Boosting: An Ensemble Learning Tool for Compound Classification and QSAR Modeling

Vladimir Svetnik,*,† Ting Wang,† Christopher Tong,† Andy Liaw,† Robert P. Sheridan,‡ and Qinghua Song§

Biometrics Research and Molecular Systems, Merck Research Laboratories, P.O. Box 2000, Rahway, New Jersey 07065, and Department of Statistics, University of Wisconsin−Madison, Madison, Wisconsin 53706

Received January 31, 2005

A classification and regression tool, J. H. Friedman's Stochastic Gradient Boosting (SGB), is applied to predicting a compound's quantitative or categorical biological activity based on a quantitative description of the compound's molecular structure. Stochastic Gradient Boosting is a procedure for building a sequence of models, for instance regression trees (as in this paper), whose outputs are combined to form a predicted quantity, either an estimate of the biological activity, or a class label to which a molecule belongs. In particular, the SGB procedure builds a model in a stage-wise manner by fitting each tree to the gradient of a loss function: e.g., squared error for regression and binomial log-likelihood for classification. The values of the gradient are computed for each sample in the training set, but only a random sample of these gradients is used at each stage. (Friedman showed that the well-known boosting algorithm, AdaBoost of Freund and Schapire, could be considered as a particular case of SGB.) The SGB method is used to analyze 10 cheminformatics data sets, most of which are publicly available. The results show that SGB's performance is comparable to that of Random Forest, another ensemble learning method, and are generally competitive with or superior to those of other QSAR methods. The use of SGB's variable importance with partial dependence plots for model interpretation is also illustrated.

1. INTRODUCTION

In the past few years, a number of statistical, machine learning, and data mining methods have been introduced into cheminformatics for the in silico prediction of compounds' chemical or biological activities, as determined experimentally, using a quantitative description of the compounds' molecular structure and/or properties. In this paper, we shall use the term "QSAR," or quantitative structure−activity relationship modeling, to refer to predicting both quantitative and categorical activities (regression and classification, respectively). For example, recently these methods have been used extensively for QSAR modeling of ADMET (absorption, distribution, metabolism, excretion, and toxicity) properties.[1−3] Decision tree analysis[4,5] (also known as recursive partitioning) is a particularly attractive QSAR approach, since it has a good combination of properties, such as the ability to handle a large number of diverse descriptors, the ability to ignore irrelevant descriptors, the ability to model nonlinearity and multiple mechanisms of action, and computational efficiency. Unfortunately, decision trees often suffer from low prediction accuracy, and many attempts have been made to improve their performance.

Recently, it has been shown that *ensembles* of trees provide much higher prediction accuracy than that of a single tree. Procedures for building tree ensembles are based on the concept of ensemble learning,[6] in which combining multiple learners gives an improved performance over that of an individual learner. In statistics and machine learning, the most prominent tree ensemble approaches are Bagging,[7] Boosting,[8,9] and Random Forest.[10] (These ensemble methods may also be applied to base learners other than trees.) In Bagging, multiple trees are grown independently on bootstrap samples of the training data, and predictions are made by aggregating the outputs of all the trees (in classification, the trees vote; in regression, their outputs are averaged). Random Forest is like Bagging except that in the tree growing procedure, at each node only a random sample of the descriptors is considered for splitting, rather than all the descriptors. Boosting is an iterative reweighting procedure where the current tree is built on weighted data samples whose current weights are modified based on how accurately the previous learners predict these samples. Friedman[11] generalized this procedure by interpreting Boosting as a forward stage-wise additive model-building and showed its relation with gradient optimization. (We will discuss Boosting in greater detail in section 2.) So far, empirical evidence suggests that Boosting and Random Forest usually outperform Bagging and are quite similar to each other in performance.[12] Another advantage of Boosting and Random Forest is that they provide a measure of variable importance, and this measure can be used in conjunction with partial dependence plots[11] for interpretation.

A number of tree ensemble approaches, which are referred to as "forests", have been applied to QSAR modeling. An approach similar to Bagging, except that the bootstrap training sets are constrained to have equal numbers of samples from each class, has been applied to a number of

* Corresponding author phone: (732)594-5544; fax: (732)594-1565; e-mail: vladimir_svetnik@merck.com.
† Biometrics Research, Merck Research Laboratories.
‡ Molecular Systems, Merck Research Laboratories.
§ University of Wisconsin−Madison.

10.1021/ci0500379 CCC: $30.25
Published on Web 04/20/2005

QSAR problems.[13−15] Random Forest has been examined at length in our previous reports.[16,17] One of the most popular versions of Boosting, AdaBoost, has been applied to both decision trees and neural networks for classification by He et al.[18] In their examples, they showed that Boosting allows a higher prediction accuracy compared to that of the base classifier. Aside from the forest methods discussed earlier, a number of others have appeared in the cheminformatics literature. In Random FIRM,[19] there is an ensemble of trees, each built on all the training data, but at each split, a variable is randomly selected according to probabilities related to the variable's significance from an appropriate statistical test defining the split. Although Random FIRM could certainly be used for prediction, using one of the tree aggregation procedures, it is used primarily for model interpretation. Decision Forest[20] is an ensemble built such that each tree uses variables not available to the other trees, but such that the prediction accuracy of each tree does not fall below a specified threshold. (This method is not to be confused with an earlier forest method of the same name.[21]) Recursion forest[22] is an ensemble of trees all grown on the same training data, but with the tree growing parameters systematically varied to produce different trees; a consensus selection procedure is then used to compute a Boolean intersection of the resulting outputs. Finally, there is current work underway on building ensembles of trees grown using evolutionary programming rather than recursive partitioning.[23]

As noted earlier, Boosting is one of the most prominent ensemble learning procedures in the machine learning community, yet we know of only one published application of Boosting to a QSAR problem.[18] Moreover, there are actually several implementations of Boosting, such as Freund and Schapire's AdaBoost,[8] Friedman's Stochastic Gradient Boosting (SGB),[24] and many others.[25−27] The earlier study of Boosting for QSAR was based on AdaBoost.[18] In this paper, our goal is to evaluate Friedman's Stochastic Gradient Boosting as a QSAR procedure, on 10 data sets, both small and large. We will compare the performance of SGB on these data sets with a number of commonly used QSAR methods: a single decision tree (DT), Random Forest (RF), *k*-nearest neighbors (KNN), partial least squares (PLS), naive Bayes (NB), and support vector machines (SVM) with both linear and radial basis function (RBF) kernels. We will also examine the details of how to use SGB's variable importance measure for the descriptors, and partial dependence functions to interpret important descriptors. Variable importance and partial dependence functions were also used to aid in the interpretation of Boosting results in QSAR by He et al.[18]

We keep in mind Wolpert's No Free Lunch theorem[28] that there is no one best algorithm for all problems. Theoretically speaking, for a given set of molecules, one can investigate what combination of descriptors and QSAR methods would give the best performance. However this approach would be almost impossible in practical applications. The number of such combinations is extremely large, and tuning very diverse methods would require extraordinary expertise and deep knowledge of each of these methods. An alternative approach would be to use a small number of methods without parameter tuning, or with relatively easy parameter tuning, and "preferred" descriptors and be confident that the obtained results are close to the results obtained with the best combination of method and descriptors. Although we understand that our analysis in this paper is a limited one, this analysis and our experience at Merck indicate that ensembles of trees are these "safe" types of methods which can be used with a variety of descriptors. This observation, which relates to the performance of methods, as well as additional features of these methods, such as descriptor importance, make them especially appealing for QSAR modeling.

Table 1. List of Major Abbreviations

ADMET	absorption, distribution, metabolism, excretion, and toxicity
BBB	blood-brain barrier
CART	classification and regression trees
Cdk2	cyclin-dependent kinase 2
CNS	central nervous system
COX-2	cyclooxygenase-2
CV	cross-validation
DT	decision tree
ER	error rate
GBM	generalized boosted regression modeling
KNN	*k*-nearest neighbors
MART	multiple additive regression trees
MDRR	multidrug resistance reversal
MSE	mean square error
NB	naive Bayes
OOB	out-of-bag
P-gp	P-glycoprotein
PLS	partial least squares
QSAR	quantitative structure−activity relationship
RBF	radial basis function
RF	random forest
RMSE	root mean square error
SAR	structure−activity relationship
SE	standard error
SGB	stochastic gradient boosting
SVM	support vector machines
TN	true negative
TP	true positive

This paper is organized as follows. In section 2, we describe in detail the Stochastic Gradient Boosting algorithm and its features. In section 3, we describe 10 data sets which we used to evaluate SGB prediction accuracy in classification and regression; the results of the evaluation are presented in section 4. Section 5 illustrates the use of SGB's descriptor importance measure and partial dependence functions. Finally, section 6 is a concluding discussion. In Appendix A we use the bias-variance decomposition to gain further insight into SGB. In Appendix B we introduce the DRUGBITS descriptors that are used in section 5. A list of major abbreviations used in this paper is given in Table 1.

2. STOCHASTIC GRADIENT BOOSTING

Ensemble of Trees. Stochastic Gradient Boosting (SGB)[24] is a general numerical procedure for building models sequentially. In this section, we describe how this procedure is used to create ensembles of classification and regression trees, $\{T_1,..., T_M\}$, and an additive function, $F_M(T_1,..., T_M)$, combining these trees' outputs. Below we will follow Friedman's treatment.[24] (The data analysis presented in this paper was done using Ridgeway's R-language software GBM (Generalized Boosted Regression Modeling),[29] freely available from CRAN.[30] This software implements Friedman's SGB, but the details of implementation are slightly different. For details, see the user's manual.[29]) Assume that $X = \{x_1,..., x_p\}$ is a *p*-dimensional vector of molecular descriptors associated with a molecule and $Y(X)$ is this molecule's biological activity (in the regression case) or the label of a

class to which the molecule belongs (in the classification case). Then, in the regression case the estimate $\hat{Y}(X)$ of $Y(X)$ is

$$\hat{Y}(X) = F_M(X) = F_M(T_1(X),..., T_M(X))$$

In classification, we assume a two-class case, $Y(X) = \{-1, +1\}$ (the extension to the multiclass case is also available). Then the function $F_M(T_1,..., T_M)$ is the estimate of half of the log-odds ratio

$$\frac{1}{2}\log\frac{\Pr(Y=+1|X)}{\Pr(Y=-1|X)}$$

where $\Pr(Y=+1|X)$ and $\Pr(Y=-1|X)$ are the probabilities of $Y(X)=+1, -1$, given X, respectively. The estimate $\hat{Y}(X)$ of the class label $Y(X)$ is calculated as

$$\hat{Y}(X) = \text{sign}(F_M(X))$$

and where the estimates of $\Pr(Y=+1|X)$ and $\Pr(Y=-1|X)$ are

$$\hat{\Pr}(Y=+1|X) = \frac{1}{(1+\exp(-2F_M(X)))}$$

$$\hat{\Pr}(Y=-1|X) = 1 - \hat{\Pr}(Y=+1|X)$$

respectively.

Training Procedure. Given data on a set of N molecules for training, $D = \{(X_1, Y_1),...,(X_N, Y_N)\}$, the SGB training algorithm proceeds as follows to build a sequence of M trees (and $M+1$ output functions F_m, $m = 0,..., M$):

Regression

1. START: $F_0 =$ **mean of** $\{Y_1,..., Y_N\}$
2. FOR $m = 1$ TO M DO:
3. FOR $n = 1$ TO N DO:
4. $r_n = Y_n - F_{m-1}(X_n)$
5. END n
6. $\{n_1,..., n_s\} =$ **random_subsample_from**$\{1,..., N\}$
7. Fit a regression tree, T_m, to the data, $\{(r_{n1}, X_{n1}),...,(r_{ns}, X_{ns})\}$.
8. $F_m(T_1(X),..., T_m(X)) = F_{m-1}(T_1(X),..., T_{m-1}(X)) + \nu T_m(X)$
9. END m

Classification

1. START: $F_0 = 0$
2. FOR $m = 1$ TO M DO:
3. FOR $n = 1$ TO N DO:
4. $r_n = 2Y_n/(1 + \exp(2Y_nF_{m-1}(X_n)))$
5. END n
6. $\{n_1,..., n_s\} =$ **random_subsample_from**$\{1,..., N\}$
7. Fit a regression tree, T_m, to the data, $\{(r_{n_1}, X_{n_1}),...,(r_{ns}, X_{ns})\}$.
8. $F_m(T_1(X),..., T_m(X)) = F_{m-1}(T_1(X),..., T_{m-1}(X)) + \nu T_m(X)$
9. END m

There are two constant parameters, s and ν. The parameter s, $1 \ll s \leq N$, is the size of the random subset of values of $\{r_n\}$ used to build each tree. We will sometimes refer instead to the parameter $f = s/N$, the fraction of training data used by each tree. The parameter ν, $0 < \nu \leq 1$, is the so-called shrinkage parameter introduced to mitigate overfitting.[9]

These algorithms have three distinctive features which explain the name "Stochastic Gradient Boosting". We will start with the gradient part. The values r_n in the fourth lines of the algorithms are gradients of the loss function $L(Y, F(X))$:

$$r_n = \frac{\partial}{\partial F(X_n)} L(y_n, F(X_n)|_{F=F_{m-1}(X_n)})$$

The regression algorithm corresponds to the least squares loss function $L = (Y - F(X))^2$, and the classification algorithm uses the loss function $L = \log(1 + \exp(-2YF(X)))$, which is called the binomial log-likelihood function. Friedman showed that building additive models can be done in a stage-wise manner in such a way that at each stage a new tree is built by fitting the gradients.[9] This is a general approach which allows building additive models of any type, e.g., a weighted sum of neural networks,[31] and/or using various loss functions.[24]

The stochastic part of the algorithm's name is due to the fact that at each iteration (stage) m, the tree is fitted to a randomly selected subset of size s of the gradients r_{ni}, $i = 1,..., s$. This random sampling plays the role of data perturbation similar to that in Bagging and Random Forest. As in those cases the idea is to increase diversity among the trees in the ensemble.[24]

The Boosting part of the name is due to the fact that when (a) the loss function in classification is $L = \exp(-YF(X))$, (b) no random sampling is used, and (c) the shrinkage coefficient is unity, $\nu = 1$, the SGB produces the same ensemble classifier as Freund and Shapire's Adaboost.[8] Because AdaBoost can be thought of as a special case of SGB, the above SGB classification algorithm (without random sampling and $\nu = 1$) is often called LogitBoost.

Lines 7 of both algorithms represent the tree building procedure. Unlike Random Forest and Bagging where trees are grown to the maximum length, the SGB trees are usually smaller with the size of the tree being an adjustable parameter. No pruning is used. We discuss the choice of the tree size in the next subsection. In both algorithms the regression tree building procedures are used.

The difference between regression and classification is in the outputs of the terminal nodes once the tree is built. Suppose R_l, $l = 1,..., L$ is one of the terminal nodes. Then $T_m(X \in R_l) = \beta_m(R_l)$ where for regression

$$\beta_m(R_l) = \text{mean of } (r_{n_i} I(X_{n_i} \in R_l))$$

and for classification

$$\beta_m(R_l) = \frac{\text{mean of } (r_{n_i} I(X_{n_i} \in R_l))}{\text{mean of}(|Y_{n_i}|*(2 - |Y_{n_i}|) I(X_{n_i} \in R_l))}$$

Here $I(.)$ is the indicator function. In Appendix A we use the bias-variance decomposition to gain further insight into SGB.

Algorithm Parameter Optimization. At least two different algorithms have been proposed in the SGB literature for building trees with an upper limit on their size.[24,29] The data analysis presented in this paper was done using Ridgeway's R-language software GBM (Generalized Boosted Regression Modeling),[29] freely available from CRAN.[30] In this software Ridgeway uses the number of splits as a

Table 2. Summary of Data Sets Used for Performance Comparison

name	QSAR for	no. of samples	no. of descriptors	property[a] or quantity[b] to predict	no. of active/ inactive[a] or range of activity[b]	descriptor type
BBB	CNS permeability[38] (blood-brain barrier)	325	9	active/inactive	180/145	chemical and physical properties
estrogen	estrogen receptor binding[20]	232	197	binding/nonbinding	131/101	Cerius-2
P-gp	P-glycoprotein transport activity[39]	186	1522	nonsubstrate/substrate	78/108	binarized atom pairs
MDRR	multidrug resistance reversal activity[40]	528	342	active/inactive	298/230	DRAGON
COX-2 (classification)	COX-2 inhibition[47]	314	135	active/inactive	153/161	topological
COX-2 (regression)	COX-2 inhibition[47]	272	135	log(IC50)	0.23−5.00	topological
dopamine	dopamine receptor binding affinity[44]	116	374	−log(IC50)	4.60−8.20	binarized atom pairs
LogD	shake flask octanol/water partition at pH 7.4	11260	6303	log(*D*)	−4.34−6.33	atom pairs
Chan2	bindings to channel protein	23102	6918	−log(IC50)	−0.71−11.23	atom pairs
CDK2	cyclin-dependent kinase-2 activity[43]	15440	6926	active/inactive	361/15079	atom pairs

[a] For classification. [b] For regression.

Table 3. Parameter Settings for Tuning Procedure[a]

method	software	tuning parameter(s) search region	tuning procedure	nontunable parameters other than default
SGB	R-package: gbm	large: n·trees = 20 to 1000 by 20 small: n·trees = 10 to 500 by 1	large: 50/50 on training set small: 5-fold CV on training set	depth = 6 shrinkage = 0.1 weighted sample[b] (for Cdk2 only) sample fraction = 0.5 for large set and = 0.9 for small set
RF	R-package:	none	none	sampsize = # of actives (for Cdk2 only)
DT	R-package: rpart	none	none	weighted sample[b] (for Cdk2 only)
PLS	R-package: pls·pcr	number of PLS = 1 to 50 or less	factors 10-fold CV on training data (default)	none
KNN	R-package: class R-function: dice	large: k = 1 to 29 by 2 small: k = 1 to 21 or less by 2	large: 50/50 on training set small: 5-fold CV on training set	dice similarity for atom pair descriptors; Euclidean otherwise
lin_SVM	libSVM 2.6	large: cost = 2^{-6}, 2^{-4}, 2^{-2}, 20 small: cost = 2^{-5}, 2^{-3}, ..., 215	large: 50/50 on training set small: 5-fold CV on training set	classification: C-SVM regression: ϵ-SVM, $\epsilon = 0.1$ weighted class 10:1 (active:inactive
rbf_SVM	libSVM 2.6 python easy·py	large: cost = 2^{-6}, 2^{-4}, 2^{-2}, 20 gamma = 2^{-11}, 2^{-9}, 2^{-7}, 2^{-5} small: chosen by python easy·py	large: 50/50 on training set small: 5-fold CV on training set	classification: C-SVM regression: ϵ-SVM, $\epsilon = 0.1$ weighted class 10:1 (active:inactive)
NB	R-package: e1071	none	none	none

[a] "large" and "small" refer to the settings for large and small data sets, respectively. "50/50" refers to a split of the training data into subtraining and subtesting parts of equal size. Parameter names correspond to those in the respective software packages. [b] Weighted sample use 1/(# of actives) and 1/(# of inactives) for the active and inactive molecules respectively.

parameter limiting the tree size. He calls this parameter the *interaction depth*. For brevity we will refer to it as "depth". The tree growing process first does one split (depth = 1), thus partitioning the data into 2 nodes. Then an attempt is made to partition each of these nodes into 2 nodes. The process continues until either the number of splits ("depth") reaches *L*, or it stops earlier when no other splits can be made according to the node splitting criterion.[4]

The second GBM parameter controlling the tree size in this package is the minimum number of samples in a node. Any node with less than this number of samples is not split further. Hence, the larger the minimum node size, the smaller the tree.

In principle, using the training data one can optimize all of the SGB parameters: number of trees; shrinkage parameter; fraction of training samples used to build each tree; interaction depth; and node size minimum. However this optimization via, e.g., a multiparameter grid search, usually requires a prohibitively long time. Moreover, when the training data are relatively small, optimization may lead to overfitting due to excessive use of the data.

A large volume of literature suggests that the number of trees is one of the most important parameters in Boosting procedures. When the number of trees continues to increase beyond a data specific value, the test error starts to increase, indicating overfitting. To avoid it, the optimal number of trees has to be determined for each data set. Our experience suggests that SGB performance is much less sensitive to the other parameters. Based on this observation we did not optimize them. Instead, we kept them fixed at the values given in Table 3.

Descriptor Importance. Decision Tree is known for its ability to select "important" descriptors among many and ignore (often irrelevant) others. In addition, Decision Tree gives a readily interpretable model, describing the relationship between these descriptors and the predictions. Ensembles of trees inherit the ability to select "important" descriptors. However, the interpretability of the ensemble model is a lot more difficult, to the extent that some consider it a "black box".[11,32] Nonetheless, a measure of how each descriptor contributes to the prediction accuracy of the ensemble can be calculated during the course of training. In a previous

report[16] we showed, following Breiman,[10] how Random Forest calculates a descriptor importance based on the degree of the decrease in the performance when values of the descriptor are permuted. For SGB, Friedman[11] uses a different measure of variable importance which is also available in Random Forest[33] as one of the options. According to Friedman[11] the importance (squared) of variable x_p is

$$V_p^2 = \frac{1}{M}\sum_{m=1}^{M} V_p^2(T_m)$$

where $V_p^2(T_m)$ is the importance (squared) of x_p in the "construction" of the mth tree, calculated according to Breiman[4] as

$$V_p^2(T_m) = \sum_{t=1}^{J_m-1} \delta_t^2 I(\text{split\_variable}(t) = p)$$

Here, J_m is the number of nonterminal nodes in the mth tree; split_variable(t) is the variable which is split on at node t; and δ_t^2 is the improvement in the mean square error if node t is split, compared to if it were left as a terminal node. Hence, $V_p^2(T_m)$ is the sum of these improvements over all internal nodes of tree m for which variable p is the splitting variable, and V_p^2 is its mean over all trees. Once the squared importance V_p^2 of all variables is calculated, Friedman[11] suggests taking the square root of them and then normalize by assigning a value of 100 to the largest and scaling the others accordingly.

Partial Dependence Functions. While variable importance ranks the variables by their contribution to the reduction of the error loss, the partial dependence function[11] allows interpretation of how the response, for simplicity called $F(X)$ here, depends on the variables, according to the model. This is especially important when ensembles are used for finding structure−activity relationships, and the question is whether a molecule's activity increases or decreases if the value of a descriptor or combination of descriptors changes in a particular way. Usually, this interpretation is done by visualization using a partial dependence plot.[11] This plot shows the molecule activity (in regression) or the log-odds ratio (in classification) vs values of the "most important" descriptors and their combinations. In most cases, combinations of only two variables are considered, as the higher order combinations are very difficult to visualize and interpret.

The partial dependence function is defined by Friedman as follows.[11] Suppose the set X of descriptors can be partitioned as $X = \{X_I, X_U\}$, where the subset X_I contains descriptors whose dependence are of interest, while X_U contains all other descriptors. Then the partial dependence $F_I(X_I)$ is the marginal average

$$F_I(X_I) = E_{X_U}(F(X_I, X_U))$$

where the expectation E_{X_U} of the function $F(X)$ is taken with respect to the marginal probability distribution of components X_U. Thus, the partial dependence function represents "the effect of X_I on $F(X)$ after accounting for the (average) of the other variables X_U".[11] Suppose we are interested in $F_I(X_I^*)$ at a particular value X_I^*. It can be estimated as

$$\bar{F}_I(X_I^*) = \frac{1}{N}\sum_{n=1}^{N} F(X_I^*, X_{U,n})$$

where $X_{U,n}$ are samples of X_U in the training data.

It is important to point to two examples from Friedman[11] illustrating how the partial dependence function works. The first case is when $F(X)$ has the following additive structure:

$$F(X) = H_1(X_I) + H_2(X_U)$$

and the second case is when it is multiplicative:

$$F(X) = H_1(X_I)H_2(X_U)$$

It can be shown[11] that in the first case

$$F_I(X_I) = H_1(X_I) + \text{constant}$$

and in the second case

$$F_I(X_I) = H_1(X_I)\cdot\text{constant}$$

Thus, in both cases the partial dependence function is equal, up to a constant, to the function of interest, $H_1(X_I)$, while the constant term is $E_{X_U}(H_2(X_U))$ which is just the average of $H_2(X_U)$.

3. DATA SETS AND ASSESSMENT PROCEDURES

In this section, we describe the 10 data sets that we used to evaluate SGB and other QSAR methods. We also briefly describe the methods and procedure for performance assessment.

The data sets BBB, estrogen, P-gp, MDRR, and Cdk2 presented classification problems. The data sets dopamine, LogD, and Chan2 represent regression problems. The last data set, COX-2, presented data that could be treated as either classification or regression (we did both). A summary of the data sets is shown in Table 2. We consider Cdk2, LogD, and Chan2 as relatively large data sets (having on the order of 10^4 molecules and around 6000 descriptors) and the others as relatively small. We compare the performance of SGB with that of Random Forest (RF) and other commonly used QSAR methods: single Decision Tree (DT), partial least squares (PLS), k-nearest neighbors (KNN), support vector machines (SVM) with linear (lin_SVM) and radial basis function (rbf_SVM) kernels, and Naive Bayes (NB).

Except for NB which we used only for classification, all methods were used for both regression and classification tasks. Like SGB, the three methods, RF, DT, and SVM, have regression and classification capabilities which we used on the respective data sets. The use of PLS for classification and KNN for regression is to some extent arbitrary since PLS, in principle, is a regression technique, and KNN is a classification technique. In this paper, we used a classification modification of PLS. For all data sets we train the regression model using a pseudoresponse taking only two values, 0 and 1, representing the two classes. Except in the case of Cdk2 (see below), we then thresholded the predicted pseudoresponse at a level of 0.5 to convert it into a predicted class label. We also used the simplest regression modification of KNN: a compound's predicted activity is just the median of its k nearest neighbors' activities. For the data sets with

both binarized and nonbinarized atom pair descriptors[34] (see Table 2), we calculated nearest neighbors using Dice similarity,[35] while Euclidean distance was used for nonbinary and mixed data.

Four methods, SGB, RF, DT, and NB, do not require data normalization. For the other three, PLS, SVM, and KNN, the data were normalized in the following way. For PLS, each variable was normalized to have zero mean and unit variance. For SVM, each variable was scaled to be in the range -1 to 1. For KNN, when Dice similarity was used, the data were not normalized, and when Euclidean distance was used, the normalization was the same as that for PLS. In all cases, normalization was done on the training data, and the resulting scaling parameters were applied to the test data.

The data set Cdk2 is highly unbalanced, i.e., the number of active molecules (361) is by far smaller than the number of inactive ones, 15 079. A classification algorithm trained on highly unbalanced data tends to classify all samples to the majority class. To cope with this, we handled the Cdk2 data differently from the other classification data sets. For methods that can be effective with unbalanced data by reweighting the observations according to the class frequencies, we did so. These methods were SGB and DT. For other methods where this weighting leads to meaningless results, we had to utilize somewhat ad hoc approaches. For Random Forest we used its "balanced" version.[36] Here each tree of the forest is built on a sample from the data constructed as follows: half of the sample consists of bootstrap samples from the small class, and the other half has the same number of bootstrap samples from the large class. For PLS, the value of its tuning parameter, the number of factors, was selected simultaneously with the threshold on the predicted quantity, the pseudoresponse. For SVM, this approach can also be used, with the thresholding of estimated class probability.[37] However, we found that doing only this leads to meaningless results, so an ad hoc weighting of 10:1 of actives to inactives was added. For KNN the number of nearest neighbors k was optimized simultaneously with the threshold on the number of actives among the neighbors. See Table 3 for details.

Also, for the Cdk2 data, the criterion for parameter optimization for the SGB, PLS, SVM, and KNN methods was the average accuracy, which is equal to the average of true positive (TP) and true negative (TN) rates defined as follows:

$$\mathrm{TP} = \frac{\#(\text{correctly predicted actives})}{\#(\text{actives})}$$

$$\mathrm{TN} = \frac{\#(\text{correctly predicted inactives})}{\#(\text{inactives})}$$

To assess performance for small data sets, we did 10 replications of 5-fold cross-validation (CV). For all three large data sets, LogD, Chan2 and Cdk2, we performed 3 repetitions of random partitions of the data into training and test sets. The sizes of the partitions are given in Table 4.

To avoid an overly optimistic bias in performance, parameter optimization was done on the training data only. To accomplish this on large data sets, the training data were further split into a subtraining and subtesting part, whose sizes are given in Table 3. The subtraining part was used to build models with varying values of the parameters, while the subtesting part was used to test the performance of these models. Parameter values providing the best performance on the subtesting part were then used to build the final model on the whole training data. Once the model was created, it was used for prediction and performance evaluation on the test data. On small data sets we used nested cross-validation where the inner loop was used for parameter optimization, while the outer one was used for performance assessment. Thus, the parameter optimization was completely separated from the performance assessment. Software packages used in our analysis are given in Table 3. The R code we used to generate most of the results shown in this paper is available as a text file in Supporting Information. Sources for the publicly available data sets can be found in our previous report.[16]

Table 4. Sample Sizes of Training and Test Sets for Large Data Sets

HT	training size	test size
LogD	5631	5629
Chan2	5140	17962
Cdk2	7721	7719

Below, we describe the data sets in detail, case by case.

BBB. Doniger et al.[38] studied a set of molecules' permeabilities across the blood-brain barrier (BBB) into the central nervous system (CNS). A set of 325 compounds was culled from the literature and from a number of databases, and each compound was rated CNS active (180 compounds) or inactive (145 compounds) based on their passive diffusion through the BBB. There were nine continuous descriptors used (all given in their paper[38]), chemical and physical properties such as molecular weight, octanol−water partitioning coefficient (logP), various hydrogen bonding characteristics, etc.

Estrogen. Tong et al.[20] studied a set of 232 compounds measured by the National Center for Toxicological Research (NCTR) for their estrogen receptor binding activity. There are 131 actives (binding) and 101 inactives (nonbinding) in the data set. Dr. Weida Tong supplied us with a set of 197 Cerius-2 descriptors (of 8 different types).

P-gp. Penzotti et al.[39] compiled a set of 186 compounds from the literature, with their reported P-glycoprotein (P-gp) transport activity. In this data set, 108 compounds were reported as P-gp substrates and 78 were P-gp nonsubstrates. We worked with a set of 1522 binarized atom pair descriptors[34] generated in-house. (The original data set contained 195 compounds, but a number of them had proprietary structures that were not disclosed. We omitted the proprietary compounds from our study.)

MDRR. Bakken and Jurs[40] studied a set of compounds originally discussed by Klopman et al.,[41] who were interested in multidrug resistance reversal (MDRR) agents. The original response variable is a ratio measuring the ability of a compound to reverse a leukemia cell's resistance to adriamycin. However, the problem was treated as a classification problem, and compounds with the ratio > 4.2 were considered active, and those with the ratio ≤ 2.0 were considered inactive. Compounds with the ratio between these two cutoffs were called moderate and removed from the data for two-class classification, leaving a set of 528 compounds (298 actives and 230 inactives). (Various other arrangements of

Table 5. Mean Accuracy and (2SE) of 10 CV Replicates for Small Data Sets[a]

	BBB	estrogen	P-gp	MDRR	COX-2	Cdk2
SGB	0.789 (0.013)	0.824 (0.008)	0.757 (0.011)	0.826 (0.006)	0.789 (0.011)	0.721 (0.712, 0.728)
RF	0.806 (0.007)	0.827 (0.008)	0.804 (0.014)	0.831 (0.006)	0.771 (0.011)	0.747 (0.739, 0.759)
DT	0.727 (0.014)	0.758 (0.010)	0.703 (0.019)	0.779 (0.006)	0.734 (0.013)	0.700 (0.691, 0.711)
PLS	0.714 (0.008)	0.769 (0.008)	0.798 (0.016)	0.817 (0.007)	0.768 (0.009)	0.630 (0.573, 0.661)
KNN	0.771 (0.007)	0.718 (0.009)	0.739 (0.017)	0.799 (0.003)	0.693 (0.008)	0.716 (0.696, 0.736)
lin_SVM	0.729 (0.006)	0.778 (0.009)	0.760 (0.015)	0.815 (0.006)	0.774 (0.008)	0.705 (0.701, 0.712)
rbf_SVM	0.770 (0.011)	0.791 (0.012)	0.787 (0.009)	0.815 (0.009)	0.771 (0.011)	0.723 (0.714, 0.734)
NB	0.730 (0.005)	0.743 (0.014)	0.758 (0.007)	0.733 (0.002)	0.715 (0.012)	0.686 (0.670, 0.700)

[a] Mean average accuracy of 3 random partition replicates and (min,max) for large data set Cdk2.

Table 6. Mean RMSE and (2SE) of 10 CV Replicates for Small Data Sets[a]

	dopamine	COX-2	LogD	Chan2
SGB	0.558 (0.007)	0.859 (0.012)	0.711 (0.688, 0.729)	0.572 (0.570, 0.576)
RF	0.517 (0.010)	0.828 (0.009)	0.776 (0.756, 0.790)	0.575 (0.572, 0.579)
DT	0.627 (0.021)	0.998 (0.031)	1.220 (1.213, 1.230)	0.726 (0.718, 0.732)
PLS	0.527 (0.013)	0.929 (0.017)	0.865 (0.854, 0.871)	0.632 (0.631, 0.633)
KNN	0.568 (0.012)	0.979 (0.014)	0.988 (0.978, 0.994)	0.639 (0.637, 0.642)
lin_SVM	0.555 (0.024)	0.938 (0.013)	0.758 (0.744, 0.769)	0.594 (0.593, 0.596)
rbf_SVM	0.656 (0.042)	0.972 (0.048)	0.753 (0.745, 0.762)	0.565 (0.562, 0.568)

[a] Mean RMSE of 3 random partition replicates and (min,max) for large data sets LogD and Chan2.

Table 7. Mean R^2 and (2SE) of 10 CV Replicates for Small Data Sets[a]

	dopamine	COX-2	LogD	Chan2
SGB	0.391 (0.015)	0.390 (0.017)	0.860 (0.852, 0.868)	0.513 (0.504, 0.519)
RF	0.477 (0.020)	0.434 (0.012)	0.833 (0.829, 0.840)	0.508 (0.503, 0.510)
DT	0.227 (0.050)	0.175 (0.051)	0.586 (0.584, 0.589)	0.216 (0.207, 0.229)
PLS	0.454 (0.027)	0.287 (0.026)	0.792 (0.788, 0.796)	0.406 (0.400, 0.410)
KNN	0.366 (0.026)	0.208 (0.023)	0.729 (0.724, 0.733)	0.392 (0.382, 0.401)
lin_SVM	0.395 (0.051)	0.273 (0.020)	0.841 (0.838, 0.845)	0.843 (0.841, 0.845)
rbf_SVM	0.149 (0.108)	0.215 (0.078)	0.474 (0.473, 0.477)	0.525 (0.521, 0.528)

[a] Mean R^2 of 3 random partition replicates and (min,max) for large data sets LogD and Chan2.

these data were examined by Bakken and Jurs,[40] but we will focus on this particular one.) We did not have access to the original descriptors, but we generated a set of 342 descriptors of three different types that should be similar to the original descriptors, using the DRAGON[42] software.

Cdk2. Bradley et al.[43] studied a set of 17 550 compounds in a cyclin-dependent kinase 2 antagonists project to compare an informative design to a standard protocol (diversity - - similarity) under realistically simulated conditions. (There are 2832 compounds in the original data set that were not disclosed; we omitted those from our study.) The original response variable has three levels: inactive bin, gray bin, and active bin. We treated the gray bin as active bin, thus we have 10 579 inactives and 361 actives. We generated 6926 atom pair descriptors in-house for this data set.

Dopamine. Gilligan et al.[44] synthesized a set of 116 disubstituted piperidines and tested them for their binding affinity for the dopamine D_2 receptor. These drugs are of interest for treating schizophrenia and were previously examined using the trend vector method.[45] The biological activity is the $-\log(IC_{50})$, which ranges from 4.60 to 8.20.

LogD. This is a set of LogD measurements on 11 260 compounds, where LogD is defined as the shake flask water/octanol partition coefficient at pH 7.4. This is the same in-house data set described in Sheridan et al.,[46] and it is not publicly available. We used a set of 6303 in-house generated atom pair descriptors. The range of logD is -4.34 to 6.33.

Chan2. This is a set of measurements on binding to an unspecified channel protein for 23 102 compounds with 6918 atom pair descriptors. This is the same in-house data set described in Sheridan et al.,[46] and it is not publicly available. The range of biological activity, $-\log(IC50)$, is -0.71 to 11.23.

COX-2. Kauffman and Jurs[47] studied a set of 314 compounds whose inhibitory activities for the cyclooxygenase-2 (COX-2) enzyme were assayed by a laboratory at G. D. Searle and reported in a series of five literature publications. These drugs are of interest for use as nonsteroidal antiinflammatory drugs (NSAIDs). The biological activity is the $\log(IC_{50})$, which ranges from 0.23 to 5.00 reported for 272 compounds; the remainder have only > 5.00 recorded. For regression, only the 272 compounds with definite values were used; for classification, a cutoff of 2.5 was applied to the $\log(IC_{50})$, with values smaller than 2.5 considered active and values greater than 2.5 considered inactive. All 314 compounds were used in classification, so that there were 153 actives and 161 inactives. The original set of 135 continuous topological descriptors was kindly provided to us by Dr. Gregory Kauffman and Prof. Peter Jurs.

4. RESULTS OF COMPARISONS ON EXAMPLE DATA SETS

The results of the analysis described above are shown in Tables 5−10. In the tables, for all data sets except for Cdk2, "accuracy" in classification is the proportion of correctly classified molecules out of all predicted test molecules. For

Table 8. Median Accuracy of 10 CV Replicates for Small Data Sets[a]

	BBB	estrogen	P-gp	MDRR	COX-2	Cdk2
SGB	0.791	0.825	0.758	0.826	0.788	0.722
RF	0.805	0.825	0.804	0.830	0.771	0.744
DT	0.718	0.754	0.699	0.782	0.732	0.697
PLS	0.715	0.767	0.806	0.819	0.771	0.656
KNN	0.768	0.720	0.737	0.799	0.691	0.716
lin_SVM	0.728	0.778	0.750	0.814	0.772	0.702
rbf_SVM	0.778	0.793	0.785	0.812	0.775	0.721
NB	0.729	0.750	0.753	0.733	0.717	0.689

[a] Median average accuracy of 3 random partition replicates for large data set Cdk2.

Table 9. Median RMSE of 10 CV Replicates for Small Data Sets[a]

	dopamine	COX-2	LogD	Chan2
SGB	0.560	0.856	0.717	0.571
RF	0.517	0.827	0.783	0.575
DT	0.635	0.998	1.217	0.729
PLS	0.525	0.923	0.871	0.633
KNN	0.564	0.982	0.993	0.639
lin_SVM	0.554	0.939	0.760	0.594
rbf_SVM	0.652	0.958	0.752	0.565

[a] Median RMSE of 3 random partition replicates for large data sets LogD and Chan2.

Table 10. Median R^2 of 10 CV Replicates for Small Data Sets[a]

	dopamine	COX-2	LogD	Chan2
SGB	0.386	0.395	0.859	0.516
RF	0.475	0.435	0.829	0.510
DT	0.209	0.177	0.586	0.213
PLS	0.460	0.296	0.792	0.407
KNN	0.377	0.203	0.729	0.394
lin_SVM	0.397	0.271	0.839	0.842
rbf_SVM	0.167	0.241	0.473	0.527

[a] Median R^2 of 3 random partition replicates for large data sets LogD and Chan2.

Cdk2 "accuracy" is the average accuracy, i.e., the average of TP and TN calculated for the test data (see definitions in section 3). For regression, we report both the test set root-mean-square error (RMSE) and prediction R^2:

$$R^2 = 1 - \frac{(\mathrm{RMSE})^2}{\mathrm{Var}(Y)}$$

First we consider Tables 5−7. For the small data sets we report means and standard deviations of the performance statistics for 10 replicates of 5-fold cross-validation. For large data sets we report their means and the minimum and maximum values for random partitions of the data into equal sized training and test sets. To be consistent with our previous papers[16,17] where some of the same data sets were analyzed but only medians of the performance statistics were reported, we also report the medians in Tables 8−10. Some differences between the results in this and our previous papers are primarily due to (a) a different number of cross-validation replicates, 10 in this paper vs 50 before; (b) the use of different software packages for KNN and SVM; (c) a uniform approach for performance assessment undertaken in this paper, where we used a nested cross-validation for parameter tuning as described in section 3; and (d) possibly different types of data normalization.

To simplify the comparison of different methods, in Figures 1 and 2 we present plots of the regression and

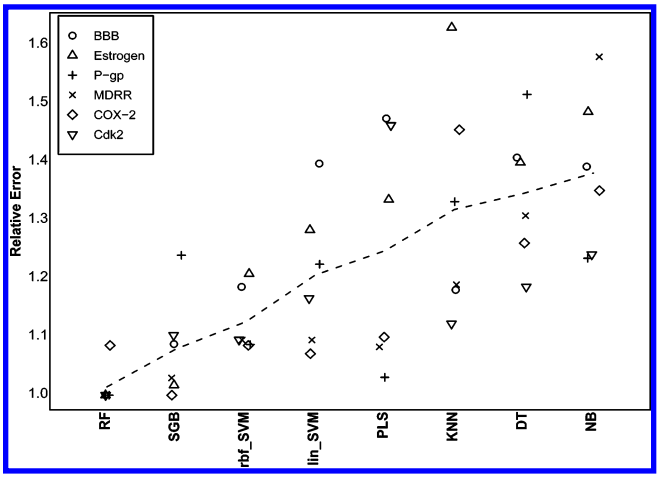

Figure 1. Summary of relative error, as described in the text, for classification data sets. Results are sorted by the mean performance of each method; the dashed line connects the means for each method. Data points are jittered for clarity.

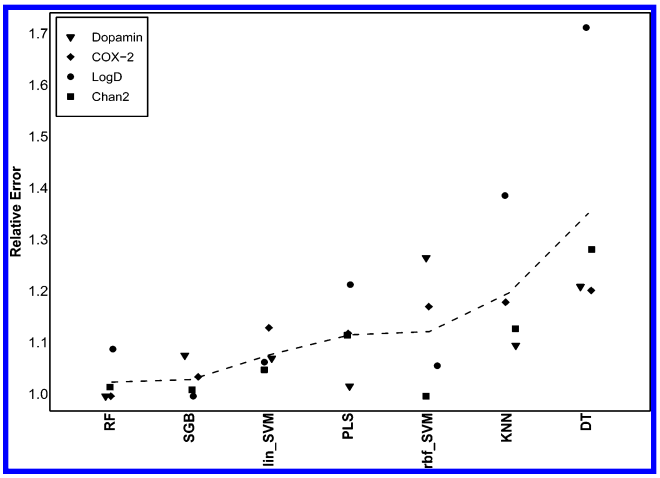

Figure 2. Summary of relative error, as described in the text, for regression data sets. Results are sorted by the mean performance of each method; the dashed line connects the means for each method. Data points are jittered for clarity.

classification relative errors suggested by Friedman.[48] To introduce them, let us denote by RMSE(M, D) the RMSE of a regression method M applied to the regression data set D. Similarly, denote by Accuracy(M, D) the accuracy of classification method M applied to classification data D. The relative error in the case of regression is the ratio

$$\frac{\mathrm{RMSE}(M, D)}{\mathrm{RMSE}(M^*, D)}$$

and in the case of classification it is

$$\frac{1 - \mathrm{Accuracy}(M, D)}{1 - \mathrm{Accuracy}(M^*, D)}$$

Here RMSE(M^*, D) is the minimum RMSE which is obtained on data D by the "best" (for this data set) regression method M^*. Similarly, Accuracy(M^*, D) is the maximum accuracy which is obtained on data D by the "best" classification method M^*. Thus, the "best" method (or methods in the case of ties) will have a relative error equal to one, while for all other methods it will be greater than one.

As one can see, the ensemble methods SGB and RF are on average the two best methods over all data sets considered in this paper. RF has on average a lower error in classification, and both RF and SGB have about the same average error in regression. Note also that SGB has the lowest error on the large regression data set logD, and the second lowest (after rbf_SVM) on another large data set, Chan2. This may suggest that SGB may be considered a preferred ensemble method for regression on large data sets. It is also notable that the training of SGB for the logD and Chan2 regression data sets took 4 to 9 times less time than that of RF.

On the large classification data set Cdk2, SGB is slightly worse than RF, on par with rbf_SVM, and superior to the other methods. The performance of SGB on the classification of small data sets is not consistently high. SGB's accuracy on the P-gp data is inferior to that of all methods except for DT and KNN. We tried to improve its performance by reducing the shrinkage parameter to 0.01, as suggested by Greg Ridgeway (personal communication), but it did not improve the performance. To this end, SGB may not be the preferred ensemble method for classification on small data sets.

Other results indicate that DT is significantly less accurate than both tree ensembles, RF and SGB. This is consistent with our previous study[16] and reports by many other investigators. The Naive Bayes method, which is becoming popular for compound classification, did even worse than DT.

The performance of KNN and PLS in classification varies significantly depending on the data. For example, KNN has a high accuracy on the large and unbalanced classification data Cdk2, on which PLS performs quite poorly. However on three small data sets, COX-2, P-gp, and estrogen, KNN's error was over 30% higher than that of PLS. The high performance of KNN on the large and unbalanced data set, Cdk2, might be at least partially attributed to the use of a similarity metric appropriate for very sparse data, which is usually the case when one uses atom pair descriptors.

The lin_SVM performed relatively well on the regression data sets. However, lin_SVM exhibits high variability in its performance among classification data sets. This was almost the opposite to that of rbf_SVM, which has high variability in performance on regression data instead. Of note is the relatively good performance of rbf_SVM on all 3 large regression and classification data sets. However, parameter tuning of both SVM methods for large data sets requires an excessive (almost prohibitive) amount of time.

In summary, tree ensembles seem to perform better than the other methods, on average, on the data sets examined here. In classification, RF seems to have less variability in performance than SGB. Both of them seem to be more consistent in performance across all data sets than the other methods, making them more suitable for general purpose modeling of diverse data sets.

5. DESCRIPTOR IMPORTANCE AND PARTIAL DEPENDENCE FUNCTIONS

To illustrate how descriptor importance and partial dependence functions can be used to generate hypotheses about SAR rules, we consider a proprietary set of in vitro Merck data on bidirectional P-glycoprotein (P-gp) transport for compounds from a certain drug program. In these data, the response variable is the logarithm of a measure of P-gp transport, where the higher the value, the stronger the substrate. We generated DRUGBITS descriptors for these molecules (described in Appendix B). These are descriptors corresponding to selected functional groups in the molecular structure. Such descriptors are more transparent than topological descriptors for purposes of interpretation, although the latter are usually better for model prediction accuracy. The data set has 900 compounds and 140 descriptors. The set of descriptors is fairly generic, rather than optimized for this particular data set.

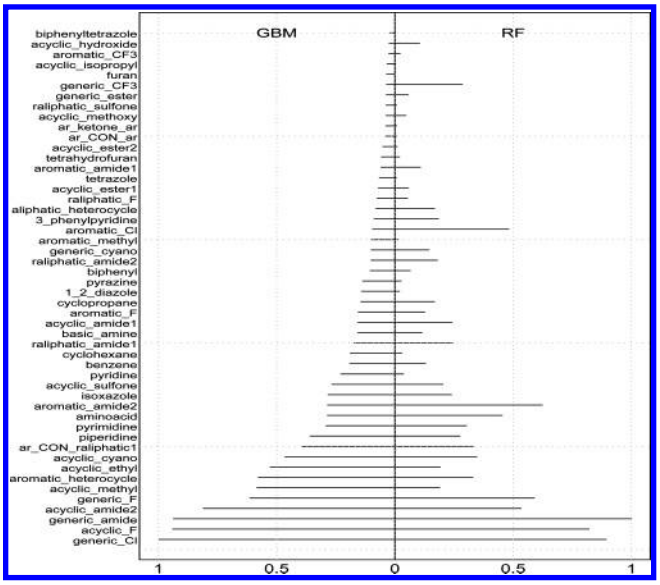

Figure 3. Descriptor importances for the top 49 DRUGBITS descriptors, sorted by importance, as found by SGB (left) and the corresponding importance values found by RF (right).

In this exercise, we first determined the optimal number of trees for the ensemble by running SGB within 5-fold cross-validation. Once the number of trees was determined, we trained the final ensemble using all the data (without splitting). All other SGB parameters were fixed as in Table 3. From the final model, we extracted the descriptor importance for all descriptors. We normalized them as described in section 2 (under "descriptor importance") and ranked them in order of importance. The highest ranking 49 of them are shown in Figure 3 along with the corresponding descriptor importance values from Random Forest run on the same data. (At the moment we do not have a criterion for selecting the number of descriptors, so for the convenience of plotting, we show about 1/3 of them in the figure.)

For these 49 descriptors, we generated partial dependence function plots, shown in Figure 4. Also, for comparison, the analogous plots for Random Forest were obtained and are shown in the same figure. For the examples discussed below, the partial dependence plot trends between SGB and RF are fairly consistent with each other.

Some of the descriptors are difficult to interpret and will not be discussed further here. Other descriptors have partial dependence trends that can be compared to SAR rules that were found for a specific series of compounds discussed in Hochman,[49] which constitute a very small subset of the data used in this exercise. In Hochman,[49] a number of functional groups were considered as substituents at a particular location in the compound. Compounds with each of the potential

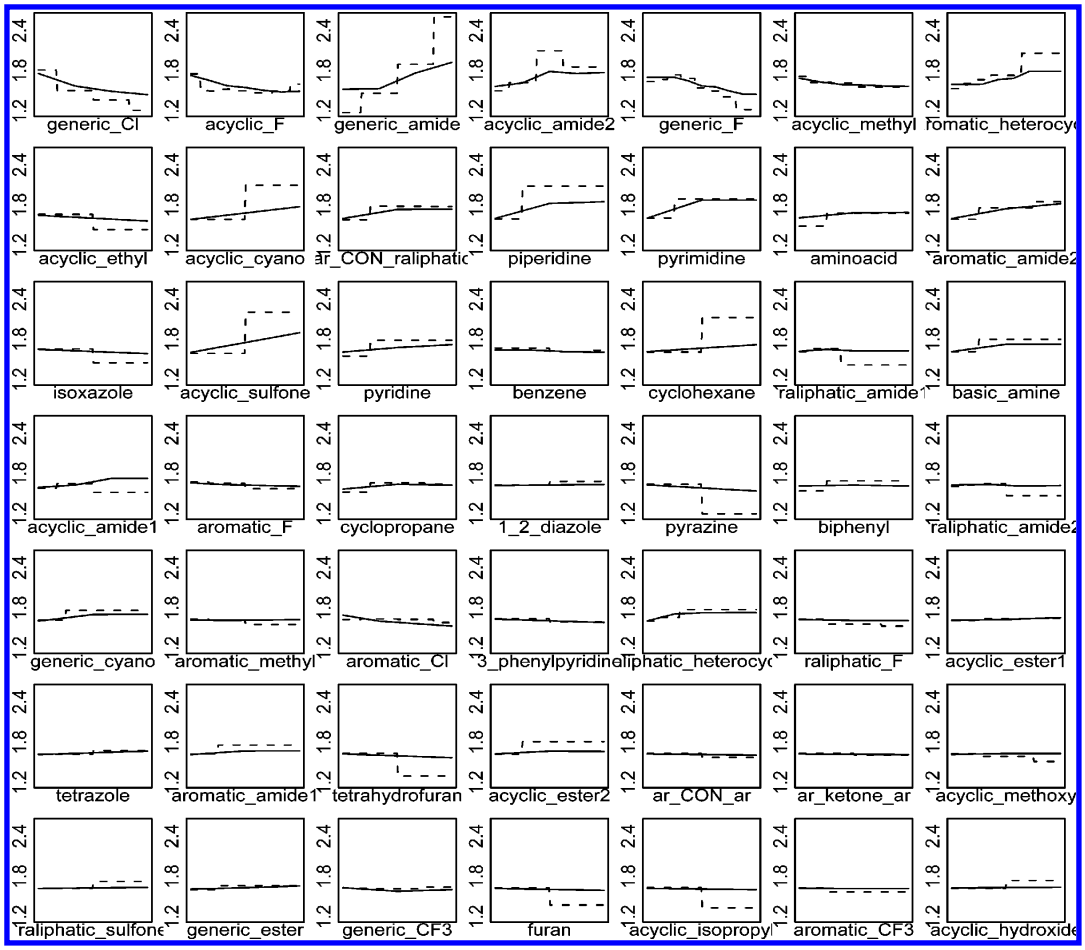

Figure 4. Partial dependence plots for the top 49 DRUGBITS descriptors according to SGB (SGB: dashed line, RF: solid line).

substituents had been synthesized, with the rest of the molecular structure kept fixed, and the corresponding P-gp activities were experimentally measured. The rules were generated by classifying these P-gp activities into three categories: low, moderate, and high transport. This type of analysis is sometimes called a "transformations" analysis, and only a few examples of the results were given in Hochman.[49] As a check on our partial dependence plot results, we compared the trends indicated therein with Hochman's transformations results. Because our trends indicate only whether a functional group tends to raise or lower P-gp activity, we compare them only with the transformations rules for low or high transport, which can clearly be associated with lowering and raising tendencies, respectively. We expect partial dependence functions to be much less sensitive for detecting structures identified with moderate activity by the transformations approach. This is because partial dependence functions deal with an average over diverse sets of molecules, in contrast to transformations which deal with identical molecules except for the substructure in question.

First, compare the results for descriptors that Hochman[49] found to be associated with low P-gp activity: aromatic chlorine, aromatic trifluoromethyl, and pyridine. (A fourth substructure, a phenyl ring with no substituents, is also included by Hochman[49] but is not represented in our descriptor set.) In the case of aromatic chlorine, although SGB found this descriptor important, the partial dependence plot from SGB shows essentially no trend. However, aromatic chlorine has a correlation of 0.975 with generic chlorine, which does show a strong decreasing trend, consistent with the hypothesis that this substructure tends to lower P-gp activity. Moreover, generic chlorine was found to be the most important descriptor by SGB (and second most important by RF). The generic chlorine descriptor is a count of how many chlorine atoms are present in a compound, whereas the aromatic chlorine descriptor counts only those bound to an aromatic ring. It seems that the presence of the generic chlorine descriptor in the model masks the effect of aromatic chlorine, with which it is highly correlated. We may surmise that our results are consistent with the transformations results.

In the case of aromatic trifluoromethyl, SGB ranks it low in importance, and the partial dependence plot shows no trend. However, this descriptor is highly correlated with two high ranking descriptors showing a strong decreasing trend, acyclic fluorine (correlation 0.855) and generic fluorine (correlation 0.749). Once again, these descriptors' presence in the SGB model may be masking the effect of aromatic trifluoromethyl. The Pg-p lowering effect of fluorine atoms, suggested by the partial dependence plots, is nonetheless consistent with the transformations rule. In the case of pyridine though, the partial dependence plot exhibits a weak increasing trend, which is not consistent with the transformations results. We could attribute this disagreement to the failure of the trend from the molecular series analyzed in Hochman[49] to generalize to the much larger data set used in this exercise.

Next, compare the descriptors that Hochman[49] found to be associated with high P-gp activity: aromatic sulfonamide and aromatic and ring-aliphatic amides. (A fourth substructure, *n*-alkoxyamide, was not represented in our data, probably due to a stringent requirement on apparent permeability that we used to exclude some compounds.) Most of the amide descriptors do correspond to the expected trend, and several have very high rank in terms of importance. Generic amide, in particular, has a very pronounced increasing trend and is ranked quite high and probably masks the effects of the more specific amide descriptors.

On the other hand, aromatic sulfonamide was ranked extremely low by SGB, and is not found among the top 49 descriptors shown in Figure 4. Nonetheless, it is highly correlated with acyclic sulfone (correlation 0.78), which is a relatively high ranking and has a strong increasing trend in its partial dependence plot. Thus, we surmise that the acyclic sulfone descriptor may be masking the effects of the aromatic sulfonamide descriptor, for which it is a substructure.

In conclusion, with one exception (pyridine), Hochman's transformation results[49] for a small subset of our data are consistent with the results of SGB partial dependence plots, if masking by highly correlated, chemically related descriptors is taken into account. This suggests that partial dependence functions may be useful for suggesting other, novel hypotheses about the influence of various functional groups on compounds' biological activity. However, the process of using descriptor importance and partial dependence plots for such work is not straightforward, since they cannot be taken at face value but instead interpreted using knowledge about the correlations of all descriptors with each other. The use of problem-specific descriptors may simplify the process of interpreting the partial dependence plots. See the work of He et al.[18] for further discussion of the use of descriptor importance and partial dependence functions for the interpretation of Boosting models in QSAR.

6. DISCUSSION

It is always difficult to make definitive conclusions about modeling methods based on a limited number of examples. However, it is still possible to point to some trends in the performance which should be verified or disproved by further research. Among these trends are the following.

Tree ensemble methods are the "safe methods" to be used with a variety of descriptors and small and large data sets. Using one of these methods (SGB or RF) will allow one to reach either the best or close to the best performance with no or relatively easy parameter tuning. This and additional features, such as descriptor importance and partial dependence functions, make tree ensembles powerful modeling tools particularly suited for QSAR modeling. Below we summarize our findings in more detail.

Our analysis did not indicate that one of the two ensemble methods, SGB and RF, is preferable in terms of performance, although it appears that SGB is slightly more accurate on large regression tasks and RF is slightly more accurate on classification tasks. Hence the choice between these two methods could be made based on other criteria. Thus, for example, Random Forest requires less tuning, whereas in Boosting, at least the number of trees should be optimized. However, on large data sets, Random Forest training requires very significant computing resources, and in some cases it may be prohibitively time-consuming. To this end, Boosting will provide a very valuable alternative, especially when the number of splits in the trees is small and/or the fraction of samples selected for tree building is small as well. Recent research by Friedman and Popescu suggested a very promising approach to increase the speed of Random Forest using regularized postprocessing.[48] We are currently investigating this approach.

During our investigation, we confirmed that a single decision tree is in general inferior in performance to most commonly used QSAR methods. The same could be said about Naive Bayes, which has become a quite popular QSAR method. Support vector machines, which are also popular in QSAR modeling, can provide a reasonable performance, in some cases comparable or even better than RF and SGB. However, the effort for tuning SVM and the lack of interpretability of the results for rbf_SVM make these tools less attractive in QSAR modeling. KNN, when used with Dice similarity, seems to be an adequate method for classification in large data sets with topological descriptors. Of the two linear methods, PLS and lin_SVM, the latter seems on average to perform better in both classification and regression, although the training of lin_SVM requires a lot more time. However, both of these methods exhibit high performance variability in classification.

During the course of this analysis, we observed that the results may depend on the specifics of the software implementation for different methods. Specifically, we noticed that results generated by the software used in this paper, GBM, are somewhat different from those generated by Friedman's software ("MART," which stands for "multiple additive regression trees"), on a published data set (spam data).[48] Although the conclusions we made in this paper would not practically change if the original Friedman software had been used, investigators should not be surprised by a difference between the results. In the interest of full disclosure, we were not able to use Friedman's software, which is licensed by him to Salford Systems. The version of this software sold to us by Salford was not operable on Merck's computers.

In general we are pleased with the results illustrating how variable importance and partial dependence can be used in QSAR. In the exercise on proprietary P-gp data, we were able to infer five out of six known SAR rules using these methods. This is especially rewarding since we did not use descriptors optimized for this data set. However, this exercise clearly indicates that there is no magic bullet allowing an easy induction of SAR rules from empirical data. The results of any statistical procedure should be verified by a subject matter expert, and even after that, they should be considered only as hypotheses and not the final conclusions.

ACKNOWLEDGMENT

We are very grateful to Jerry Friedman, Leo Breiman, and Tom Dietterich for many discussions and consultations regarding ensemble learning. We thank Jerry in particular for his outstanding research on Stochastic Gradient Boosting. We thank the following individuals for providing data and comments on their work: Peter Grootenhuis, Jerome Hochman, Michelle Lamb, Peter Jurs, Gregory Kauffman, Julie

Penzotti, Weida Tong, and Joanne Yeh. We also thank the authors of the software that we used, especially Greg Ridgeway for GBM, David Meyer for SVM in the e1071 package, and Chih-Jen Lin for LIBSVM. We are grateful to Greg Ridgeway and Bruce Bush for a careful reading of the manuscript and comments to improve the paper. Finally, we thank Bogdan Popescu, Chris Culberson, and Brad Feuston for useful discussions.

A. APPENDIX: BOOSTING INSIGHTS

In a previous communication[16] we demonstrated following Breiman[25] and using Domingos' Bias-Variance decomposition[50] that the tree ensembles, Bagging and Random Forest, significantly improve on a single unpruned tree's performance by reducing its variance while leaving its already low bias almost unchanged. (The exact definitions of bias and variance are given below for the classification case). The interaction depth of the trees in GBM is a tuning parameter,[29] thus the trees comprising the ensemble may have low bias if they have relatively many splits, or high bias if they they have relatively few splits. In the example below we demonstrate that the remarkable gain in performance of the ensembles of trees built by SGB over a single tree is due to a reduction in either the bias or the variance, depending on which is the dominant component of a single tree's error. (The bias-variance decomposition for boosting, using various formulations, has also been discussed elsewhere, e.g., by Schapire et al.[51])

To give formal definitions, suppose that a feature vector $X \in \mathcal{R}^p$ and associated class label $Y \in \{1, 2,..., n_{\text{class}}\}$ are two random variables whose joint probability distribution is $p(x, y)$. The classification problem is to determine a classifier such that given a vector, X, issues a prediction of its class label, $\hat{Y}(X)$. The quality of this prediction is measured by the expectation of some loss function. The expectation is taken over all X and Y with respect to their distribution $p(x, y)$. Although as presented in section 2 the loss function for SGB classification is the binomial log-likelihood, we will consider here label misclassification as the loss function. The function, binomial log-likelihood, chosen by SGB is in fact a substitute for label misclassification because of the ease of fitting an additive model to a function. It also turns out that the log-likelihood loss is more robust to outliers.[11] The expectation of label misclassification, the error rate, is equal to the probability of misclassification:

$$\text{ER} = E_{X,Y} I(\hat{Y} \neq Y) = \Pr\{\hat{Y} \neq Y\}$$

Here $E_{X,Y}$ is the expectation with respect to $p(x, y)$ and Pr is the probability.

When the distribution $p(x, y)$ is known, one can build the optimal, Bayes classifier, i.e., the classifier with the minimum error rate.[52] However, in most practical applications the distribution $p(x, y)$ is unknown; instead only the training data $D = \{(X_1, Y_1), ...,(X_n, Y_n)\}$, randomly drawn from this distribution is available. The goal then is to create a classifier that makes a prediction $\hat{Y}(X, D)$ by minimizing some estimate of the error rate

$$\text{ER} \approx E_{X,Y,D}(\hat{Y}(X, D) \neq Y) = \Pr\{\hat{Y}(X, D) \neq Y\}$$

where the expectation is taken over all X, Y and training data D. The bias-variance decomposition is a way of representing the error rate ER as a sum of interpretable components. We will follow here Domingos' development.[50] In addition to the classifier prediction $\hat{Y}(X, D)$, Domingos considers two other predictions. The first is $Y^*(X)$ which is the prediction of the optimal, Bayes classifier, whose error rate is minimum. The second is called the main prediction in Domingos, but we will call it the aggregated, or average prediction

$$\hat{Y}_A(X) = \text{argmax}\{V_1(X), V_2(X), ..., V_{n_{\text{class}}}(X)\}$$

where $V_i(X) = \Pr_D\{\hat{Y}(X, D) = i\}$ is the probability that given a feature vector X, a classifier trained on the *randomly selected* training data D, casts a vote $\hat{Y}(X, D)$ for class i. Hence, $\hat{Y}_A(X)$ is the class label most frequently voted for by the replicas of the classifier trained on different data sets. With these definitions, Domingos showed the following

$$\text{ER} = E_X[h_1(X)\cdot\text{Noise}(X) + \text{Bias}(X) + h_2(X)\cdot\text{Var}(X)]$$

where

$$\text{Noise}(X) = I(Y^*(X) \neq Y)$$

$$\text{Bias}(X) = I(\hat{Y}_A(X) \neq Y^*(X))$$

$$\text{Var}(X) = \Pr_D\{\hat{Y}(X, D) \neq \hat{Y}_A(X)\}$$

$$h_1(X) = 2\cdot\Pr_D\{\hat{Y}(X, D) = Y^*(X)\} - 1$$

$$h_2(X) = 1 \text{ if } \text{Bias}(X) = 0, -1 \text{ otherwise}$$

The meanings of the elements in this decomposition are the following. Noise(X) is an "irreducible" component of the error rate, which is independent of the classifier. Bias(X) reflects how the average prediction of the feature vector X is different from the optimal prediction. Var(X) shows the classifier's "variability", i.e., how much the prediction by each replica of the classifier would disagree with the average of such predictions. Coefficients h_1 and h_2 reflect interactions between all three components. The presence of these interactions is fundamental for classification as opposed to regression. Specifically, an increase in the variance in the presence of the bias, Bias(X) = 1, *decreases* the error rate ($h_1(x) = -1$). The opposite, an *increase* in the error rate with the increase in the variance, happens when the bias is zero. Similarly, the error rate decreases when Noise(X) = 1 if simultaneously $\Pr_D\{\hat{Y}(X, D) = Y^*(X)\} < 0.5$, i.e., the prediction of the majority of classifiers disagrees with the optimal prediction. Taking expectation (averaging over all X) in the decomposition yields

$$\text{ER} = \text{Noise} + \text{Bias} + \text{Var}$$

where Noise = $E_X(h_1(x)\cdot\text{Noise}(x))$; Bias = $E_X(\text{Bias}(X))$; and Var = $E_X(h_2(X)\cdot\text{Var}(x))$. In addition to the variance, Domingos considers its two components, "biased", Var_B, and "unbiased", Var_U, such that Var = $\text{Var}_U - \text{Var}_B$ with $\text{Var}_U = E_X\{I(\text{Bias}(x) = 0)\cdot\text{Var}(x)\}$ and $\text{Var}_B = E_X\{(1 - I(\text{Bias}(x) = 0))\cdot\text{Var}(x)\}$.

We now consider an example of a 2-class classification problem with simulated data from Refs. arcing:breiman and rf:breiman, where it is called "threenorm". This example consists of 20 dimensional, 2 class data. Class #1 is drawn

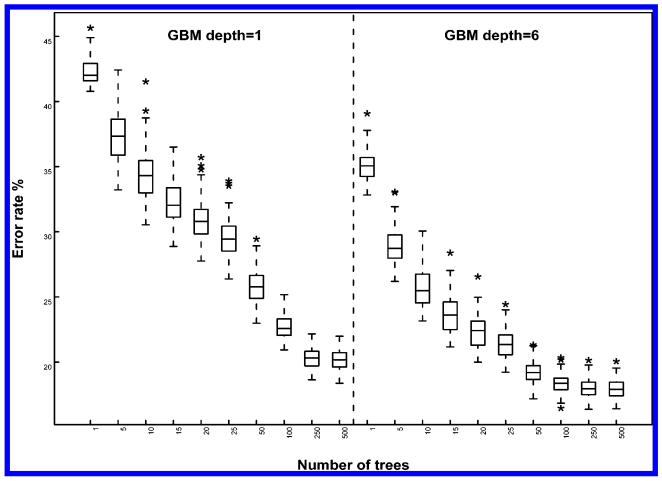

Figure 5. Boxplots of error rates for 100 replications of classification of threenorm data (simulation). Results are shown for GBM with depth = 1 and depth = 6. Boxplots of GBM results are shown for different numbers of trees. Horizontal lines inside the boxes are the median error rates. The plot shows the effect of an ensemble of trees on the reduction of the error rate. Moreover, the error rate of GBM with depth = 6 is progressively lower than that for depth = 1.

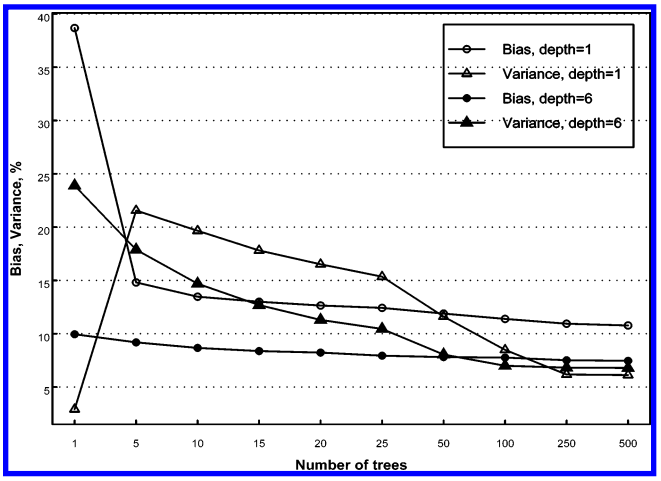

Figure 6. Bias-variance decomposition for GBM with depth = 1 and depth = 6 for the threenorm simulated data. The bias and variance components (in percent) are shown. See text for discussion.

with probability 1/2 from a multivariate normal distribution with mean vector (a, a,..., a) and with probability 1/2 from a multivariate normal with mean vector ($-a$, $-a$,..., $-a$). Class #2 is drawn from a multivariate normal with mean vector (a, $-a$, a, $-a$,..., a, $-a$). All normal distributions have unit covariance matrices. Both classes have the same number of samples.

For training we generated 100 data sets D_1, D_2,..., D_{100} with $N = 300$ samples in each. Using these 100 data sets we created groups of 100 classifiers with each group being a particular classifier type, SGB ensembles of varying number of trees having either depth = 1 (stumps) or depth = 6. All classifiers were then used to predict a large $n_{test} =$ 18 000 test set; error rates and components of the bias-variance decomposition were calculated.

Breiman[25] notes that this problem where "the separating surface, formed by the continuous join of two oblique hyperplanes" is difficult for classification trees since it is hard to approximate this surface by multidimensional rectangles with sides parallel to the axes.

Figure 5 shows the error rates for SGB ensembles of a varying number of depth = 1 and depth = 6 trees. Each boxplot represents the distribution of 100 error rates. The effect of tree aggregation is evident. When the number of trees increases, the median error rate decreases from 42.3% (single depth = 1 tree) and 35.1% (depth = 6 tree) to 20.2% and 17.9%, respectively. Note that the Bayes error rate in this example is 10.4%.

Figure 6 shows the bias and variance components of the error rate decomposition. To avoid overloading the figure, we omit the Noise component, which is much smaller than the others, and plot only the total variance, omitting Var_U and Var_B. In the case of the depth = 1 tree ensemble, an increase in the number of trees decreases the bias from about 38.7% to about 10.8%. The variance, on the other hand, first increases from about 2.3% to 21.6% and then decreases to about 6.1%, still exceeding its initial value. Thus, the reduction in the error rate is entirely due to the decrease in bias. In the case of the depth = 6 tree ensemble, the situation is different. Here the bias decreases from about 10% to only 7.5%, while the variance decreases from 23.9% to 6.8%. Thus, the reduction in the error rate is almost entirely to due to a decrease in the variance.

B. APPENDIX: DRUGBITS

DRUGBITS is an attempt to define substructure descriptors that are immediately interpretable by chemists in that they encode recognizable groups. A drawback of such an approach, however, is that interpretable descriptors often do not give the best QSAR fits and/or predictions. Having been built up over a period of years, DRUGBITS is a mixture of (presumably useful) substructures from various sources. There is no attempt to keep the definitions mutually exclusive. For example, a molecule may contain both an "aromatic heterocycle" and "imidazole", even though these refer to the same set of atoms.

Sources are as follows:

1. The most common ring systems, the most common ring substituents, and most common linkers in the MDDR[53] database and the Merck in-house sample collection. The parsing of these is similar to that used by Bemis and Murcko[54,55] except that the element and the hybridization of the atoms was taken into account.

2. Agreed upon "bad groups" to be eliminated from the sample collection, e.g. hydrazines, nitrosos, etc.

3. Later additions requested by Merck scientists. The substructures here are given in PATTY[56] notation. They can be found in the Supporting Information file, drug_bits.txt.

Supporting Information Available: The R code used to generate most of the results. This material is available free of charge via the Internet at http://pubs.acs.org.

REFERENCES AND NOTES

(1) Penzotti, J. E.; Landrum, G. A.; Putta, S. Building predictive ADMET models for early decisions in drug discovery. *Curr. Opin. Drug Discovery Dev.* **2004**, *7*, 49−61.
(2) Tong, W.; Welsh, W. J.; Shi, L.; Fang, H.; Perkins, R. Structure−activity relationship approaches and applications. *Environ. Tox. Chem.* **2003**, *22*, 1680−1695.
(3) Weaver, D. C. Applying data mining techniques to library design, lead generation and lead optimization. *Curr. Opin. Chem. Biol.* **2004**, *8*, 264−270.

(4) Breiman, L.; Friedman, J. H.; Olshen, R. A.; Stone, C. J. *Classification and Regression Trees;* Chapman & Hall/CRC: Boca Raton, 1984.
(5) Rusinko, A., III; Farmen, M. W.; Lambert, C. G.; Brown, P. L.; Young, S. S. Analysis of a large structure/biological activity data set using recursive partitioning. *J. Chem. Inf. Comput. Sci.* **1999**, *39*, 1017−1026.
(6) Dietterich, T. G. Ensemble learning. In *The Handbook of Brain Theory and Neural Networks*, 2nd ed.; Arbib, M. A., Ed.; The MIT Press: Cambridge, 2002.
(7) Breiman, L. Bagging predictors. *Machine Learning* **1996**, *24*, 123−140.
(8) Freund, Y.; Schapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. System Sci.* **1997**, *55*, 119−139.
(9) Friedman, J. H. Greedy function approximation: a gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189−1202.
(10) Breiman, L. Random forests. *Machine Learning* **2001**, *45*, 5−32.
(11) Hastie, T.; Tibshirani, R.; Friedman, J. H. *The Elements of Statistical Learning;* Springer-Verlag: New York, 2001.
(12) Meyer, D.; Leisch, F.; Hornik, K. The support vector machine under test. *Neurocomputing* **2003**, *55*, 169−186.
(13) Dixon, S. L.; Villar, H. O. Investigation of classification methods for the prediction of activity in diverse chemical libraries. *J. Comput.-Aided Mol. Des.* **1999**, *13*, 533−545.
(14) Susnow, R. G.; Dixon, S. L. Use of robust classification techniques for the prediction of human cytochrome P450 2D6 inhibition. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 1308−1315.
(15) Cheng, A.; Dixon, S. L. In silico models for the prediction of dose-dependent human hepatotoxicity. *J. Comput.-Aided Mol. Des.* **2003**, *17*, 811−823.
(16) Svetnik, V.; Liaw, A.; Tong, C.; Culberson, J. C.; Sheridan, R. P.; Feuston, B. P. Random forest: a classification and regression tool for compound classification and QSAR modeling. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 1947−1958.
(17) Svetnik, V.; Liaw, A.; Tong, C.; Wang, T. Application of Breiman's random forest to modeling structure−activity relationships of pharmaceutical molecules. In *Multiple Classifier Systems 2004*; Roli, F., Kittler, J., Windeatt, T., Eds.; LNCS 3077; Springer-Verlag: Berlin, 2004; 334−343.
(18) He, P.; Xu, C.-J.; Liang, Y.-Z.; Fang, K.-T. Improving the classification accuracy in chemistry via boosting technique. *Chemom. Intell. Lab. Syst.* **2004**, *70*, 39−46.
(19) Hawkins, D. M.; Musser, B. J. One tree or a forest? Alternative dendrographic models. *Comput. Sci. Stat.* **1999**, *30*, 534−542.
(20) Tong, W.; Hong, H.; Fang, H.; Xie, Q.; Perkins, R. Decision forest: combining the predictions of multiple independent decision tree models. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 525−531.
(21) Ho, T. K. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Analysis Machine Intell.* **1998**, *20*, 832−844.
(22) van Rhee, A. M. Use of recursion forests in the sequential screening process: consensus selection by multiple recursion trees. *J. Chem. Inf. Comput. Sci.* **2003**, *43*, 941−948.
(23) DeLisle, R. K.; Dixon, S. L. Induction of decision trees via evolutionary programming. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 862−870.
(24) Friedman, J. H. Stochastic gradient boosting. *Comput. Stat. Data Analysis* **2002**, *38*, 367−378.
(25) Breiman, L. Arcing classifiers. *Annals Stat.* **1998**, *26*, 801−849.
(26) Codrington, C. W. Boosting with confidence information. *Proc. Eighteenth Int. Conf. Machine Learning* **2001**, 59−65.
(27) Meir, R.; Ratsch, G. An introduction to boosting and leveraging. In *Advanced Lectures on Machine Learning*; Mendelson, S., Smola, A., Eds.; Springer: New York, 2003; pp 119−184.
(28) *The Mathematics of Generalization;* Wolpert, D. H., Ed.; Addison-Wesley: Reading, 1995.
(29) Ridgeway, G. *The gbm package*. Reference manual available at http://cran.r-project.org/.
(30) The Comprehensive R Archive Network, http://cran.r-project.org.
(31) Schwenk, H.; Bengio, Y. Boosting neural networks. *Neural Comput.* **2000**, *12*, 1869−1887.
(32) Breiman, L. Wald lectures. 277th meeting of the Institute of Mathematical Statistics, Banff, Alberta, Canada (July 28 to July 31, 2002).
(33) Liaw, A.; Wiener, M. *The randomForest package*, http://cran.r-project.org/src/contrib/Descriptions/randomForest.html
(34) Carhart, R. E.; Smith, D. H.; Venkataraghavan, R. Atom pairs as molecular features in structure−activity studies: definition and applications. *J. Chem. Inf. Comput. Sci.* **1985**, *25*, 64−73.
(35) Willett, P.; Barnard, J. M.; Downs, G. M. Chemical similarity searching. *J. Chem. Inf. Comput. Sci.* **1998**, *38*, 983−996.
(36) Chen, C.; Liaw, A.; Breiman, L. *Using random forest to learn imbalanced data*; Technical report; Statistics Department, UC Berkeley, 2004.
(37) Chang, C.-C.; Lin, C.-J. LIBSVM- -A library for support vector machines, http://www.csie.ntu.edu.tw/cjlin/libsvm/.
(38) Doniger, S.; Hofmann, T.; Yeh, J. Predicting CNS permeability of drug molecules: comparison of neural network and support vector machine algorithms. *J. Comput. Biol.* **2002**, *9*, 849−864.
(39) Penzotti, J. E.; Lamb, M. L.; Evensen, E.; Grootenhuis, P. D. J. A computational ensemble pharmacophore model for identifying substrates of P-glycoprotein. *J. Med. Chem.* **2002**, *45*, 1737−1740.
(40) Bakken, G. A.; Jurs, P. C. Classification of multidrug-resistance reversal agents using structure-based descriptors and linear discriminant analysis. *J. Med. Chem.* **2000**, *43*, 4534−4541.
(41) Klopman, G.; Shi, L. M.; Ramu, A. Quantitative structure−activity relationship of multidrug resistance reversal agents. *Mol. Pharmacol.* **1997**, *52*, 323−334.
(42) *DRAGON;* Milano Chemometrics, http://www.disat.unimib.it/chm/Dragon.htm.
(43) Bradley, E.; Miller, J.; Saiah, E.; Grootenhuis, P. Informative library design as an efficient strategy to identify and optimize leads: Application to cyclin-dependent kinase 2 antagonists. *J. Med. Chem.* **2003**, *46*, 4360−4364.
(44) Gilligan, P. J.; Cain, G. A.; Christos, T. E.; Cook, L.; Drummond, S.; Johnson, A. L.; Kergaye, A. A.; McElroy, J. F.; Rohrbach, K. W.; Schmidt, W. K.; Tam, S. W. Novel piperidine σ receptor ligands as potential antipsychotic drugs. *J. Med. Chem.* **1992**, *35*, 4344−4361.
(45) Sheridan, R. P.; Nachbar, R. B.; Bush, B. L. Extending the trend vector: the trend matrix and sample-based partial least squares. *J. Comput.-Aided Mol. Des.* **1994**, *8*, 323−340.
(46) Sheridan, R. P.; Feuston, B. P.; Maiorov, V. N.; Kearsley, S. K. Similarity to molecules in the training set is a good discriminator for prediction accuracy in QSAR. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 1912−1928.
(47) Kauffman, G. W.; Jurs, P. C. QSAR and *k*-nearest neighbor classification analysis of selective cyclooxygenase-2 inhibitors using topologically based numerical descriptors. *J. Chem. Inf. Comput. Sci.* **2001**, *41*, 1553−1560.
(48) Friedman, J. H.; Popescu, B. E. Importance sampled learning ensembles, http://www-stat.stanford.edu/ jhf/ftp/isle.pdf.
(49) Hochman, J. H. Role of mechanistic transport studies in lead optimization. AAPS workshop on optimization of drug-like properties, 19−22 September 2004, Parsippany, NJ.
(50) Domingos, P. A unified bias-variance decomposition for zero-one and squared loss. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*; AAAI Press: Austin, 2000.
(51) Schapire, R. E.; Freund, Y.; Bartlett, P.; Lee, W. S. Boosting the margin: a new explanation for the effectiveness of voting methods. *Ann. Stat.* **1998**, *26*, 1651−1686.
(52) Fukunaga, K. *Introduction to Statistical Pattern Recognition*, 2nd ed.; Morgan Kaufmann: San Diego, 1990.
(53) MDL Drug Data Report licensed from Molecular Design Ltd., San Leandro, CA.
(54) Bemis, G. W.; Murcko, M. A. Properties of known drugs. 1. molecular frameworks. *J. Med. Chem.* **1996**, *39*, 2887−2893.
(55) Bemis, G. W.; Murcko, M. A. Properties of known drugs. 2. side chains. *J. Med. Chem.* **1999**, *42*, 5095−5099.
(56) Bush, B. L.; Sheridan, R. P. PATTY: a programmable atom typer and language for automatic classification of atoms in molecular databases. *J. Chem. Inf. Comput. Sci.* **1993**, *33*, 756−762.

CI0500379