housing data used in Section 4.7 takes about 90 seconds on a standard computer. The complexity of the Horseshoe sampling depends mostly on the number of linear terms and decision rules, and increases only slowly with N. Li and Yao (2014) suggest a computational shortcut where a given β_j is sampled in a given iteration only if the corresponding scale $(\lambda_j \cdot \tau)$ is higher than a threshold. The λ_j needs to be sampled in every iteration to give every covariate the chance of being chosen in the next iteration. We have implemented this approach and seen that it can give tremendous computational gains, but we have not used it when generating the results here since the effects it has on the invariant distribution of the MCMC scheme needs to be explored further. Finally, for very large N ($> 10,000$) the linear algebra operations in the Gibbs sampling can become time consuming, and GPU acceleration can be used to speed up sampling [Terenin, Dong and Draper (2016)].

3.6. *Sampling the splitting points.* The BART model can be seen as the sum of trees with a Gaussian prior on the terminal node values

$$\mu_j \sim \mathcal{N}\left(0, \frac{0.5}{\tau\sqrt{k}}\right),$$

where k denotes the number of trees. BART uses a fixed regularization parameter τ and samples the tree structure, while HorseRule uses a fixed rule structure and adapts to the data through sampling the shrinkage parameters λ_j and τ. Using a fixed tree structure offers dramatic computational advantages, as no Metropolis–Hastings updating steps are necessary, but the splits are likely to be suboptimal with respect to the whole ensemble.

As shown in Section 4, both HorseRule and BART achieve great predictive performance through different means, and a combination in which both shrinkage and tree structure are sampled in a fully Bayesian way could be very powerful, but computational very demanding. An intermediate position is to keep the splitting variables fixed in HorseRule, but to sample the splitting points. We have observed that HorseRule often keeps very similar rules with slightly different splitting points in the ensemble, which is a discrete approximation to sampling the splitting points. Hence this could also improve interpretability since a large number of rules with nearby splitting points can be replaced by a single rule with an estimated splitting point. It is also possible to replace many similar rules with suitable basis expansions, such as cubic terms or splines.

4. Results. This section starts out with a predictive comparison of HorseRule against a number of competitors on 16 benchmark datasets. The following subsections explore several different aspects of HorseRule on simulated and real data to evaluate the influence of different components of the model. Section 4.2 contrasts the ability of RuleFit and HorseRule to recover a true linear signal in models with additional redundant rules. The following subsection uses two real datasets to demonstrate the effect of having linear effects in HorseRule, and the advantage of using horseshoe instead of L1 for regularization. Section 4.4 addresses that HorseRule uses the training data both to generate the rules and for learning the weights. Section 4.5 explores the role of the rule generating process in HorseRule, and Section 4.6 the sensitivity to the number of rules. Finally in Sections 4.7 and 4.8 we showcase HorseRule's ability to make interpretable inference from data in different domains.

4.1. *Prediction performance comparison on* 16 *datasets.* We compare the predictive performance of HorseRule with competing methods on 16 regression datasets. The datasets are a subset of the datasets used in Chipman, George and McCulloch (2010). From the 23 datasets that were available to us online we excluded datasets that lacked a clear description of which variable to use as response, or which data preprocessing has to be applied to get to the version described in

Summary of the 16 *regression datasets used in the evaluation. N*, *Q and C are the number of observations*, *quantitative and categorical predictors*, *respectively*

Name	N	Q	C	Name	N	Q	C
Abalone	4177	7	1	Diamond	308	1	3
AIS	202	11	1	Hatco	100	6	4
Attend	838	6	3	Heart	200	13	3
Baskball	96	4	0	Fat	252	14	0
Boston	506	13	0	MPG	392	6	1
Budget	1729	10	0	Ozone	330	8	0
CPS	534	7	3	Servo	167	2	2
CPU	209	6	1	Strike	625	4	1

Chipman, George and McCulloch (2010). Since both RuleFit and HorseRule assume Gaussian responses, we also excluded datasets with clearly non-Gaussian response variables, for example count variables with excessive number of zeros. HorseRule can be straightforwardly adapted by using a negative-binomial data augmentation scheme [Makalic and Schmidt (2016)], but we leave this extension for future work. Table 2 displays the characteristics of the datasets.

We compare HorseRule to RuleFit [Friedman and Popescu (2008)], Random Forest [Breiman (2001)], Bayesian Additive Regression Trees (BART) [Chipman, George and McCulloch (2010)], Dirichlet Adaptive Regression Trees (DART) [Linero (2018)], a recent variant of BART that uses regularization on the input variables, and XGBoost [Chen and Guestrin (2016)] a highly efficient implementation of gradient boosting.

We use 10-fold cross validation on each dataset and report the relative RMSE (RRMSE) in each fold; RRMSE for a fold is the RMSE for a method divided by the RMSE of the best method on that fold. This allows us to compare performance over different datasets with differing scales and problem difficulty. We also calculate a worst RRMSE (wRRMSE) on the dataset level, as a measure of robustness. wRRMSE is based on the maximal difference across all datasets between a method's RRMSE and the RRMSE of the best method for that dataset; hence a method with low wRRMSE is not far behind the winner on any dataset. We also calculate the mean RRMSE (mRRMSE) as the relative RMSE on dataset level averaged over all datasets.

To ensure a fair comparison we use another (nested) five-fold cross validation in each fold to find good values of the tuning parameters for each method. For BART and Random Forest the cross-validation settings from Chipman, George and McCulloch (2010) are used. DART is relatively independent of parameter tuning, through the usage of hyperpriors, so we only determine the optimal number of trees. For RuleFit we cross-validate over the number of rules and the depth of the trees, as those are the potentially most impactful parameters. The shrinkage τ

TABLE 3
Settings for the compared methods

Method	Parameter settings
HR-default	Ensemble: GBM+RF; $L = 5$; $(\mu, \eta) = (1, 2)$.
HR-CV	Ensemble: GBM+RF; $L = (2, 5, 8)$; $(\mu, \eta) = ((0, 0), (0.5, 0.5), (1, 2))$.
RuleFit	$k = 500, 1000, \ldots, 5000$; $L = (2, 5, 8)$.
Random Forest	Fraction of variables used in each tree $= (0.25, 0.5, 0.75, 1, \sqrt{p}/p)$.
BART	$(\gamma, q) = ((3, 0.9), (3, 0.99), (10, 0.75))$; $\tau = 2, 3, 5$; number of trees: 50, 200.
DART	Number of trees: 50, 100.
XGBoost	Number of trees: 50, 100, 200, 350, 500; $\nu = 0.1, 0.05, 0.01$; tree depth: 4, 6, 8.

in RuleFit is determined by the model internally. XGBoost has many parameters that can be optimized, we chose the number of trees, the shrinkage parameter and the tree depth as the most important. For HorseRule we use cross-validation to identify suitable hyperparameters (μ, η) as well as the tree depth. We also run a HorseRule version with the proposed standard settings without cross-validation. Table 3 summarizes the settings of all methods.

We first compare the three different HorseRule versions. Figure 2 shows the predictive performance of the HorseRule models over $10 \cdot 16 = 160$ dataset and cross-validation splits. While the $(\mu, \eta) = (1, 2)$ already performs better than the prior without rule structure $[(\mu, \eta) = (0, 0)]$, cross-validation of (μ, η) helps to improve performance further.
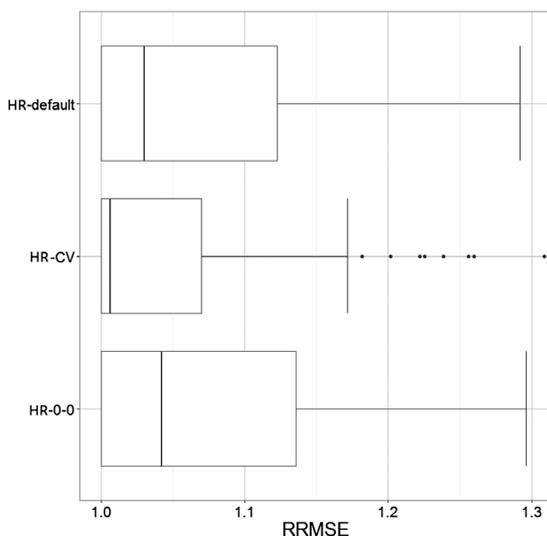

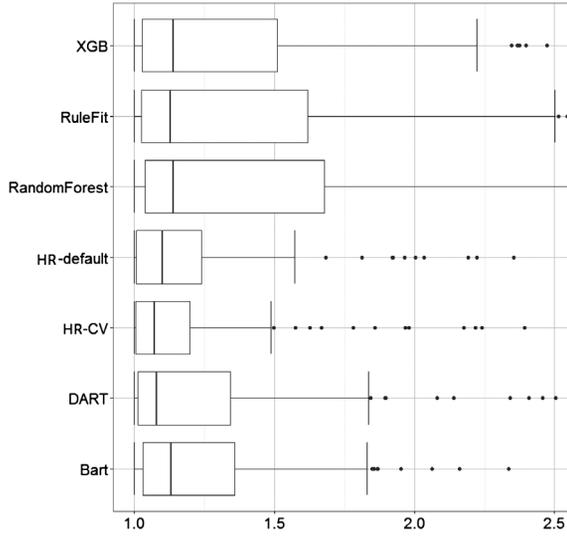FIG. 2. *RRMSE comparison of the different HorseRule versions across all folds.*

FIG. 3. *RRMSE comparison of HorseRule with competing methods across all folds.*

Figure 3 and Table 4 show that HorseRule has very good performance across all datasets and folds, and the median RRMSE is smaller than its competitors. DART also performs well and is second best in terms of median RRMSE. HorseRule-default is the third best method for the median and best for the mean, which is quite impressive since it does not use cross-validation.

Table 5 summarizes the performance on the dataset level. DART is the best model on 7/16 datasets and has the best average rank. HorseRule-CV is the best method on 5/16 datasets and has a slightly worse rank than DART. The last rows of Table 5 displays the wRRMSE and mRRMSE over all datasets for each method; it shows that whenever HorseRule is not the best method, it is only marginally behind the winner. This is not true for any of the other methods which all perform substantially worse than the best method on some datasets. RuleFit performs the

TABLE 4
RRMSE distribution over the 160 *crossvalidation folds of the competing methods*

	25%-Quant	Median	Mean	75%-Quant
XGBoost	1.02	1.139	1.496	1.509
RuleFit	1.026	1.129	1.426	1.618
RandomForest	1.039	1.137	1.508	1.677
HR-default	1.007	1.101	**1.247**	1.238
HR-CV	**1.004**	**1.072**	1.262	**1.198**
DART	1.012	1.080	1.376	1.342
BART	1.030	1.131	1.377	1.357

TABLE 5

Cross-validated prediction performance for the 16 *regression datasets. Each entry shows the RMSE and in parentheses the rank on this dataset. The best result is marked in bold*

	BART	RForest	RuleFit	HorseRule	HorseRule-CV	DART	XGBoost
Abalone	2.150 (7)	2.119 (3)	2.139 (5)	2.115(2)	**2.114** (1)	2.129 (4)	2.147 (6)
AIS	1.144 (4)	1.247 (7)	1.207 (6)	0.713 (2)	**0.699** (1)	1.061 (3)	1.188 (5)
Attend	394,141 (5)	411,900 (7)	**345,177** (1)	398,485 (6)	365,010 (2)	370,006 (4)	367,231 (3)
Baskball	0.087 (3)	0.086 (2)	0.088 (4)	0.088 (4)	0.092 (7)	**0.083** (1)	0.089 (6)
Boston	2.867 (2)	3.153 (7)	3.037 (6)	2.940 (4)	2.926 (3)	**2.819** (1)	2.97 (5)
Budget	0.039 (2)	**0.038** (1)	0.061 (7)	0.041 (4)	0.042 (5)	0.056 (6)	0.039 (2)
CPS	4.356 (3)	4.399 (6)	4.386 (5)	**4.348** (1)	4.370 (4)	4.353 (2)	4.448 (7)
CPU	41.52 (4)	54.08 (6)	54.50 (7)	**36.03** (1)	37.47 (3)	42.87 (5)	36.75 (2)
Diamond	215.0 (3)	465.9 (7)	233.7 (4)	184.5 (2)	**171.27** (1)	245.8 (5)	343.6 (6)
Hacto	0.453 (7)	0.311 (6)	0.297 (5)	0.261 (2)	**0.260** (1)	0.264 (3)	0.269 (4)
Heart	8.917 (2)	9.048 (3)	9.349 (7)	9.241 (5)	9.070 (4)	**8.869** (1)	9.310 (6)
Fat	1.306 (6)	1.114 (2)	1.173 (3)	1.264 (5)	1.245 (4)	**1.072** (1)	1.329 (7)
MPG	2.678 (3)	2.692 (5)	2.672 (2)	2.714 (6)	2.689 (4)	**2.642** (1)	2.750 (7)
Ozone	4.074 (3)	4.061 (2)	4.189 (7)	4.120 (4)	4.165 (5)	**4.054** (1)	4.174 (6)
Servo	0.588 (5)	0.486 (3)	0.502 (4)	0.409 (2)	**0.403** (1)	0.671 (6)	0.719 (7)
Strikes	458.4 (7)	453.7 (5)	447.7 (3)	449.2 (4)	447.2 (2)	**447.1** (1)	456.6 (6)
Av. Rank	3.9375	4.5625	4.8750	3.5625	3	**2.9375**	5.3125
wRRMSE	1.742	2.720	1.726	1.179	**1.160**	1.666	2.006
mRRMSE	1.128	1.250	1.182	1.051	**1.035**	1.141	1.201

best on 1/16 datasets, and the median RRMSE is slightly lower than for Random Forest and XGBoost. XGBoost has the hightest median RRMSE and rank in this experiment. This is probably due to the fact, that all methods except Random Forest rely to a certain degree on boosting and improve different aspects of it, making it a hard competition for XGBoost.

To summarize, the results show that HorseRule is a highly competitive method with a very stable performance across all datasets. The rule structured prior was found to improve predictive performance, and performs well also without time-consuming cross-validation of its hyperparameters.

4.2. *Regularization of linear terms and rules—RuleFit vs. HorseRule.* This subsection uses simulated data to analyse the ability of HorseRule and RuleFit to recover the true signal when the true relationship is linear and observed with noise. The data is generated with $X_i \sim \mathcal{N}(0, 1)$, $i = 1, \ldots, 100$, $Y = 5X_1 + 3X_2 + X_3 + X_4 + X_5 + \varepsilon$ and $\varepsilon \sim \mathcal{N}(0, 1)$. The first five predictors thus have a positive dependency with y of varying magnitude while the remaining 95 covariates are noise. Table 6 reports the results from 100 simulated datasets. RMSE measures the discrepancy between the fitted values and the true mean for unseen test data. RuleFit and HorseRule model use 500 rules in addition to the linear terms. The best

TABLE 6
Simulation study. The true effect is linear

	RMSE			$\Delta\beta_{\text{true}}$			$\Delta\beta_{\text{noise}}$		
	$n = 100$	$n = 500$	$n = 1000$	$n = 100$	$n = 500$	$n = 1000$	$n = 100$	$n = 500$	$n = 1000$
OLS	3.23	1.10	1.06	1.25	0.19	0.14	2302	3.78	2.54
Horseshoe Regression	1.14	1.01	1.01	0.40	0.18	0.13	1.72	0.70	0.49
HorseRule $\alpha = 0, \beta = 0$	1.54	1.02	1.01	1.99	0.39	0.29	2.74	0.22	0.15
HorseRule $\alpha = 1, \beta = 2$	1.25	1.02	1.01	1.15	0.37	0.28	3.14	0.37	0.24
RuleFit $k = 2000$	1.84	1.23	1.15	3.58	1.42	1.05	1.18	0.91	0.99

model in RMSE is as expected the Horseshoe regression without any rules. The OLS estimates without any regularization struggles to avoid overfitting with all the unnecessary covariates and does clearly worse than the other methods. HorseRule without the rule structure prior outperforms RuleFit, but adding a rule structured prior gives an even better result. The differences between the models diminishes quickly with the sample size (since the data is rather clean), the exception being RuleFit which improves at a much lower rate than the other methods. Table 6 also breaks down the results into the ability to recover the true linear signal, measured by $\Delta\beta_{\text{true}} = |(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) - (5, 3, 1, 1, 1)|_1$, and the ability to remove the noise covariates, measured by $\Delta\beta_{\text{noise}} = |(\beta_6, \ldots, \beta_{100}) - (0, \ldots, 0)|_1$. We see that the HorseRule's horseshoe prior is much better at recovering the true linear signal compared to RuleFit with its L1-regularization. OLS suffers from its inability to shrink away the noise.

Even though such clear linear effects are rare in actual applications, the simulation results in Table 6 shows convincingly that HorseRule will prioritize and accurately estimate linear terms when they fit the data well. This is in contrast to RuleFit which shrinks the linear terms too harshly and compensates the lack of fit with many rules. HorseRule will only try to add nonlinear effects through decision rules if they are really needed.

4.3. *Influence of linear terms in HorseRule, and regularizing by horseshoe instead of L1.* In this section we analyze to what extent HorseRule's good performance depends on having linear terms in the model, and how crucial the horseshoe regularization is for performance. We demonstrate the effect of these model specification choices on the two datasets Diamonds and Boston. The Diamonds dataset was selected since HorseRule is much better than its competitors on that dataset. The Boston data was chosen since it will be used for a more extensive analysis in
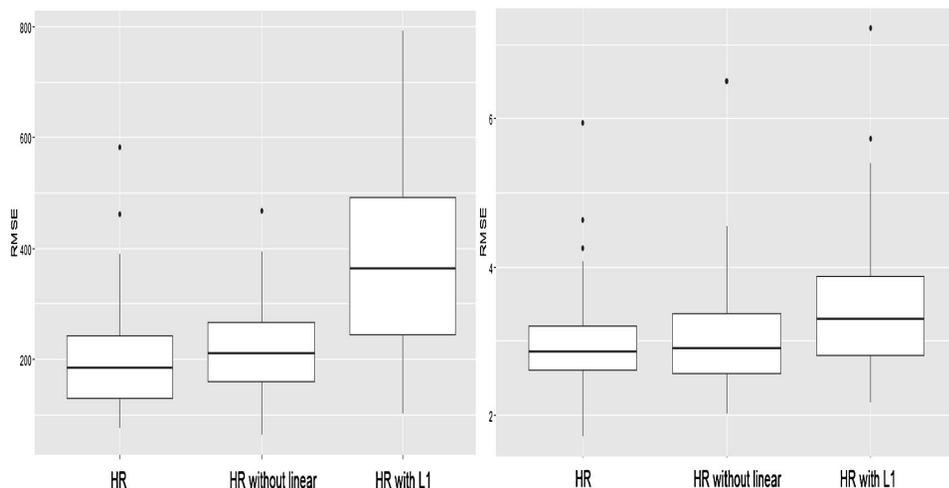
FIG. 4. *RMSE on the Diamond (*left*) and the Boston (*right*) dataset when linear terms are removed and when using L*1 *regularization instead of horseshoe.*

Section 4.7. Figure 4 shows the RMSE distribution over the folds used in 10-fold cross-validation. The results are replicated 10 times using different random seeds. The results show that the aggressive shrinkage offered of the horseshoe prior is essential for HorseRule; changing to L1 increases RMSE, especially for the Diamond data. Note that the L1-version is not entirely identical to RuleFit, as RuleFit uses different preprocessing on rules and only boosting generated rules [Friedman and Popescu (2008)]. Figure 4 also shows that adding linear terms gives small decrease of RMSE, but seems less essential for HorseRule's performance.

4.4. *Influence of the two-step procedure.* One concern of our two-step procedure is that the same training data is used to find rules and to learn the weights. This double use of the data can distort the posterior distribution and uncertainty estimates. It should be noted however that the rule generation uses only random subsets of data, which mitigates this effect to some extent. It is also important to point out that the predictive results presented in this paper are always on an unseen test set so this is not an issue for the performance evaluations.

One way to obtain a more coherent Bayesian interpretation is to split the training data in two parts: one part for the rule generation and one part for learning the weights. We can view this as conditionally coherent if the rule learned from the first part of the data is seen as prior experience of the analyst in analyzing the second part of the data. An obvious drawback with such an approach is that less data can be used for learning the model, which will adversely affect predictive performance. Table 7 displays how predictive performance on the Diamonds ($N = 308$) and Boston ($N = 506$) data deteriorates from a 50/50 split of the training data. Both these datasets are small and we have also included the moderately large Abalone

TABLE 7
Median RMSE for different splitting strategies

	Diamond	**Boston**	**Abalone**
All data	184.6	2.851	2.115
50/50 split	283.7	3.555	2.136

data ($N = 4177$); for this dataset the data splitting has essentially no effect on the performance. Hence, data-splitting may be an attractive option for moderately large and large data if proper Bayesian uncertainty quantification is of importance.

4.5. *Influence of the rule generating process.* In this section we analyze the influence of different rule generating processes on model performance for the Diamond dataset with ($N = 308$ and $p = 4$) and the Boston housing data ($N = 506$ and $p = 13$).

In each setting 1000 trees with an average tree depth of $L = 5$ are used, using different ensemble strategies for the rule generation:

1. Random Forest generated rules plus linear terms.
2. Gradient boosting generated rules plus linear terms.
3. A combination of 30% of the trees from Random Forest and 70% from gradient boosting plus linear terms.

The results are shown in Figure 5. As expected the error-correcting rules found by gradient boosting outperforms randomly generated rules from Random For-
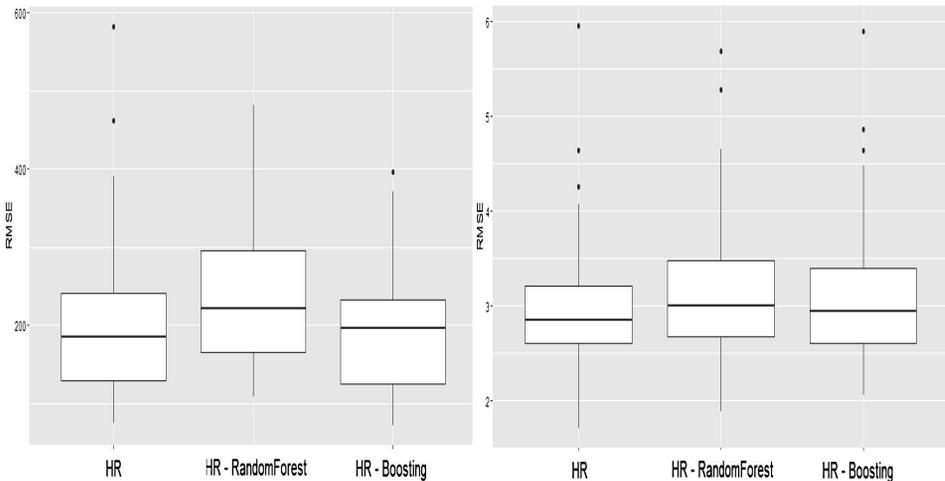

FIG. 5. *RMSE on the Diamond (left) and the Boston (right) dataset for different rule generating strategies.*
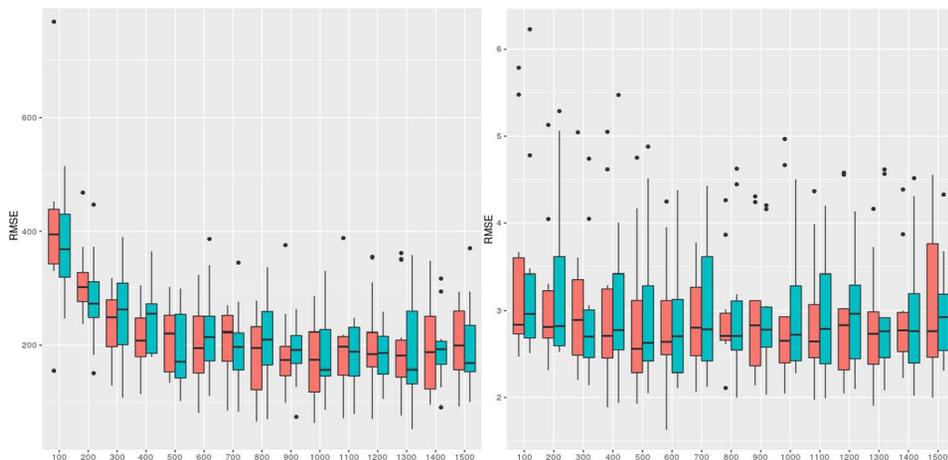
FIG. 6. *RMSE depending on the number of trees on the Diamond* (*left*) *and Boston* (*right*) *dataset for* $(\mu, \eta) = (0, 0)$ (*red*) *and* $(\mu, \eta) = (1, 2)$ (*blue*).

est. However, combining the two types of rules leads to a lower RMSE on both datasets. In our experiments it rarely hurts the performance to use both type of rules, and on some datasets it leads to a dramatically better prediction accuracy. The mixing proportion for the ensemble methods can also be seen as a tuning parameter to give a further boost in performance.

4.6. *Influence of the number of rules.* Another parameter that is potentially crucial is the number of trees used to generate the decision rules. In gradient boosting limiting the number of trees (iterations) is the most common way to control overfitting. Also in BART the number of trees has a major impact on the quality and performance of the resulting ensemble [Chipman, George and McCulloch (2010)]. The same is expected for RuleFit, as it uses L1-regularization; with an increasing number of rules the overall shrinkage τ increases, leading to an overshrinkage of good rules.

To investigate the sensitivity of HorseRule to the number of trees, we increase the number of trees successively from 100 to 1500 in the Boston and Diamond datasets. This corresponds to $500, 550, \ldots, 5 \cdot 1500 = 7500$ decision rules before removing duplicates. We also test if the rule structured prior interacts with the effect of the number of trees by running the model with $(\mu, \eta) = (0, 0)$ and $(\mu, \eta) = (1, 2)$. Figure 6 shows the performance of HorseRule as a function of the number of trees used to extract the rules. Both HorseRule models are relatively insensitive to the choice of k, unless the number of trees is very small. Importantly, no overfitting effect can be observed, even when using an extremely large number of 1500 trees on relatively small datasets ($N = 308$ and $N = 506$ observations, respectively). We use 1000 trees as a standard choice, but a small number of trees

TABLE 8
The 10 *most important rules in the Boston housing data*

Rule		$5\% I$	$\bar{I}$	$95\% I$	$\bar{\beta}$
1	$RM \leq 6.97$	0.96	0.99	1.00	24.1
	$LSTAT \leq 14.4$				
2	$RM \leq 6.97$	0.77	0.89	1.00	-21.9
	$DIS > 1.22$				
	$LSTAT \leq 14.4$				
3	$LSTAT \leq 4.66$	0.00	0.27	0.51	12.35
4	$TAX \leq 416.5$	0.00	0.21	0.43	-10.46
	$LSTAT \leq 4.65$				
5	$NOX \leq 0.59$	0.00	0.12	0.21	-2.94
6	$NOX \leq 0.67$	0.00	0.10	0.33	3.87
	$RM > 6.94$				
7	$NOX > 0.67$	0.00	0.11	0.37	-3.24
8	$LSTAT > 19.85$	0.00	0.15	0.53	-3.18
9	$linear : AGE$	0.00	0.09	0.15	-0.03
10	$linear : RAD$	0.00	0.07	0.19	0.10

can be used if computational complexity is an issue, with little to no expected loss in accuracy.

4.7. *Boston housing.* In this section we apply HorseRule to the well known Boston Housing dataset to showcase its usefulness in getting insights from the data. For a detailed description of the dataset see Section (2.1). The HorseRule with default parameter settings is used to fit the model. Table 8 shows the 10 most important effects. Following Friedman and Popescu (2008), the importance of a linear term is defined as

$$I(x_j) = |\beta_j| \text{sd}(x_j),$$

where $\text{sd}(\cdot)$ is the standard deviation, and similarly for a predictor from a decision rule

$$I(r_l) = |\alpha_l| \text{sd}(r_l).$$

We use the notation I_j when it is not important to distinguish between a linear term and a decision rule. For better interpretability we normalize the importance to be in [0, 1], so that the most important predictor has an importance of 1. Table 8 reports the posterior distribution of the normalized importance (obtained from the MCMC draws) of the 10 most important rules or linear terms. The most important single variable is LSTAT, which appears in many of the rules, and as a single variable in the third most important rule. Note also that LSTAT does not appear as a linear predictor among the most important predictors.
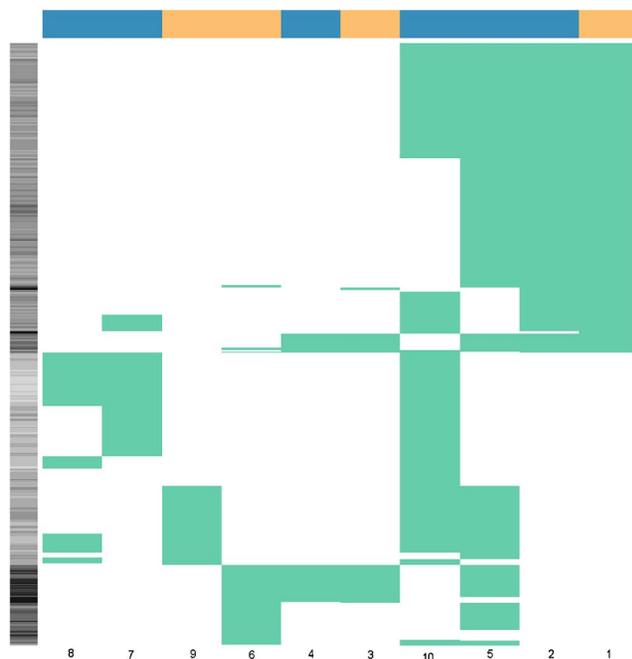
FIG. 7.   *RuleHeat for the Boston housing data. See the text for details.*

To interpret the more complex decision rules in Table 8 it is important to understand that decision rules in an ensemble have to be interpreted with respect to other decision rules, and in relation to the data points covered by a rule. A useful way to explore the effects of the most important rules is what we call a *RuleHeat* plot, see Figure 7 for an example for the Boston housing data. The horizontal axis lists the most important decision rules and the vertical axis the N observations. A square is green if $r_l(\mathbf{x}) = 1$. The grayscale on the bar to the left indicates the outcome (darker for higher price) and the colorbar in the top of the figure indicates the sign of the covariate's coefficient in the model (sand for positive). RuleHeat makes it relatively easy to to find groups of similar observations, based on the rules found in HorseRule, and to assess the role a rule plays in the ensemble. For example, Figure 7 shows that the two most important rules differ only in a few observations. The two rules have very large coefficients with opposite signs. Rule 1 in isolation implies that prices are substantially higher when the proportion of lower status population is low ($LSTAT \leq 14.4$) for all but the very largest houses ($RM \leq 6.97$). However, adding Rule 2 essentially wipes out the effect of Rule 1 ($24.1 - 21.9 = 2.2$) except for the six houses very close to the employment centers ($DIS < 1.22$) where the effect on the price remains high.

Similarly to the variable importance in Random Forest and RuleFit, we can calculate a variable input importance for the HorseRule model. The importance of
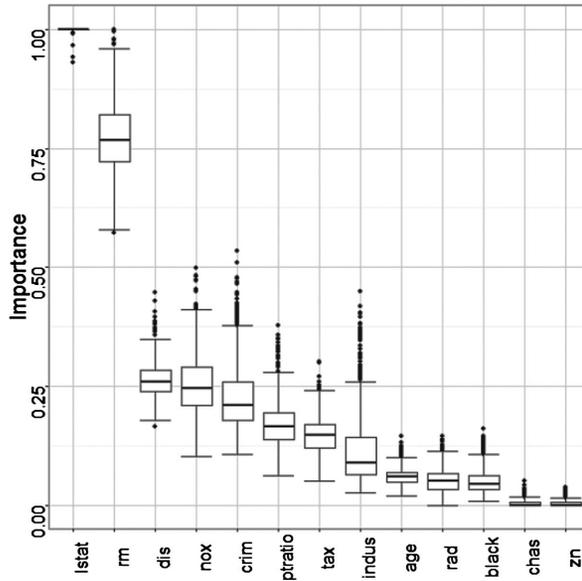
FIG. 8.   *Posterior distribution of the input variable importance for the* 13 *covariates.*

the jth predictor given the data is defined as [Friedman and Popescu (2008)]

$$J(x_j) = I(x_j) + \sum_{l:j \in Q_l} I(r_l)/|Q_l|,$$

where the sum runs over all rules where x_j is one of the predictors used to de-
fine the rule. Note how the importance of the rules are discounted by the number
of variables involved in the rule, $|Q_l|$. Figure 8 shows the posterior distribution
of $J(x_j)$ for the 13 covariates. LSTAT is the most important covariate with me-
dian posterior probability of 1 and very narrow posterior spread, followed by RM
which has a median posterior importance of around 0.75. The importance of some
variables, like NOX and INDUS, has substantial posterior uncertainty whereas for
other covariates, such as AGE, the model is quite certain that the importance is low
(but nonzero).

   The overlapping rules, as well as similar rules left in the ensemble in order to
capture model uncertainty about the splitting points make interpretation somewhat
difficult. One way to simplify the output from HorseRule is to use the *decoupling
shrinkage and summary* (DSS) approach by Hahn and Carvalho (2015). The idea
is to reconstruct the full posterior estimator $\hat{\beta}$ with a 1-norm penalized represen-
tation, that sets many of the coefficients to exactly zero and also merges together
highly correlated coefficients. We do not report systematic tests here, but in our
experiments using DSS with a suitable shrinkage parameter did not hurt the pre-
dictive performance, while allowing to set a vast amount of coefficients to zero.
Using HorseRule followed by DSS on the Boston housing data leaves 106 nonzero

TABLE 9
The ten most important rules in Boston data after DSS

Rule		$\bar{I}$	$\bar{\beta}$
1	$RM \leq 7.13$	1.00	−3.47
2	$RM \leq 6.98$	0.97	−2.36
	$PTRATIO \leq 18.7$		
	$LSTAT > 5.95$		
3	$LSTAT > 18.75$	0.81	1.80
4	$linear : RAD$	0.80	0.10
5	$RM \leq 7.437$	0.79	−2.03
	$LSTAT \leq 7.81$		
6	$NOX \leq 0.62$	0.70	−1.64
	$RM \leq 7.31$		
7	$RM \leq 7.1$	0.68	−2.47
	$RAD \leq 4.5$		
	$LSTAT \leq 7.81$		
8	$NOX > 0.59$	0.63	−1.47
9	$linear : LSTAT$	0.58	−0.09
10	$linear : AGE$	0.58	−0.02

coefficients in the ensemble. The 10 most important rules can be seen in Table 9. We can see that the new coefficients are now less overlapping. The relatively small number of rules simplify interpretation. Posterior summary for regression with shrinkage priors is an active field of research [see, e.g., Nalenz and Villani (2018), Piironen and Vehtari (2017) and Puelz, Hahn and Carvalho (2017) for interesting approaches] and future developments might help to simplify the rule ensemble further.

4.8. *Logistic regression on gene expression data*. Here we analyze how HorseRule can find interesting pattern in classification problems, specifically in using gene expression data for finding genes that can signal the presence or absence of cancer. Such information is extremely important since it can be used to construct explanations about the underlying biological mechanism that lead to mutation, usually in the form of gene pathways. Supervised gene expression classification can also help to design diagnostic tools and patient predictions, that help to identify the cancer type in early stages of the disease and to decide on suitable therapy [Van't Veer et al. (2002)].

Extending HorseRule to classification problems can be easily done using a latent variable formulation of, for example, the logistic regression. We chose to use the Pólya–Gamma latent variable scheme by Polson, Scott and Windle (2013). Methodological difficulties arise from the usually small number of available samples, as well as high number of candidate genes, leading to an extreme $p \gg n$

TABLE 10
Accuracy in training and test set for the prostate cancer data

	BART	**Random Forest**	**RuleFit**	**HorseRule**
CV-Accuracy	0.900	0.911	0.831	0.922
CV-AUC	0.923	0.949	0.953	0.976
Test-Accuracy	0.824	0.971	0.941	0.971
Test-AUC	1	0.991	0.995	1

situation. We showcase the ability of HorseRule to make inference in this diffi-
cult domain on the Prostate Cancer dataset, which consists of 52 cancerous and 50
healthy samples ($n = 102$). In the original data $p = 12{,}600$ genetic expressions are
available, which can be reduced to 5966 genes after applying the preprocessing de-
scribed in Singh et al. (2002). Since spurious relationships can easily occur when
using higher order interactions in the $p \gg n$ situation, we use the hyperparameters
$\mu = 2$ and $\eta = 4$ to express our prior belief that higher order interactions are very
unlikely to reflect any true mechanism.

Table 10 shows that HorseRule has higher accuracy and significantly higher
AUC than the competing methods. We also test the methods on an unseen test
dataset containing 34 samples not used in the previous step. All methods have
lower error here, implying that the test data consists of more predictable cases.
The difference is smaller, but HorseRule performs slightly better here as well.

The 10 most important rules for HorseRule are founds in Table 11. It contains
eight rules with one condition and only two with two conditions, implying that
there is not enough evidence in the data for complicated rules to overrule our prior
specification. All of the most important rules still contain 0 in their 5% posterior
importance distribution, implying that they are eliminated by the model in at least
5% of the samples; the small sample size leads to nonconclusive results.

Figure 9 shows the input variable importance of the 50 most important genes.
In this domain the advantage of having estimates of uncertainty can be very ben-
eficial, as biological follow up studies are costly and the probability of spurious
relationships is high. In this data the genes 37,639_*at* and 556_*s_at* contain an
importance of 1 in their 75% posterior probability bands. The gene 37,639_*at* was
found in previous studies to be the single gene most associated with prostate cancer
[Yap et al. (2004)]. However, gene 556_*s_at*, which makes up the most important
Rule 1, was only found to be the ninth important in previous studies on the same
data using correlation based measures [Yap et al. (2004)]. So, while this gene is
individually not very discriminative (77% accuracy), it becomes important in con-
junction with other rules. This is also borne out in the RuleHeat plot in Figure 10.
The outcome is binary, and the vertical bar to the left is red for cancer and black for
healthy. RuleHeat shows that Rule 1 covers all except one cancer tissue together
with a number of normal tissues, and would therefore probably not be found to be

TABLE 11
10 most important rules in the cancer data

Rule		$5\%I$	$\bar{I}$	$95\%I$	$\bar{\beta}$
1	$556\_s\_at \le 1.55$	0	0.33	1	3.10
2	$34,647\_at \le -1.18$	0	0.15	1	-1.78
	$37,639\_at \le 1$				
3	$37,478 > -0.32$	0	0.18	0.91	1.42
4	$38,087\_s\_at \le 0.83$	0	0.23	1	1.81
5	$34,678\_at > 0.38$	0	0.19	0.88	-1.58
6	$1243\_at \le 0.35$	0	0.15	0.66	1.19
7	$37,639\_at \le 1$	0	0.13	0.80	-1.10
8	$33,121\_g\_at \le 0.672$	0	0.10	0.82	-1.09
	$960\_g\_at > 0.378$				
9	$41,706\_at \le 1.33$	0	0.15	0.79	-1.13
10	$39,061\_at > 0.31$	0	0.1	0.52	-1.03

significant using traditional tests in logistic regression. Its importance arises from the combination with the other rules, especially Rule 2, Rule 7 and Rule 8, that are able to correct the false positive predictions using Rule 1 alone.

To illustrate HorseRule's potential for generating important insights from interaction rules, we present the subspaces of the two most important interaction rules in Figure 11 and Figure 12. Again healthy tissues are colored black and cancerous red. The first interaction looks somewhat unnatural. The gene $37,639\_at$ is individ-
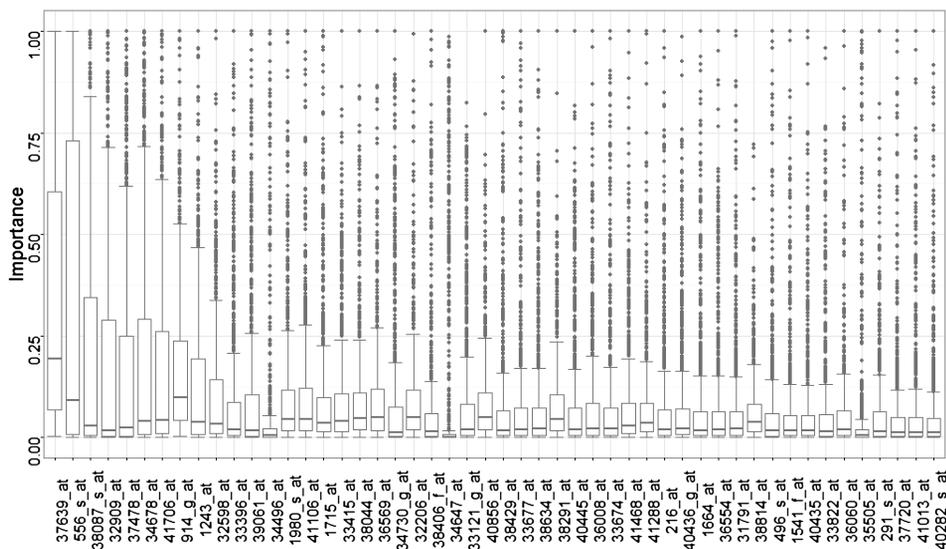

FIG. 9. *Posterior distribution of the input variable importance of the 50 most influential covariates.*
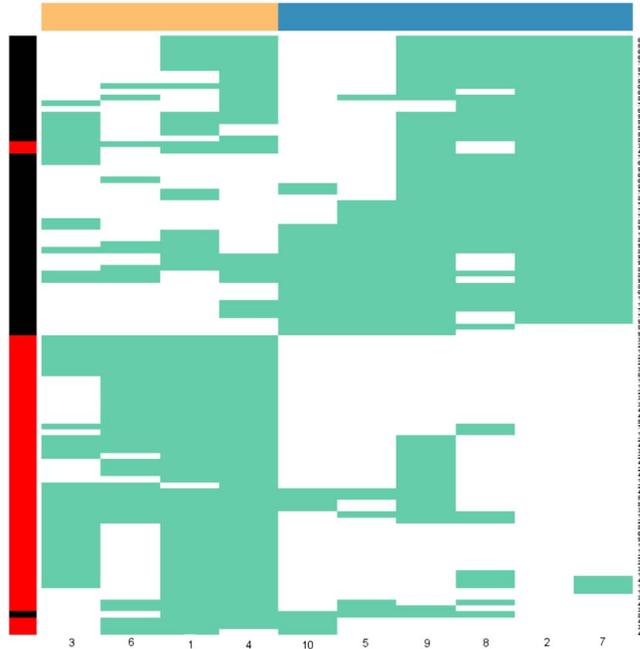
FIG. 10.   *RuleHeat for the prostate cancer data. Cancer tissues are colored in red, healthy in black.*

ually seen to be a strong classifier where higher values indicate cancer. This rule is also individually represented as Rule 7. The second split on 34,647_*at* < −1.18 corrects three misclassified tissues by the first split alone. This rule probably only works well in the ensemble but may not reflect a true mechanism. The second interaction effect is more interesting. It seems that healthy tissues have lower values in the expression of 33,121_*g_at* and higher values in the expression of 960_*g_at*. This rule might reflect a true interaction mechanism and could be worth analysing further.

Overall, this shows that HorseRules nonlinear approach with interacting rules complements the results from classical linear approaches with new information. Decision rules are especially interesting for the construction of gene-pathways [Glaab, Garibaldi and Krasnogor (2010)], diagnostic tools and identification of targets for interventions [Slonim (2002)].

5. Conclusions. We propose HorseRule, a new model for flexible nonlinear regression and classification. The model is based on RuleFit and uses decision rules from a tree ensemble as predictors in a regularized linear fit. We replace the L1-regularization in RuleFit with a horseshoe prior with a hierarchical structure especially tailored for a situation with decision rules as predictors. Our prior shrinks complex (many splits) and specific (small number of observations satisfy the rule) rules more heavily a priori, and is shown to be efficient in removing noise without
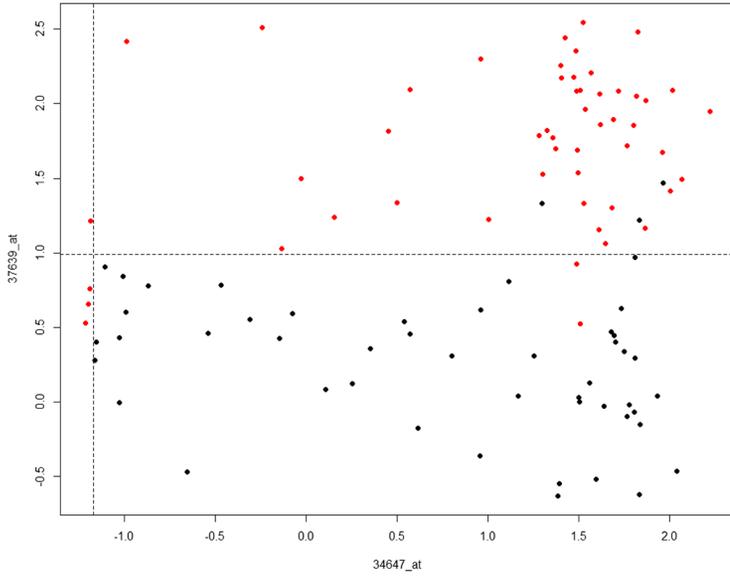
FIG. 11.   *Scatterplot for Genes* 37,639_at *and* 34,647_at. *Healthy samples in black and cancerous samples in red. Rule* 2 *is defined by the bottom right quadrant.*
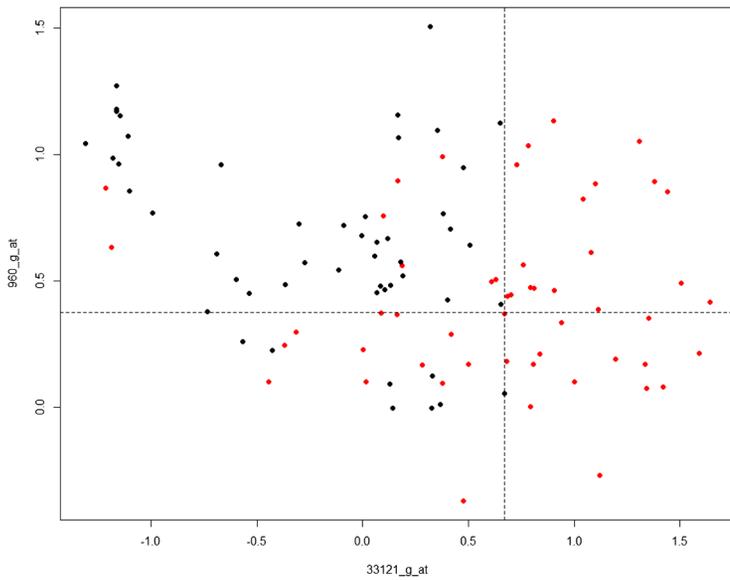

FIG. 12.   *Scatterplot for Genes* 33,121_g_at *and* 960_g_at. *Healthy samples in black and cancerous samples in red. Rule* 8 *is defined by the top left quadrant.*

tampering with the signal. The efficient shrinkage properties of the new prior also makes it possible to complement the rules from boosting used in RuleFit with an additional set of rules from random forest. The rules from Random Forest are not as tightly coupled as the ones from boosting, and are shown to improve prediction performance compared to using only rules from boosting.

HorseRule is shown to outperform state-of-the-art competitors like RuleFit, BART and Random Forest in an extensive evaluation of predictive performance on 16 widely used datasets. Importantly, HorseRule performs consistently well on all datasets, whereas the other methods perform quite poorly on some of the datasets. We explored different aspect of HorseRule to determine the underlying factors behind its success. We found that the combination of mixing rule from different tree algorithms and the aggressive but signal-preserving horseshoe shrinkage are essential, but that the addition of linear terms seems less important. Our experiments also show that the predictive performance of HorseRule is not sensitive to its prior hyperparameters. We also demonstrate the interpretation of HorseRule in both a regression and a classification problem. HorseRule's use of decision rules as predictors and its ability to keep only the important predictors makes it easy to interpret its results, and to explore the importance of individual rules and predictor variables.

Acknowledgements. We are grateful to the two reviewers and the Associate Editor for constructive comments that helped to improve both the presentation and the contents of the paper.

SUPPLEMENTARY MATERIAL

The HorseRule R-package (DOI: 10.1214/18-AOAS1157SUPP; .pdf). Example code illustrating the basic features of our HorseRule package in R with standard settings. The package is available on CRAN at https://CRAN.R-project.org/package=horserule.

REFERENCES

BREIMAN, L. (1996). Stacked regressions. *Mach. Learn.* **24** 49–64.

BREIMAN, L. (2001). Random forests. *Mach. Learn.* **45** 5–32.

CARVALHO, C. M., POLSON, N. G. and SCOTT, J. G. (2009). Handling sparsity via the horseshoe. In *AISTATS* **5** 73–80.

CARVALHO, C. M., POLSON, N. G. and SCOTT, J. G. (2010). The horseshoe estimator for sparse signals. *Biometrika* **97** 465–480. MR2650751

CHEN, T. and GUESTRIN, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the* 22*nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 785–794. ACM, New York.

CHIPMAN, H. A., GEORGE, E. I. and MCCULLOCH, R. E. (2010). BART: Bayesian additive regression trees. *Ann. Appl. Stat.* **4** 266–298. MR2758172

COHEN, W. W. (1995). Fast effective rule induction. In *Proceedings of the* 12*th International Conference on Machine Learning* (*ICML*'95) 115–123.

DEMBCZYŃSKI, K., KOTŁOWSKI, W. and SŁOWIŃSKI, R. (2010). ENDER: A statistical framework for boosting decision rules. *Data Min. Knowl. Discov.* **21** 52–90. MR2720513

FREUND, Y. and SCHAPIRE, R. E. (1996). Experiments with a new boosting algorithm. In *Machine Learning*: *Proceedings of the Thirteenth International Conference* **96** 148–156. Bari, Italy.

FRIEDMAN, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Ann. Statist.* **29** 1189–1232. MR1873328

FRIEDMAN, J. H. and POPESCU, B. E. (2003). Importance sampled learning ensembles. Technical report, Dept. Statistics, Stanford Univ., Stanford, CA.

FRIEDMAN, J. H. and POPESCU, B. E. (2008). Predictive learning via rule ensembles. *Ann. Appl. Stat.* **2** 916–954. MR2522175

FÜRNKRANZ, J. (1999). Separate-and-conquer rule learning. *Artif. Intell. Rev.* **13** 3–54.

GEORGE, E. I. and MCCULLOCH, R. E. (1993). Variable selection via Gibbs sampling. *J. Amer. Statist. Assoc.* **88** 881–889.

GLAAB, E., GARIBALDI, J. M. and KRASNOGOR, N. (2010). Learning pathway-based decision rules to classify microarray cancer samples.

HAHN, P. R. and CARVALHO, C. M. (2015). Decoupling shrinkage and selection in Bayesian linear models: A posterior summary perspective. *J. Amer. Statist. Assoc.* **110** 435–448. MR3338514

KE, G., MENG, Q., FINLEY, T., WANG, T., CHEN, W., MA, W., YE, Q. and LIU, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems* 3149–3157.

LI, L. and YAO, W. (2014). Fully Bayesian logistic regression with hyper-LASSO priors for high-dimensional feature selection. *J. Stat. Comput. Simul.* **88** 2827–2851. MR3827411

LINERO, A. R. (2018). Bayesian regression trees for high dimensional prediction and variable selection. *J. Amer. Statist. Assoc.* **113** 626–636. MR3832214

MAKALIC, E. and SCHMIDT, D. F. (2016). A simple sampler for the horseshoe estimator. *IEEE Signal Process. Lett.* **23** 179–182.

NALENZ, M. and VILLANI, M. (2018). Supplement to "Tree ensembles with rule structured horseshoe regularization." DOI:10.1214/18-AOAS1157SUPP.

PIIRONEN, J. and VEHTARI, A. (2017). Comparison of Bayesian predictive methods for model selection. *Stat. Comput.* **27** 711–735. MR3613594

POLSON, N. G., SCOTT, J. G. and WINDLE, J. (2013). Bayesian inference for logistic models using Pólya–Gamma latent variables. *J. Amer. Statist. Assoc.* **108** 1339–1349. MR3174712

PUELZ, D., HAHN, P. R. and CARVALHO, C. M. (2017). Variable selection in seemingly unrelated regressions with random predictors. *Bayesian Anal.* **12** 969–989. MR3724975

ROKACH, L. (2010). Ensemble-based classifiers. *Artif. Intell. Rev.* **33** 1–39.

SCHAPIRE, R. E. (1999). A brief introduction to boosting. In *IJCAI* 1401–1406.

SINGH, D., FEBBO, P. G., ROSS, K., JACKSON, D. G., MANOLA, J., LADD, C., TAMAYO, P., RENSHAW, A. A., D'AMICO, A. V., RICHIE, J. P. et al. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell* **1** 203–209.

SLONIM, D. K. (2002). From patterns to pathways: Gene expression data analysis comes of age. *Nat. Genet.* **32 (Supp)** 502.

SMITH, M. and KOHN, R. (1996). Nonparametric regression using Bayesian variable selection. *J. Econometrics* **75** 317–343.

TERENIN, A., DONG, S. and DRAPER, D. (2016). GPU-accelerated Gibbs sampling. Preprint. Available at arXiv:1608.04329.

TIBSHIRANI, R. (1996). Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B* **58** 267–288. MR1379242

VAN'T VEER, L., DAI, H., VAN DE VIJVER, M. J., HE, Y. D., HART, A. A. M., MAO, M., PETERSE, H. L., VAN DER KOOY, K., MARTON, M. J., WITTEVEEN, A. T. et al. (2002). Gene expression profiling predicts clinical outcome of breast cancer. *Nature* **415** 530–536.

WOLPERT, D. H. (1992). Stacked generalization. *Neural Netw.* **5** 241–259.

YAP, Y., ZHANG, X., LING, M. T., WANG, X., WONG, Y. C. and DANCHIN, A. (2004). Classi-
fication between normal and tumor tissues based on the pair-wise gene expression ratio. *BMC Cancer* **4** 72.

ZOU, H. and HASTIE, T. (2005). Regularization and variable selection via the elastic net. *J. R. Stat. Soc. Ser. B. Stat. Methodol.* **67** 301–320. MR2137327

DIVISION OF STATISTICS AND MACHINE LEARNING
DEPARTMENT OF COMPUTER AND INFORMATION SCIENCE
LINKÖPING UNIVERSITY
SE-581 83 LINKÖPING
SWEDEN
E-MAIL: malte.nlz@googlemail.com
            mattias.villani@liu.se
URL: http://mattiasvillani.com