How Should Intelligence Be Abstracted in AI Research ...
AAAI Technical Report FS-13-02

Soft Rule Ensembles for Supervised Learning

Deniz Akdemir, Nicolas Heslot, Jean-Luc Jannink
Bradfield Hall, Department of Plant Breeding & Genetics
Cornell University, Ithaca, NY 14853, USA

Abstract

In this article supervised learning problems are solved using soft rule ensembles. First, we review the importance sampling learning ensembles (ISLE) approach that is useful for generating hard rules. Soft rules are obtained with logistic regression using the corresponding hard rules and training data. Soft rule ensembles work well when both the response and the input variables are continuous because soft rules provide smooth transitions around the boundaries of hard rules. Finally, various examples and simulation results are provided to illustrate and evaluate the performance of soft rule ensembles.

Introduction

A relatively new approach to modeling data, namely the ensemble learning ((Ho, Hull, and Srihari 1990), (Hansen and Salamon 1990), (Kleinberg 1990)) challenges the monist views by providing solutions to complex problems simultaneously from a number of models. By focusing on regularities and stable common behavior, ensemble modeling approaches provide solutions that as a whole outperform the single models. Some influential early works in ensemble learning were by Breiman with Bagging (bootstrap aggregating) ((Breiman 1996)), and Freund and Shapire with AdaBoost ((Freund and Schapire 1996)). All of these methods involve random sampling the 'space of models' to produce an ensemble of models ((Seni and Elder 2010)).

Although not necessary, ensemble models are usually constructed from regression / classification trees or binary rules extracted from them which are discontinuous and piecewise constant. In order to approximate a smooth response variable, a large number of trees or rules with many splits are needed. This causes data fragmentation for high dimensional problems where the data is already sparse. The soft rule ensembles that are proposed in this paper attacks this problem by replacing the hard rules with smooth functions.

In the rest of this section, we review the ensemble model generation and post-processing approach due to Popescu & Friedman ((Friedman and Popescu 2003)). Their approach attempts to unify ensemble learning methods. In the Section 2, we explain how to convert hard rules to soft rules using bias corrected logistic regression. In Section 3, we compare the soft and hard rule ensembles with the aid of examples and simulations. The paper concludes with our comments and discussions.

Suppose we are asked to predict the continuous outcome variable y from p vector of input variables $\boldsymbol{x}$. We restrict the prediction models to the model family $\mathscr{F} = \{f(\boldsymbol{x};\theta) : \theta \in \Theta\}$. The models considered by the ISLE framework have an additive expansion of the form:

$$F(\boldsymbol{x}) = w_0 + \sum_{j=1}^{M} w_j f(\boldsymbol{x}, \theta_j) \qquad (1)$$

where $\{f(\boldsymbol{x}, \theta_j)\}_{j=1}^{M}$ are base learners selected from $\mathscr{F}$. Popescu & Friedman's ISLE approach ((Friedman and Popescu 2003)) uses a heuristic two-step approach to arrive at $F(\boldsymbol{x})$. The first step involves sampling the space of possible models to obtain $\{\widehat{\theta}_j\}_{j=1}^{M}$. The models in the model family $\mathscr{F}$ are sampled using perturbation sampling; by varying case weights, data values, variable subsets, or partitions of the input space ((Seni and Elder 2010)). The second step combines the predictions from these models by choosing weights $\{w_j\}_{j=0}^{M}$ in (1). Let $L(.,.)$ is a loss function, $S_j(\eta)$ is a subset of the indices $\{1, 2, \ldots, n\}$ chosen by a sampling scheme η, $0 \leq \nu \leq 1$ is a memory parameter. The pseudo code to produce M models $\{f(\boldsymbol{x}, \widehat{\theta}_j)\}_{j=1}^{M}$ under ISLE framework is given below:

Algorithm 0.1: ISLE(M, v, η)

```
$F_0(\boldsymbol{x}) = 0.$
for j=1 to M
    do $\begin{cases} (\widehat{c}_j, \widehat{\theta}_j) \\ = \underset{(c,\theta)}{\operatorname{argmin}} \sum_{i \in S_j(\eta)} L(y_i, F_{j-1}(\boldsymbol{x}_i) + cf(\boldsymbol{x}_i, \theta)) \\ T_j(\boldsymbol{x}) = f(\boldsymbol{x}, \widehat{\theta}_j) \\ F_j(\boldsymbol{x}) = F_{j-1}(\boldsymbol{x}) + \nu \widehat{c}_j T_j(\boldsymbol{x}) \end{cases}$
return $(\{T_j(\boldsymbol{x})\}_{j=1}^{M} and F_M(\boldsymbol{x}).)$
```

The classic ensemble methods of Bagging, Random Forest, AdaBoost, and Gradient Boosting are special cases of the generic ensemble generation procedure ((Seni and Elder 2010)). The weights $\{w_j\}_{j=0}^{M}$ can be selected in a number of ways, for Bagging and Random Forests these weights are set to predetermined values, i.e. $w_0 = 0$ and $w_j = \frac{1}{M}$ for $j = 1, 2, \ldots, M$. Boosting calculates these weights in stage wise fashion at each step by having positive memory μ, estimating c_j and takes $F_M(\boldsymbol{x})$ as the final prediction model.

Friedman & Popescu ((Friedman and Popescu 2003)) recommend learning the weights $\{w_j\}_{j=0}^{M}$ using LASSO ((Tibshirani 1996)). Let $T = (T_j(\boldsymbol{x}_i))_{i=1}^{n}{}_{m=1}^{M}$ be the $n \times M$ matrix of predictions for the n observations by the M models in an ensemble. The weights $(w_0, \boldsymbol{w} = \{w_m\}_{m=1}^{M})$ are obtained from

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}(\boldsymbol{y} - w_0\mathbf{1}_n - T\boldsymbol{w})'(\boldsymbol{y} - w_0\mathbf{1}_n - T\boldsymbol{w}) + \lambda \sum_{j=1}^{M} |w_m|. \tag{2}$$

$\lambda > 0$ is the shrinkage operator, larger values of λ decreases the number of models included in the final prediction model. The final ensemble model is given by $F(\boldsymbol{x}) = \hat{w}_0 + \sum_{m=1}^{M} \hat{w}_m T_m(\boldsymbol{x})$.

The base learners, $f(\boldsymbol{x}, \theta)$, in the ISLE framework can be of any kind however usually regression or decision trees are used. Each decision tree in the ensemble partitions the input space using the product of indicator functions of 'simple' regions based on several input variables. A tree with K terminal nodes define a K partition of the input space where the membership to a specific node, say node k, can be accomplished by applying the conjunctive rule $r_k(\boldsymbol{x}) = \prod_{l=1}^{p} I(x_l \in s_{lk})$, where $I(.)$ is the indicator function. The regions s_{lk} are intervals for continuous variables and subsets of the levels for categorical variables. Therefore, a 'simple' rule corresponds to a region that is the intersection of half spaces defined by hyperplanes that are orthogonal to the axis of the predictor variables.

A regression tree with K terminal nodes can be written as

$$T(\boldsymbol{x}) = \sum_{k=1}^{K} \beta_k r_k(\boldsymbol{x}). \tag{3}$$

Trees with many terminal nodes usually produce more complex rules and tree depth is an important meta-parameter which we can control by maximum tree depth and cost pruning.

Given a set of decision trees, rules can be extracted from each of these trees to define a collection of conjunctive rules (Figure). A conjunctive rule $r(\boldsymbol{x}) = \prod_{l=1}^{p} I(x_l \in s_l)$ can also be expressed as a logic rule (also called Boolean expressions and logic statement) involving only the $\wedge$ ('and') operator. In general, a logic statement is constructed using the operators $\wedge$ ('and'), $\vee$ ('or') and ^{c} ('not') and brackets. An example logic rule is $l(\boldsymbol{x}) = [I(x_1 \in s_1) \vee I^c(x_2 \in s_2)] \wedge I(x_3 \in s_3)$. Logic Regression ((Kooperberg and Ruczinski 2005)) is an adaptive regression methodology that constructs logic rules from binary input variables. 'Simple' conjunctive rules that are learned by the the ISLE approach can be used as input variables to logic regression to combine these rules into logic rules. However, the representation of a logic rule in general is not unique and it can be shown that logic rules can be expressed in disjunctive normal form where we only use $\vee$ combinations of $\wedge$ (but not necessarily 'simple') terms.

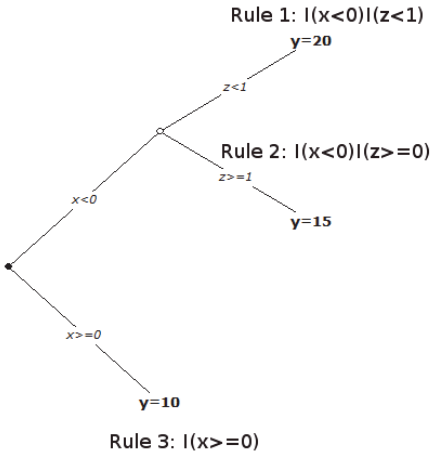

Figure 1: A simple regression tree which can be represented as $y = 20I(x < 0)I(z < 1) + 15I(x < 0)I(z \geq 1) + 10I(x \geq 0)$. Each leaf node defines a rule which can be expressed as a product of indicator functions of half spaces. Each rule specifies a 'simple' rectangular region in the input space.

Let $R = (r_k(\boldsymbol{x}_i))_{i=1}^{n}{}_{l=1}^{L}$ be the $n \times L$ matrix of rules for the n observations by the L rules in the ensemble. The **rulefit** algorithm of Friedman & Popescu (Friedman and Popescu 2008) uses the weights $(w_0, \boldsymbol{w} = \{w_l\}_{l=1}^{L})$ that are estimated from

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}(\boldsymbol{y} - w_0\mathbf{1}_n - R\boldsymbol{w})'(\boldsymbol{y} - w_0\mathbf{1}_n - R\boldsymbol{w}) + \lambda \sum_{l=1}^{L} |w_l| \tag{4}$$

in the final prediction model $F(\boldsymbol{x}) = \hat{w}_0 + \sum_{l=1}^{L} \hat{w}_l r_l(\boldsymbol{x})$.

Rule ensembles are shown to produce improved accuracy over traditional ensemble methods like random forests, bagging, boosting and ISLE ((Friedman and Popescu 2003), (Friedman and Popescu 2008) and (Seni and Elder 2010)).

Soft Rules from Hard Rules

Soft rules which take values in $[0, 1]$ are obtained by replacing each hard rule $r(\boldsymbol{x})$ with a logistic function of the form

$$s(\boldsymbol{x}) = \frac{1}{1 + exp(-g(\boldsymbol{x}; \theta))}.$$

The value of a soft rule $s(\boldsymbol{x})$ can be viewed as the probability that that rule is fired for $\boldsymbol{x}$.

In (Da Rosa, Veiga, and Medeiros 2008) hard rules are replaced with products of univariate logistic functions to build the models called tree-structured smooth transition regression models that generalize the regression tree models. A similar model called soft decision trees are introduced in (Irsoy, Yildiz, and Alpaydin 2012). These authors use logistic functions to calculate gating probabilities at each node in an hierarchical structure where the children of each node are selected with a certain probability, the terminal nodes of the incrementally learned trees are represented as a product of logistic functions. In (Dvořák and Savickỳ 2007) tree splits are softened using simulated annealing. Fuzzy decision trees were presented by (Olaru and Wehenkel 2003). Perhaps, these models can be used to generate soft rules. However, in this paper, we use a simpler approach where we utilize logistic functions only at the terminal nodes of the trees built by the CART algorithm (Breiman et al. 1984).

Each rule defines a 'simple' rectangular region in the input space and therefore, in this paper, $g(\boldsymbol{x};\theta)$ includes additive terms of order two without any interaction terms in the variables which were used explicitly in the construction of the rule $r(\boldsymbol{x})$. The model is built using best subsets regression, selecting the best subset of terms for which the AIC is minimized. The coefficients θ of the function $g(\boldsymbol{x};\theta)$ are to be estimated from the examples of $\boldsymbol{x}$ and $r(\boldsymbol{x})$ in the training data.

A common problem with logistic regression is the problem of (perfect) separation ((Heinze and Schemper 2002)) which occurs when the response variable can be (perfectly) separated by one or a linear combination of a few explanatory variables. When this is the case, the likelihood function becomes monotone and non finite estimates of coefficients are produced. In order to deal with the problem of separation, Firth's bias corrected likelihood approach ((Firth 1993)) has been recommended ((Heinze and Schemper 2002)). The bias corrected likelihood approach to logistic regression are guaranteed to produce finite estimates and standard errors.

Maximum likelihood estimators of the coefficients θ are obtained as the solution to the score equation

$$dlogL(\theta)/d\theta = U(\theta) = 0$$

where $L(\theta)$ is the likelihood function. Firth's bias corrected likelihood uses a modified likelihood function

$$L^*(\theta) = L(\theta)|I(\theta)|^{1/2}$$

where $I(\theta)$ is the Jeffreys ((Jeffreys 1946)) invariant prior, the Fisher information matrix which is given by

$$I(\theta) = -E[\frac{\partial^2 lnL(\theta)}{\partial\theta\partial\theta'}].$$

Using the modified likelihood function the score function for the logistic model is given by $U^*(\theta) = (U^*(\theta_1), U^*(\theta_2), \ldots, U^*(\theta_k))'$ where

$$U^*(\theta_j) = \sum_{i=1}^{n}\{r(\boldsymbol{x}_i) - g(\boldsymbol{x}_i;\theta) + h_i(\frac{1}{2} - g(\boldsymbol{x}_i;\theta))\}\frac{\partial g(\boldsymbol{x}_i;\theta)}{\partial\theta_j}$$

for $j = 1, 2, \ldots, k$ and k is the number of coefficients in $g(\boldsymbol{x};\theta)$. Here, h_i for $i = 1, 2, \ldots, n$ are the ith diagonal elements of the hat matrix

$$H = W^{1/2}X(X'WX)^{-1}X'W^{1/2}$$

and $W = diag\{g(\boldsymbol{x}_i;\theta)(1 - g(\boldsymbol{x}_i;\theta)).\}$ Bias corrected estimates can be obtained in an iterative fashion using

$$\theta^{t+1} = \theta^t + I^{*-1}(\theta^t)U^*(\theta^t)$$

where

$$I^*(\theta) = -E[\frac{\partial^2 lnL^*(\theta)}{\partial\theta\partial\theta'}].$$

Our programs utilize the 'brglm' package in R ((Kosmidis 2008)) that fits binomial-response generalized linear models using the bias-reduction.

In Figure 2, we present simple hard rules and the corresponding soft rules estimated from the training data. It is clear that the soft rules provide a smooth approximation to the hard rules.

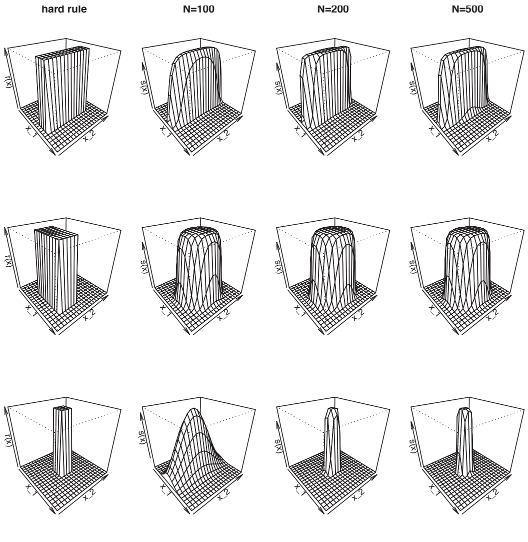

Figure 2: 'Simple' hard rules and the corresponding soft rules estimated from the training data. It is clear that the soft rules provide a smooth approximation to the hard rules. The approximation gets better as the number of examples in the training data set increases.

Let $R = (r_l(\boldsymbol{x}_i))_{i=1}^{n}{}_{l=1}^{L}$ be the $n \times L$ matrix of L rules for the n observations in the training sample. Letting $s_l(\boldsymbol{x};\hat{\theta}_l)$ be the soft rule corresponding to the lth hard rule, define $S = (s_l(\boldsymbol{x}_i))_{i=1}^{n}{}_{l=1}^{L}$ as the $n \times L$ matrix of L soft rules for the n observations.

The weights for the soft rules can be estimated from the LASSO:

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}(\boldsymbol{y} - w_0\mathbf{1}_n - S\boldsymbol{w})'(\boldsymbol{y} - w_0\mathbf{1}_n - S\boldsymbol{w}) + \lambda\sum_{l=1}^{L}|w_l|. \quad (5)$$

This leads to the final soft rule ensemble prediction model $F(\boldsymbol{x}) = \hat{w}_0 + \sum_{l=1}^{L}\hat{w}_l s_l(\boldsymbol{x})$.

The only additional step for building our soft rules ensembles is the hard to soft rule conversion step. This makes our algorithm slower than the the rulefit algorithm ((Friedman and Popescu 2003)), at times, 10 times slower. However, we have implemented our soft rules algorithm in the R language and successfully applied it to several high dimensional problems. We expect that a faster implementation is possible if the code is migrated to a faster programming language. In addition parts or the whole of the soft rule generation process can be accomplished by parallel processing.

For completeness and easy reference, we summarize the steps for soft rule ensemble generation and fitting:

1. Use ISLE algorithm to generate M trees: T(X).
2. Extract hard rules from T(X): R(X).
3. Convert hard rules to soft rules: S(X).
4. Obtain soft rule weights by LASSO.

We should note that no additional meta-parameters are introduced to produce soft rules from hard rules. We set these meta-parameters along the lines of the recommendations in previous work ((Friedman and Popescu 2003), (Friedman and Popescu 2008) and (Seni and Elder 2010)).

There are several fast algorithms that can accomplish the LASSO post processing for large datasets (large n or p): Recent pathwise coordinate descent ((Friedman et al. 2007)) (implemented in 'glmnet' with R ((Friedman, Hastie, and Tibshirani 2009))) algorithm provide the entire path of solutions. We have used the 'glmnet' in our illustrations in the next section, the value of the sparsity parameter was set by minimizing the mean cross-validated error. Due to the well known selection property of the lasso penalty, only 10%-20% of the rules are retained in the final model after the post-processing step.

Illustrations

In this section we are going to compare the soft rule and hard rule ensembles. The prediction accuracy is taken as the cross validated correlation or the mean square error (MSE) calculated for the predicted and true target variable values for regression examples and cross validated area under the ROC curve for classification examples.

Unless otherwise stated, the number of trees to be generated by the ISLE algorithm was set to $M = 400$, each tree used 30% of of the individuals and 10% of the input variables randomly selected from the training set. Larger trees allow more complex rules to be produced and therefore controlling tree depth controls the maximum rule complexity. A choice of this parameter can be based on prior knowledge or based on experiments with different values. We have tried differing three depths for obtaining models with varying complexity. For most of the examples accuracies were reported for each tree depth. In example 0.2, models with differing rule depths were trained and only the models with best cross validated performances were reported. In addition, the memory parameter ν of the ISLE ensemble generation algorithm is set to zero in all the following examples.

Example 0.1. *(Boston Housing Data, Regression) In order to compare the performance of prediction models based on hard and soft rules we use the famous benchmark 'Boston Housing' data set ((Harrison, Rubinfeld, and others 1978)). This data set includes n=506 observations and p=14 variables. The response variable is the median house value from the rest of the 13 variables in the data set. 10 fold cross validated accuracies are displayed in Table 1.*

	Correlation		RMSE	
Depth	Hard	Soft	Hard	Soft
2	0.91	0.92	3.78	3.60
3	0.93	0.93	3.40	3.42
4	0.94	0.93	3.18	3.29
5	0.93	0.94	3.33	3.18

Table 1: The 10-fold cross validated prediction accuracies as measured by the correlation of the true and predicted values are given for the 'Boston housing data'.

For problems with only categorical or discrete input variables, we do not expect to see the same improvements. The following example only uses discrete SNP markers (biallelic markers values coded as -1,0 and 1) as input variables and hard rules and soft rules give approximately the same accuracies.

Example 0.2. *(Plant Breeding Data, Regression) In our second example we analyze plant breeding data and compare the predictive performance of hard rules with soft rules. In both cases, the objective is to predict a quantitative trait (observed performance) using molecular markers data providing information about the genotypes of the plants. Predictions of phenotypes using numerous molecular markers at the same time is called genomic selection and has received lately a lot of attention in the plant and animal breeding communities. Rule ensembles used with genetic marker data implicitly captures epistasis (interaction between markers) in a highly dimensional context while retaining interpretability of the model.*

The first data set (Bay x Sha) contains measurements on flowering time under short day length (FLOSD), dry matter under non limiting (DM10) or limiting conditions (DM3) from 422 recombinant inbred lines from a biparental population of Arabidopsis thaliana plants from 2 ecotypes, Bay-0 (Bay) and Shadara (Sha) genotyped with 69 SSRs. Data available from the Study of the Natural Variation of A. thaliana website ((Heslot et al. 2012), (Lorenzana and Bernardo 2009)).

The second data set (Wheat CIMMYT) is composed of 599 spring wheat inbred lines evaluated for yield in 4 different target environments (YLD1-YLD4). 1279 DArT markers were available for the 599 lines in the study ((Crossa et al. 2010)). The results are displayed in Table 2.

When the task is classification, no significant improvement is achieved by preferring soft rules over hard rules. To compare soft and hard rule ensembles in the context of classification, we use the Arcene and Madelon datasets downloaded from UCI Machine Learning Repository.

Example 0.3. *(Arcene data, Classification) The task in arcene data is to classify patterns as normal or cancer based on mass-spectrometric data. There were 7000 initial input variables, 3000 probe input variables were added to increase the difficulty of the problem. There were 200 individuals in the data set. Areas under the ROC curves for soft*

Table 2: Accuracies of hard and soft rule ensembles are compared by the cross validated Pearson's correlation coefficients between the estimated and true values. For each data set we have trained models based on rules of depth 1 to 3, the results from the model with best cross validated accuracies is reported. The results show no significant difference between hard or soft rules.

CIMMYT						Bay-Sha					
		Accuracy		RMSE				Accuracy		RMSE	
Depth		Hard	Soft	Hard	Soft	Depth		Hard	Soft	Hard	Soft
2	YLD1	0.50	0.51	0.87	0.86	2	FLOSD	0.86	0.86	4.66	4.64
	YLD2	0.41	0.41	0.92	0.92		DM10	0.67	0.67	2.17	2.18
	YLD3	0.36	0.36	0.96	0.96		DM3	0.37	0.37	1.18	1.18
	YLD4	0.42	0.42	0.92	0.92	3	FLOSD	0.86	0.85	4.75	4.77
3	YLD1	0.52	0.52	0.86	0.86		DM10	0.62	0.62	2.30	2.30
	YLD2	0.42	0.43	0.92	0.91		DM3	0.33	0.33	1.22	1.22
	YLD3	0.40	0.40	0.93	0.93						
	YLD4	0.40	0.41	0.94	0.93						

and hard rule ensemble models based on rules with depths 2 to 4 are compared in Table 3 (left).

Example 0.4. *(Madelon data, Classification) Madelon is contains data points grouped in 32 clusters placed on the vertices of a five dimensional hypercube and randomly labeled +1 or -1. There were 500 continuous input variables, 480 of these were probes. There were 2600 labeled examples. Areas under the ROC curves for rules with depths 2 to 4 are compared in Table 3 (right).*

Arcene			Madelon		
Depth	Hard	Soft	Depth	Hard	Soft
2	0.82	0.82	2	0.83	0.87
3	0.87	0.82	3	0.86	0.88
4	0.79	0.80	4	0.90	0.92

Table 3: 10- fold cross validated areas under the ROC curves for soft and hard rule ensemble models based on rules with depths 2 to 4 for the Arsene (left) and Madelon (right) data sets. We do not observe any significant difference between soft and hard rules.

When both response and input variables are continuous, soft rules perform better than hard rules. In the last two examples, we compare our models via mean squared errors.

Example 0.5. *(Simulated Data, Regression) This regression problem is described in Friedman ((Friedman 1991)) and Breiman ((Breiman 1996)). Elements of the input vector* $\boldsymbol{x} = (x_1, x_2, \ldots, x_{10})$ *are generated from* $uniform(0, 1)$ *distribution independently, only 5 out of these 10 are actually used to calculate the target variable* y *as*

$$y = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + e$$

where $e \sim N(0, 1)$. *1000 independent realizations of* $(\boldsymbol{x}, y)$ *constitute the training data. Mean squared errors for models are calculated on a test sample of the same size. The boxplots in left Figure 3 summarize the prediction accuracies for soft and hard rules over 30 replications of the experiment.*

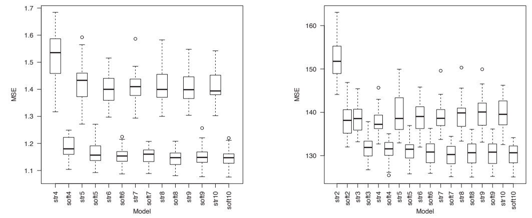

Figure 3: The boxplots summarize the prediction accuracies for soft and hard rules over 30 replications of the experiments described in Examples 3.5 and 3.6. In terms of mean squared errors the soft rule ensembles perform better than the corresponding hard rule ensembles for all values of tree depth. The hard rule ensemble models are denoted by 'str' and soft rule ensemble models are denoted by 'soft'. The numbers next to these acronyms is the depth of the corresponding hard rules.

Example 0.6. *(Simulated Data, Regression) Another problem described in Friedman ((Friedman 1991)) and Breiman ((Breiman 1996)). Inputs are 4 independent variables uniformly distributed over the ranges* $0 \leq x_1 \leq 100$, $40\pi \leq x_2 \leq 560\pi$, $0 \leq x_3 \leq 1$, $1 \leq x_4 \leq 11$. *The outputs are created according to the formula* $y = (x_1^2 + (x_2 x_3 - (1/(x_2 x_4))^2)^{0.5} + e$ *where* e *is* $N(0, sd = 125)$. *1000 independent realizations of* $(\boldsymbol{x}, y)$ *constitute the training data. Mean squared errors for models are calculated on a test sample of the same size. The boxplots in right Figure 3 summarize the prediction accuracies for soft and hard rules.*

Discussions

As our examples in the previous section suggest, the best case for soft rules is when both input and output variables are continuous. For data sets with mixed input variables it might be better to use two sets of rules (hard rules for categorical variables and soft rules for numerical variables) and combine in a supervised learning model with group lasso model ((Yuan and Lin 2005)) or perhaps these rules can be combined in a multiple kernel model ((Bach, Lanckriet, and Jordan 2004), (Gönen and Alpaydın 2011)).

The hard rules or the soft rules can be used as input variables in any supervised or unsupervised learning problem. In (Akdemir 2011), several promising hard rule ensemble methods were proposed for supervised, semi-supervised and unsupervised learning. For instance, the model weights can be obtained using partial least squares regression. A similarity matrix obtained from hard rules can be used as a learned kernel matrix in Gaussian process regression. It is straightforward to use these and similar methods with soft rules.

Several model interpretation tools have been developed to use with rule ensembles and the ISLE models. These include local and global rule, variable and interaction importance measures and partial dependence functions ((Friedman and Popescu 2008)). We can use the same tools to interpret the soft rule ensemble model. For example, the absolute values of the standardized coefficients can be used to evaluate the importance of rules. A measure of importance for each input variable can be obtained as the sum the importances of rules that involve that variable. The interaction importance measures and the partial dependence functions described in (Friedman and Popescu 2008) are general measures and they also naturally apply to our soft rule ensembles.

The ensemble approaches are also a remedy for memory problems faced in analyzing big data sets. By adjusting the sampling scheme in the ISLE algorithm we were able to analyze large data sets which have thousands of variables and tens of thousands of individuals by only loading fractions of the data into the memory at a time.

Acknowledgments

This research was supported by the USDA-NIFA-AFRI Triticeae Coordinated Agricultural Project, award number 2011-68002-30029.

References

Akdemir, D. 2011. Ensemble learning with trees and rules: Supervised, semi-supervised, unsupervised. *Arxiv preprint arXiv:1112.3699*.

Bach, F.; Lanckriet, G.; and Jordan, M. 2004. Multiple kernel learning. *Conic Duality, and the SMO Algorithm*.

Breiman, L.; Friedman, J.; Stone, C. J.; and Olshen, R. A. 1984. *Classification and regression trees*. Chapman & Hall/CRC.

Breiman, L. 1996. Bagging predictors. *Machine learning* 24(2):123–140.

Crossa, J.; de los Campos, G.; Pérez, P.; Gianola, D.; Burgueño, J.; Araus, J.; Makumbi, D.; Singh, R.; Dreisigacker, S.; Yan, J.; et al. 2010. Prediction of genetic values of quantitative traits in plant breeding using pedigree and molecular markers. *Genetics* 186(2):713–724.

Da Rosa, J. C.; Veiga, A.; and Medeiros, M. C. 2008. Tree-structured smooth transition regression models. *Computational Statistics & Data Analysis* 52(5):2469–2488.

Dvořák, J., and Savickỳ, P. 2007. Softening splits in decision trees using simulated annealing. *Adaptive and Natural Computing Algorithms* 721–729.

Firth, D. 1993. Bias reduction of maximum likelihood estimates. *Biometrika* 80(1):27–38.

Freund, Y., and Schapire, R. 1996. Experiments with a new boosting algorithm. In *Machine Learning-International Workshop then Conference-*, 148–156. Morgan Kaufmann Publishers, Inc.

Friedman, J., and Popescu, B. 2003. Importance sampled learning ensembles. *Journal of Machine Learning Research* 94305.

Friedman, J., and Popescu, B. 2008. Predictive learning via rule ensembles. *The Annals of Applied Statistics* 2(3):916–954.

Friedman, J.; Hastie, T.; Höfling, H.; and Tibshirani, R. 2007. Pathwise coordinate optimization. *The Annals of Applied Statistics* 1(2):302–332.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2009. glmnet: Lasso and elastic-net regularized generalized linear models. *R package version* 1.

Friedman, J. 1991. Multivariate adaptive regression splines. *The annals of statistics* 1–67.

Gönen, M., and Alpaydın, E. 2011. Multiple kernel learning algorithms. *Journal of Machine Learning Research* 12:2211–2268.

Hansen, L., and Salamon, P. 1990. Neural network ensembles. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 12(10):993–1001.

Harrison, D.; Rubinfeld, D.; et al. 1978. Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management* 5(1):81–102.

Heinze, G., and Schemper, M. 2002. A solution to the problem of separation in logistic regression. *Statistics in medicine* 21(16):2409–2419.

Heslot, N.; Sorrells, M.; Jannink, J.; and Yang, H. 2012. Genomic selection in plant breeding: A comparison of models. *Crop Science* 52(1):146–160.

Ho, T.; Hull, J.; and Srihari, S. 1990. Combination of structural classifiers.

Irsoy, O.; Yildiz, O. T.; and Alpaydin, E. 2012. Soft decision trees. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, 1819–1822. IEEE.

Jeffreys, H. 1946. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* 186(1007):453–461.

Kleinberg, E. 1990. Stochastic discrimination. *Annals of Mathematics and Artificial intelligence* 1(1):207–239.

Kooperberg, C., and Ruczinski, I. 2005. Identifying interacting snps using monte carlo logic regression. *Genetic epidemiology* 28(2):157–170.

Kosmidis, I. 2008. brglm: Bias reduction in binary-response glms. *R package version 0.5-4, URL http://CRAN. R-project. org/package= brglm*.

Lorenzana, R., and Bernardo, R. 2009. Accuracy of genotypic value predictions for marker-based selection in biparental plant populations. *TAG Theoretical and Applied Genetics* 120(1):151–161.

Olaru, C., and Wehenkel, L. 2003. A complete fuzzy decision tree technique. *Fuzzy sets and systems* 138(2):221–254.

Seni, G., and Elder, J. 2010. Ensemble methods in data mining: improving accuracy through combining predictions. *Synthesis Lectures on Data Mining and Knowledge Discovery* 2(1):1–126.

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288.

Yuan, M., and Lin, Y. 2005. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68(1):49–67.