Classification Tree Models for the Prediction of Blood−Brain Barrier Passage of Drugs

Eric Deconinck,[†] Menghui H. Zhang,[†] Danny Coomans,[‡] and Yvan Vander Heyden*,[†]

Department of Analytical Chemistry and Pharmaceutical Technology, Pharmaceutical Institute, Vrije Universiteit Brussel-VUB, Laarbeeklaan 103, B-1090 Brussels, Belgium, and Statistics and Intelligent Data Analysis Group, James Cook University, Townsville, Australia 4814

Received November 28, 2005

The use of classification trees for modeling and predicting the passage of molecules through the blood−brain barrier was evaluated. The models were built and evaluated using a data set of 147 molecules extracted from the literature. In the first step, single classification trees were built and evaluated for their predictive abilities. In the second step, attempts were made to improve the predictive abilities using a set of 150 classification trees in a boosting approach. Two boosting algorithms, discrete and real adaptive boosting, were used and compared. High-predictive classification trees were obtained for the data set used, and the models could be improved with boosting. In the context of this research, discrete adaptive boosting gives slightly better results than real adaptive boosting.

1. INTRODUCTION

In drug discovery, many molecules found to be potentially useful as drugs fail in a later stage of drug development, because of improper ADME-Tox (absorption, distribution, metabolization, elimination, and toxicological) properties. The pharmaceutical industry is, therefore, very interested in the development of methods that allow screening for these properties in the earliest stages of drug development. In silico methods are particularly interesting because they allow screening for these properties when molecules are not even synthesized. In silico methods build relationships between the ADME-Tox properties of a molecule and theoretical or experimental parameters (descriptors), describing chemical and physical characteristics of the molecule. In general, these kinds of relationships are called quantitative structure−activity relationships (QSARs). An important property for potential drug molecules is their ability to pass the blood−brain barrier. Some molecules need to pass this barrier because their target is situated in the central nervous system, while others may not, because of unwanted toxicological properties. Many authors have built QSAR models for the prediction of blood−brain barrier passage using different chemometric techniques such as multiple linear regression,[1−4] principal components analysis (PCA) and principal components regression,[5,6] partial least squares regression,[5,7] and artificial neural networks.[8,9] Most models obtained have the drawback of being not easily interpretable for researchers with a limited statistical and chemometric background.

In this paper, attempts were made to build an easily interpretable model resulting in a classification of molecules, allowing a decision of whether a molecule will or will not pass the blood−brain barrier. To do so, the research was focused on the use of a tree-based method, because decision trees are easy to use and interpret. In the first step, classification trees were built using the classification and regression trees **(**CART) method for categorical variables, as described by Breiman et al.[10] In the second step, boosting[11] was applied using classification trees as weak learners. This method was called boosting CART. Boosting CART generates a set of classification trees, in which each tree focuses on the objects that were wrongly classified in the previous tree. In previous work, CART was applied successfully for the prediction of absorption classes.[12,13] The potential of CART in quantitative structure retention and selectivity relationships was also shown by Put et al.[14] and Caetano et al.[15] Boosting CART was applied successfully by Zhang et al.[16] to five real chemical data sets, situated in the fields of environmental science, pharmaceutics, food science, and analytical chemistry. He et al.[17,18] showed recently that boosting, in general, can significantly improve the prediction performance of any single classification method and this in the fields of epidemiology, chemistry, and spectral data.

In the first step, classification trees were built using CART. In the second step, two different boosting algorithms, discrete adaptive boosting (AdaBoost)[16] and real AdaBoost,[16,19] were applied in order to improve the obtained CART model. The models were evaluated for their predictive abilities using a 20-bootstrap with replacement method.[20] Both types of boosting models were compared, and it was evaluated whether the application of boosting resulted in a significant improvement of the prediction abilities.

2. THEORY

2.1. Classification And Regression Trees (CART). CART is a nonparametric statistical technique, developed by Breiman et al.,[10,12] that is capable of solving classification and regression problems for both categorical and continuous dependent variables. If the dependent variable is categorical, CART will produce a classification tree; when it is continuous, a regression tree will be obtained. In both types, CART tries to build a decision tree, which describes a response

* Corresponding author tel: +32 2 477 47 34; fax: +32 2 477 47 35; e-mail: yvanvdh@vub.ac.be.
[†] Vrije Universiteit Brussel-VUB.
[‡] James Cook University.

10.1021/ci050518s CCC: $33.50
Published on Web 02/18/2006

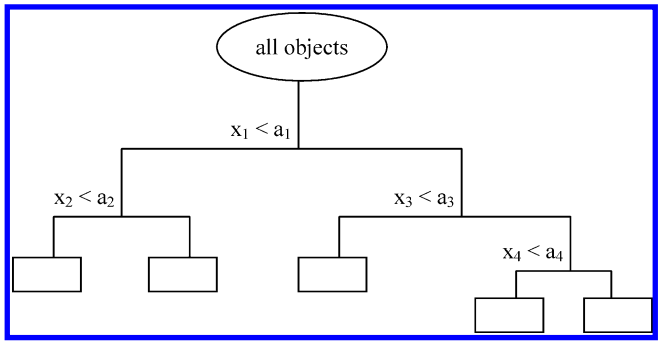

Figure 1. General structure of a CART model. x_i = selected split variable, a_i = selected split value.

variable as a function of different explanatory variables (Figure 1).

A CART analysis generally consists of three steps. In the first step, the maximum tree is built, using a binary split procedure. The maximum tree is an overgrown model, which closely describes the training set and usually shows overfitting. In the second step, this overfitted model is pruned. This procedure results in a series of less-complex trees, derived from the maximum tree. In the third and final step, the optimal tree is selected using a cross-validation procedure.[10,12]

2.1.1. Building the Maximum Tree. The maximum tree is built using a binary split procedure, starting at the tree root. The tree root consists of all objects in the training set. In every step of the procedure, a mother group is considered and split into two daughter groups. The split is chosen in such a way that the impurity of the daughter groups is lower than that of the mother group. This means that the daughter groups become more homogeneous in the response variable (class numbers). In the following step, each daughter group is considered a mother group. Every split is defined by one value of one explanatory variable. For continuous explanatory variables, the splits are defined by "$x_i < a_j$" where x_i is the selected variable and a_j its split value.[10,12]

To choose the most appropriate variable and split value, CART uses an algorithm in which all descriptors and all possible split values are considered. The split resulting in the highest decrease in impurity between the mother group (t_p) and the daughter groups (t_L and t_R) is selected. Mathematically, this is expressed as

$$\Delta i(s,t_p) = i_p(t_p) - p_L i(t_L) - p_R i(t_R)$$

where i is the impurity, s is the candidate split value, and p_L and p_R are the fractions of the objects in the left and right daughter groups, respectively.[10,12]

For classification trees, the impurity can be defined by different split criteria.[10] The three commonly used split criteria are the Gini index, the twoing index, and the information index. In this work, CART models were built using both the Gini and the information indices. The twoing index was not used because this measure is not useful in two-class classification problems.

Consider $j = 1, 2, 3, ..., k$ the number of classes of the categorical response variable and $p_j(t)$ the probability of correct classification for class j at node t. The Gini index is then defined as

$$\Delta i = 1 - \sum_{j=1}^{k} [p_j(t)]^2$$

and the information index as

$$\Delta i = \sum_{j=1}^{k} - p_j(t) \ln p_j(t)$$

2.1.2. Tree Pruning. The obtained maximum tree usually shows overfitting; therefore, the model is pruned by successively cutting terminal branches. This procedure results in a series of smaller subtrees derived from the maximum tree. The different subtrees are then compared to find the optimal. This comparison is based on a cost-complexity measure $R_\alpha(T)$, in which both tree accuracy and complexity are considered.[10,12] For each subtree T, it is defined as

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}|$$

with $R(T)$ the average within-node sum of squares, $|\tilde{T}|$ the tree complexity, defined as the total number of nodes of the subtree, and α the complexity parameter, which is a penalty for each additional terminal node. During the pruning procedure, α is gradually increased from 0 to 1, and for each value of α, the tree is selected which minimizes $R_\alpha(T)$. For a value of α equal to zero, $R_\alpha(T)$ is minimized by the maximum tree. By gradually increasing α, a series of trees with decreasing complexity is obtained.[10,12]

2.1.3. Selection of the Optimal Tree. From the obtained sequence of subtrees, the optimal has to be selected. The selection is usually based on the evaluation of the predictive error of the models using a cross-validation procedure. Most commonly, a 10-fold cross-validation procedure[21] is used. The predictive error is then given as the overall misclassification rate for each of the subtrees.[10] The optimal model is the simplest model, with a predictive error within one standard error (SE) of the minimal predictive error. This rule, generally referred to as the one SE-rule, allows the selection of a less-complex model than the one with the minimal misclassification rate, without a significant loss of information and accuracy.[10]

2.2. Boosting. Boosting is a technique that uses a set of models for prediction. Boosting was first proposed by Schapire.[11] In fact, all types of chemometrical models can be used as learners in boosting. Most commonly, boosting is used with so-called weak learners. In general, a learner refers to a decision rule and a weak learner is a learner that performs better than random but does not perform as desired. Another characteristic of a weak learner is that predictions are unstable; this means that the models, and as consequence the predictions, change with small changes in the training set. In the first step, a model is built following the normal procedure of the selected weak learner. In the second step, boosting is applied, resulting in a second model focusing on the objects that were wrongly predicted in the first model. The boosting algorithm is applied iteratively, resulting in a set of n models, where n is the number of applied iterations. Predictions of new objects are based on the set of n

models.[19,22] In this paper, the weak learners are the classification trees obtained with CART.

The most frequently used boosting method is called Adaboost (adaptive boosting).[19] In this study, two variants of the Adaboost algorithm, discrete Adaboost[16] and real Adaboost,[16,19] were evaluated and compared. The main difference between the algorithms is that real Adaboost takes into account that different terminal nodes in one tree can have different accuracies, while discrete Adaboost considers only the accuracy of the complete tree. This means that the confidence of a prediction in real AdaBoost is based on the accuracies of the nodes of the different trees in which the object is predicted, while in discrete AdaBoost, it is only based on the accuracies of the different trees in the boosting model.

2.2.1. The Discrete Adaboost Algorithm with CART.[16] This algorithm performs as follows. Consider a training set of n objects belonging to two classes, indicated with the class labels -1 and 1.

Step 1. In the first step, equal weights, w_i, are assigned to each of the n objects in the training set

$$w_i^1 = 1/n \qquad i = 1,2,...n$$

and a classification tree $f^1(x)$ is built.

Step 2. In the next step, boosting is applied in an iterative way. For each iteration $t = 1, 2, ..., T$, the following is performed:

Step 2a. A subset of M samples is selected from the original training set, using a bootstrap resampling method with replacement.[15] The chance that an object is selected for the new data set is dependent on its weight w. An object with a higher weight has a higher probability to be selected.

Step 2b. A new classification tree $f^t(x)$ is built for the resampled data set.

Step 2c. The obtained tree $f^t(x)$ is applied to the original training set. If a sample is misclassified, its error $\mathrm{err}_i^t = 1$; otherwise, it is 0.

Step 2d. The sum of the weighted errors err^t of all of the training objects is calculated:

$$\mathrm{err}^t = \sum_i w_i^t \mathrm{err}_i^t$$

The confidence index of the learner $f^t(x)$ is calculated as

$$c^t = \log[(1 - \mathrm{err}^t)/\mathrm{err}^t]$$

A low err^t results in a high confidence index for learner $f^t(x)$. Sometimes, err^t becomes equal to 0 or 1. In this case, c^t cannot be calculated. Therefore, err^t is set equal to 0.995 when it is larger than 0.995 and equal to 0.005 when it is smaller than 0.005. The purpose is to avoid a situation where one tree with an extremely high c^t dominates the predictions.

Step 2e. The weights of the objects in the training set are updated as follows:

$$w_i^{t+1} = w_i^t \exp(c^t \mathrm{err}_i^t) \qquad i = 1, 2, ..., n$$

The weights of the correctly classified objects remain unchanged while those of the misclassified objects are increased.

Step 2f. The weights are renormalized, and the different steps are repeated in the next iteration.

Step 3. After t iterations, the boosting model consists of a set of t classification trees $f^t(x)$. A new object is predicted with each of the learners in the set. The final prediction is given as

$$y_i = \sum_t c^t f^t(x_j)$$

The higher the confidence index of a learner $f^t(x)$, the higher its role in the final decision. When $y_i < 0$, the object is allocated to class -1, and if $y_i \geq 0$, the object is allocated to class 1.

2.2.2. The Real Adaboost Algorithm with CART.[16,19] Consider the same training set as that described in section 2.2.1. Steps 1, 2a, and 2b are the same as those in the discrete Adaboost algorithm. Then, the algorithm is as follows:

Step 2c. For each terminal node h of $f^t(x)$, the probability is calculated that the objects allocated to this node belong to class -1.

$$p_h^t(x) = \frac{\sum_{y_i=-1} w_i^t}{\sum_{y_i=1} w_i^t + \sum_{y_i=-1} w_i^t}$$

The confidence index c_h^t of node h is defined as

$$c_h^t(x) = \frac{1}{2}\log\left(\frac{p_h^t}{1 - p_h^t}\right)$$

The sign of the confidence index determines the classification. A positive value for c_h^t indicates that the considered object is predicted as belonging to class 1, and a negative value indicates that it is not and, thus, belongs to class -1. The absolute value of the confidence index measures the confidence of the prediction. The higher $|c_h^t|$, the higher the confidence.

Step 2d. The weights of all original training objects are updated as follows:

$$w_i^{t+1} = w_i^t \exp[-y_i\, c^t(x_i)] \qquad i = 1, 2, ..., n$$

In this way, the weights of misclassified objects are increased, and those for correctly classified objects are decreased.

Step 2e. The weights are renormalized, and the different steps are repeated in the next iteration.

3. After t iterations, the boosting model consists of a set of t classification trees $f^t(x)$. The prediction for a new object j is given as

$$y_i = \sum_{t=1}^{T} c^t(x_j)$$

The sign indicates the class prediction, and the absolute value of y_i gives the confidence of that prediction.

2.3. Theoretical Molecular Descriptors. A theoretical descriptor is the final result of a logical and mathematical procedure, which converts the chemical information from a symbolic representation of the molecule into a useful

Table 1. The 147 Molecules and Their Class Labels, Extracted from Crivori et al.[6]

no.	substance	class	no.	substance	class	no.	substance	class
1	a56726	−1	50	fexofenadine	−1	99	naltrexone	1
2	a60616	−1	51	fleroxacin	−1	100	nordiazepam	1
3	alprazolam	1	52	flupentixol	1	101	norfloxacin	−1
4	apomorphine	1	53	furosemide	−1	102	ofloxacin	−1
5	asimadoline	−1	54	GR45809	1	103	oxazepam	1
6	astemizole	−1	55	GR85571	1	104	pefloxacin	−1
7	atenolol	−1	56	GR88377	1	105	perphenazine	1
8	bisL663581	−1	57	GR89696	1	106	pirenzepine	−1
9	BRL52537	1	58	GR89696et	1	107	piroxicam	−1
10	BRL52580	1	59	GR89696pr	1	108	progesteron	1
11	BRL52656	1	60	GR91272	1	109	promazine	1
12	BRL52871	1	61	GR94839	−1	110	promethazine	1
13	BRL52974	−1	62	GR94839A	−1	111	ranitidine	−1
14	BRL53080	1	63	GR94839B	−1	112	rivastigmine	1
15	BRL53087	1	64	GR94839C	−1	113	roxindole	1
16	caffeine	1	65	GR94839D	−1	114	RP60180	1
17	carbidopa	−1	66	GR94839E	−1	115	rufloxacin	−1
20	carebastine	−1	67	GR94839F	−1	116	salbutamol	−1
19	carmoxirol	−1	68	GR94839G	−1	117	sankyo	1
20	cetirizine	−1	69	GR94839H	−1	118	SB201708	1
21	chlorpromazine	1	70	GR94839I	−1	119	SB204454	−1
22	cimetidine	−1	71	GR94839L	−1	120	SB204457	−1
23	ciprofloxacin	−1	72	haloperidol	1	121	SB204459	−1
24	clobazam	1	73	ICI197067	1	122	SB204484	1
25	clonidine	1	74	ICI199441	1	123	SB205563	−1
26	corticosterone	−1	75	ICI204448	−1	124	SB205605	−1
27	cortisol	−1	76	ICI205640	−1	125	skba	1
28	cp102	−1	77	icotidine	−1	126	skbb	1
29	cp107	−1	78	imipramine	1	127	skbc	1
30	cp20	1	79	isoxicam	−1	128	skbd	−1
31	cp21	1	80	L364718	1	129	skbe	−1
32	cp24	1	81	L365260	1	130	skbf	−1
33	cp25	1	82	L663581	1	131	skbg	−1
34	cp29	1	83	levallorphan	1	132	skbh	−1
35	cp41	−1	84	levodopa	−1	133	skbi	−1
36	cp94	1	85	lomefloxacin	−1	134	sparfloxacin	−1
37	cyclazocine	1	86	loperamide	−1	135	tamitinol	1
38	descaloratidine	−1	87	loratidine	−1	136	temelastine	−1
39	desipramine	1	88	lupitidine	−1	137	tenoxicam	−1
40	diazepam	1	89	M3G	−1	138	terfenadine	−1
41	difloxacin	−1	90	M6G	−1	139	testosteron	1
42	diphenhydramine	1	91	mefloquine	−1	140	thiopental	1
43	domperidone	−1	92	meloxicam	−1	141	thioridazine	1
44	dopamine	−1	93	mepyramine	1	142	tifluadom	1
45	doxlamine	1	94	mequitazine	−1	143	tiotidine	−1
46	ebastine	−1	95	meta-mefloquine	−1	144	tiotidine-der	−1
47	EMD60400	−1	96	monoL663581	−1	145	U50488	1
48	enoxacin	−1	97	morphine	1	146	zolantidine	1
49	estradiol	1	98	nalorphine	1	147	zolantidine-der	1

numerical value.[23] Several thousands of molecular descriptors are already proposed in the literature, and the number is still growing. More information about molecular descriptors and their classification can be found in the work of Todeschini and Consonni.[23]

3. MATERIALS AND METHODS

3.1. Data. The used data set is extracted from Crivori et al.[6] Stereoisomers were not considered. This resulted in a data set consisting of 147 molecules classified into two classes, passing (+) and not passing (−) the blood−brain barrier. Compounds with a log BB higher than zero were classified as passing the blood−brain barrier[6] and those with a value below zero as not passing. Table 1 shows the different molecules and their class labels.

3.2. Three-Dimensional Structure Optimization. The 3D structures of the molecules were calculated using the freeware Marvinsketch 3.5.6, academic version (ChemAxon Ltd., Budapest, Hungary). After introduction of the molecule as a topological structure, geometry optimization was performed, resulting in a data matrix consisting of the Cartesian coordinates of the atoms. This data matrix is then used to calculate the molecular descriptors.

3.3. Calculating Molecular Descriptors. Molecular descriptors were calculated using the Dragon 4.0 professional software.[24] This program allows calculating 48 constitutional descriptors, 119 topological descriptors, 47 walk and path counts, 33 connectivity indices, 47 information indices, 96 2D autocorrelations, 107 edge adjacency indices, 64 BCUT descriptors, 21 topological charge indices, 44 eigenvalue-based indices, 41 randic molecular profiles, 74 geometrical descriptors, 150 RDF descriptors, 160 3D-MoRSE descriptors, 99 WHIM descriptors, 197 GETAWAY descriptors, 121 functional group counts, 120 atom-centered fragments, 14 charge descriptors, and 28 molecular properties. The software automatically eliminates constant variables in a given data

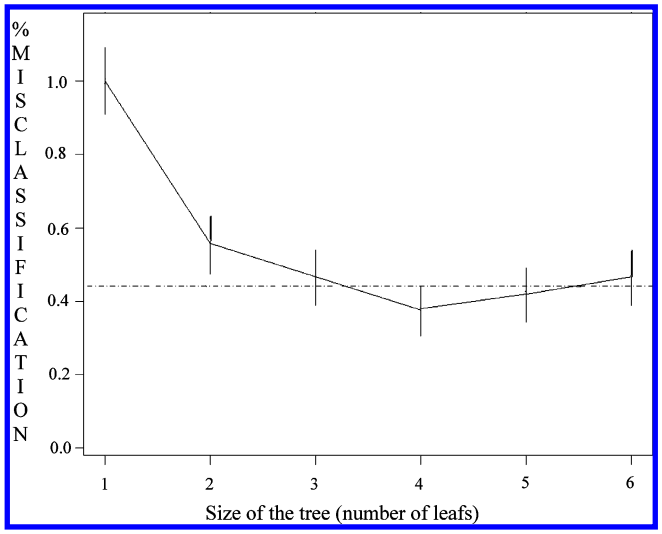

Figure 2. Tree complexity as a function of the percentage of misclassification using the Gini index as a split criterion. Vertical lines represent the standard deviation around the percentage of misclassification for the different tree complexities. The horizontal line represents the minimal percentage of misclassification plus one standard error.

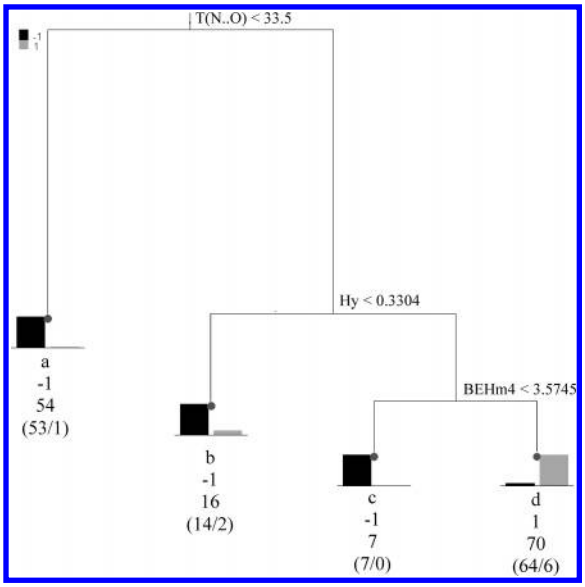

Figure 3. Best classification tree obtained with the Gini index as a split criterion. Classes (a−d) are identified by the label of the class of the majority of the molecules, the number of molecules in the class, and the ratio of correctly classified over misclassified objects.

set. For descriptors with a correlation higher than 0.98, only one is retained. More information about these descriptors can be found in the work of Todeschini and Consonni.[23] The software also performs a PCA[21] for each class of calculated descriptors. The scores on the significant principal components are added to the descriptor set. Furthermore, the Hyperchem 6.03 professional software (Hypercube, Gainesville, Florida) was used to calculate solvent-accessible surface area, molecular volume, octanol/water partition coefficient (log *P*), hydration energy, molar refractivity, molar polarizability, and molar mass.[12] The McGowans volume, a descriptor used in the linear free energy relationship of Abraham, was calculated manually.[23,25] On the complete set of calculated descriptors, a PCA[21] was performed. The scores on the principal components, which describe more than 1% of the information, are also used as descriptors.

3.4. Building CART and Boosting CART Models. The models were built using in-house algorithms written in Matlab 6.0. Programming was done according to the original CART algorithm proposed by Breiman[10] and the boosting algorithms described by Friedman et al.[19]

4. RESULTS AND DISCUSSION

4.1. Classification And Regression Trees (CART). *4.1.1. Building Tree Models.* The models were built using the class labels of all 147 molecules as response variables and the calculated descriptors as explanatory variables. During the building process, the maximum tree is built and pruned. In the next step, a 10-fold cross validation is carried out, resulting in a graph of the percentage of misclassification as a function of the tree complexity. This procedure is carried out twice using, respectively, the Gini and the information indices as split criteria. When using the Gini index, the graph of the percentage of misclassification as a function of the tree complexity (Figure 2) shows that the least misclassification is obtained at tree size 4.

A good classification (Figure 3) of the molecules in the training set is obtained.

Class a was labeled as −1. It contains one molecule (L663581) belonging to class 1, which is thus misclassified. Class b contains two misclassified molecules (clonidine and tamitinol), belonging to class 1. All molecules in class c belong to class −1. In class d, six molecules (BRL52974, carmoxirol, corticosterone, descaloratidine, loratidine, and metamofliquine) are misclassified. So, for the complete tree, only 9 molecules of 147 are misclassified. This tree shows a mean cross-validation error of 19.4%, calculated after 10 repetitions of the cross-validation procedure. This means that 80.6% of the data set is predicted correctly when using this model.

With the information index as a split criterion, a tree with size 6 could be selected on the basis of the graph of the percentage of misclassification as a function of the tree complexity (Figure 4).

Again, a good classification (Figure 5) of the molecules of the training set is obtained.

Class a, labeled as class −1, contains one misclassified molecule (L663581). Class b contains two misclassified molecules (clonidine and tamitinol) and class c one (zolantidine-der). All molecules classified in classes d, e, and f belong to 1. In this tree, 4 molecules of the 147 are misclassified. This tree shows a mean cross-validation error of 16.4%, or 83.6% of the data set is predicted correctly.

The best tree is obtained with the information index as a split criterion, because in this tree fewer molecules (four) of the training set are misclassified than in that obtained with the Gini index (nine molecules), while the percentage of misclassification also is lower, 16.4% versus 19.4%.

4.1.2. The Selected Descriptors. The two first splits of the trees obtained with both the Gini and the information indices as split criteria are defined by the same descriptors; that is, T(N··O) and Hy. T(N··O) stands for the sum of topological distances between nitrogen and oxygen atoms. This descriptor is a topological descriptor and describes properties related to the calculation of the polar surface area (PSA).[23,26] Hy represents the hydrophilicity index and is a measure for the

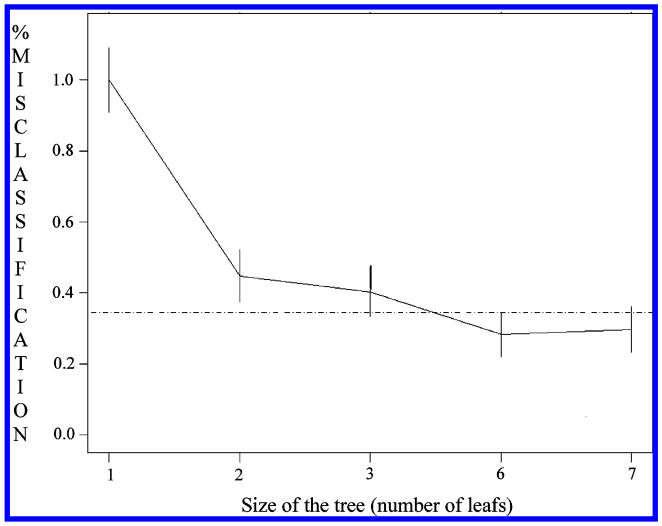

Figure 4. Tree complexity as a function of the percentage of misclassification using the information index as a split criterion. Vertical lines represent the standard deviation around the percentage of misclassification for the different tree complexities. The horizontal line represents the minimal percentage of misclassification plus one standard error.

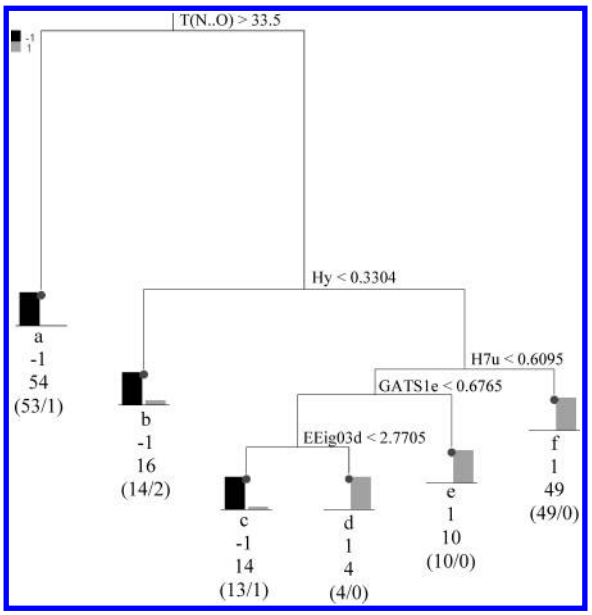

Figure 5. Best classification tree obtained with the information index as a split criterion. Classes (a−f) are identified by the label of the class of the majority of the molecules, the number of molecules in the class, and the ratio of correctly classified over misclassified objects.

hydrophilicity/lipophilicity of the molecules. Hy is related to the *n*-octanol−water partition coefficient, log *P*.[23] Both PSA as log *P* are known to be key properties determining membrane passage and, specifically, the passage of molecules through the blood−brain barrier.[27−30] In the tree obtained with the Gini index (Figure 3), the third and final split is defined by BEHm4 (highest eigenvalue *n* 4 of the Burden matrix, weighted by atomic masses). This descriptor belongs to the Burden−CAS−University of Texas eigenvalue descriptors. This group of descriptors gives information about atomic charges, atomic polarizability, and atomic hydrogen-binding properties.[23] These properties can also be related to log *P* and PSA. The descriptors defining splits 3, 4, and 5 in the tree obtained with the information index (Figure 5) are the GETAWAY descriptor H7u (H-autocorrelation of lag 7, unweighted), the 2D-autocorrelation descriptor GATS1e (Geary autocorrelation of lag 1, weighted by atomic Sanderson electronegativities), and the edge adjacency index EEig03d (eigenvalue 03 from the edge adjacency matrix weighted by dipole moments), respectively.[23] These descriptors are related to the topological (GATS1e and EEig03d) and the geometrical (H7u) structures of the molecules. In general, the molecular structure has an important role in membrane passage processes and in the passage through the blood−brain barrier. However, when using theoretical descriptors as in our approach, it is not always evident to directly link the descriptor to a physicochemical property of a molecule.

Table 2. First Test Set of the Manual 20-Bootstrap with Replacement Method Using 15 Molecules and Their Class Labels

no.	substance	class
8	bisL663581	−1
16	caffeine	1
21	chlorpromazine	1
23	ciprofloxacin	−1
42	diphenhydramine	1
43	domperidone	−1
48	enoxacin	−1
59	GR89696pr	1
61	GR94839	−1
79	isoxicam	−1
81	L365260	1
113	roxindole	1
116	salbutamol	−1
135	tamitinol	1
145	U50488	1

4.2. Boosting CART. From the above, it was concluded that classification trees using the information index as a split criterion give the best results for this data set. Therefore, the information index was chosen as a split criterion for the classification trees, used as weak learners in the boosting algorithms. To evaluate the predictive abilities of the boosting models, a manual 20-bootstrap with replacement[20] method was carried out. The data set was divided into a test set of 15 molecules and a training set of 132 molecules, on the basis of a plot of the first two principal components of the whole data set. This procedure was repeated 20 times. The boosting algorithms were applied 20 times, each time with different test and training sets. The number of iterations is set to 150, which means that the obtained boosting models will consist of sets of 150 classification trees. The mean predictive error over the 20 models is calculated. To compare the predictive abilities of the boosting models with those of a single classification tree, classification trees were built for each training set and each time the corresponding test set was predicted. The mean predictive error over the 20 models was calculated as seen in theory.

4.2.1. Building the Models. Consider a first test set, consisting of 15 molecules (Table 2), where seven belong to class −1 and eight to class 1, and the corresponding training set, consisting of the remaining 132 molecules.

When a single classification tree (Figure 6) is built with the selected training set and the test set is predicted, 80.0% of the molecules in the test set are predicted correctly.

In the next step, discrete Adaboost is applied. Figure 7 shows the improvement in predictive abilities of the model for the selected test set as a function of the number of iterations. The predictive power increases rapidly to finally

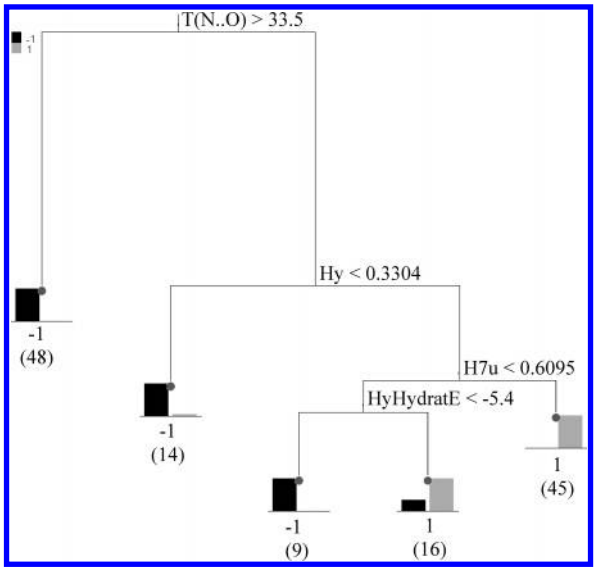

Figure 6. Classification tree obtained with the information index as a split criterion and used for prediction of the first test set of the 20-bootstrap with replacement method. Classes are identified by the label of the majority of the molecules and the number of molecules in the class.

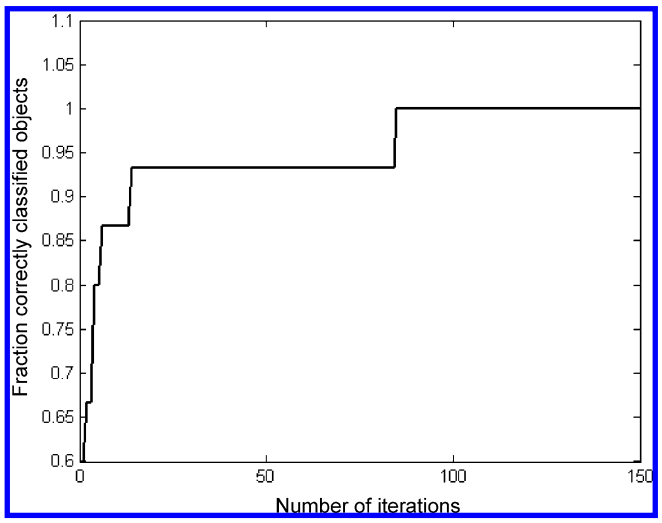

Figure 7. Fraction of correctly classified objects as a function of the number of iterations, obtained with the discrete Adaboost algorithm for the first test set of the 20-bootstrap with replacement method.

come to a plateau, where all molecules of the test set are predicted correctly.

The same procedure is repeated using the real Adaboost algorithm. Figure 8 shows the improvement of the predictive abilities as a function of the number of iterations. Again, a rapid improvement can be seen to come to a plateau, where 93.3% of the test set is predicted correctly.

For this first test set, the application of both algorithms results in a significant improvement in predictive ability compared to a single classification tree; though, a higher improvement can be seen when using the discrete Adaboost algorithm.

Application of the discrete and real Adaboost algorithms to each of the 20 selected test and training sets resulted in a mean percentage of correctly classified molecules of 94.0% for the discrete Adaboost algorithm and 93.9% for the real Adaboost algorithm, compared to 80.6% for the single classification tree. Both algorithms result in models with the same predictive abilities.

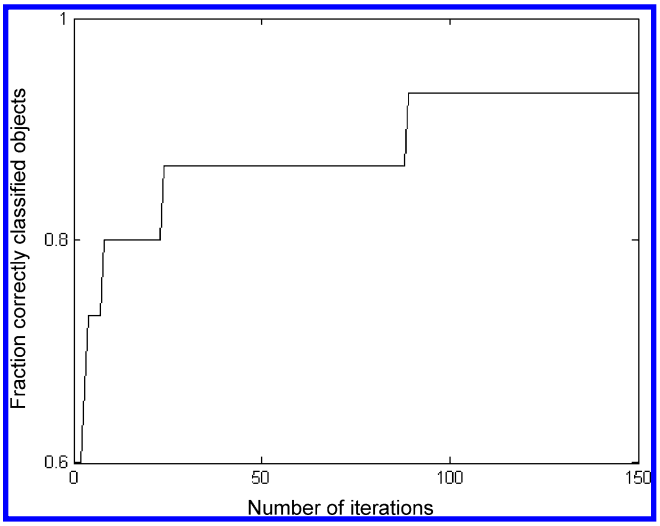

Figure 8. Fraction of correctly classified objects as a function of the number of iterations, obtained with the real Adaboost algorithm for the first test set of the 20-bootstrap with replacement method.

Table 3. The 45 Molecules of the Selected External Test Set

no.	substance	class
4	apomorphine	1
5	asimadoline	−1
7	atenolol	−1
10	BRL52580	1
15	BRL53087	1
18	carebastine	−1
21	chorpromazine	1
22	cimetidine	−1
25	clonidine	1
29	cp107	−1
31	cp21	1
40	diazepam	1
44	dopamine	−1
46	ebastine	−1
51	fleroxacin	−1
52	flupentixol	1
58	GR89696et	1
68	GR94839G	−1
72	haloperidol	1
73	ICI197067	1
78	imipramine	1
81	L365260	1
83	levallorphan	1
89	M3G	−1
91	mefloquine	−1
92	meloxicam	−1
93	mepyramine	1
94	mequitazine	−1
95	metabmefloquine	−1
100	nordazepam	1
101	norfloxacin	−1
106	pirenzepine	−1
107	piroxicam	−1
108	progesterone	1
110	promethazine	1
111	ranitidine	−1
113	roxindole	1
114	RP60180	1
116	salbutamol	−1
128	skbd	−1
130	skbf	−1
131	skbg	−1
135	tamitinol	1
142	tifluadom	1
143	tiotidine	−1

Table 4. The 20 Most Important Descriptors and Their Variable Importance for the Model Built with Discrete Adaboost (A) and Real Adaboost (B)

no.	descriptor	definition	importance
		A. Discrete Adaboost	
1	PC5	fifth principal component calculated on the whole set of descriptors	6.24
2	Hy	hydrophilicity index	6.19
3	T(N···O)	sum of topological distances between nitrogen and oxygen atoms	5.69
4	PC06-13	13th principal component calculated for the 2D autocorrelations	4.42
5	PC12-08	eighth principal component calculated for the geometrical descriptors	4.24
6	HyHydratE	The hydration energy calculated with Hyperchem	3.62
7	nHDon	number of donor atoms for H-bond (with N and O)	3.44
8	PC16-16	16th principal component calculated for the GETAWAY descriptors	3.33
9	PC12-11	11th principal component calculated for the geometrical descriptors	3.23
10	H7u	H autocorrelation of lag 7/ unweighted	3.21
11	IC1	information content index (neighborhood symmetry of 1-order)	3.08
12	PC16-11	11th principal component calculated for the GETAWAY descriptors	2.80
13	PC16-10	10th principal component calculated for the GETAWAY descriptors	2.78
14	PC13-11	11th principal component calculated for the RDF descriptors	2.76
15	T(O···O)	sum of topological distances between nitrogen and oxygen atoms	2.71
16	Ss	sum of Kier-Hall electrotopological states	2.70
17	PC05-01	first principal component calculated for the information indices	2.68
18	log *P*	logarithm of the *n*-octanol/water partition coefficient	2.63
19	PC08-04	fourth principal component calculated for the BCUT descriptors	2.62
20	PC14-08	eighth principal component calculated for the 3D-MoRSE descriptors	2.62
		B. Real Adaboost	
1	Hy	hydrophilicity index	5.95
2	PC5	fifth principal component calculated on the whole set of descriptors	5.69
3	T(N···O)	sum of topological distances between nitrogen and oxygen atoms	5.17
4	PC06-13	13th principal component calculated for the 2D autocorrelations	4.52
5	IC2	information content index (neighborhood symmetry of 2-order)	4.24
6	HyHydratE	the hydration energy calculated with Hyperchem	3.76
7	PC16-16	16th principal component calculated for the GETAWAY descriptors	3.71
8	Seigv	eigenvalue sum from van der Waals weighted distance matrix	3.54
9	H7u	H autocorrelation of lag 7/ unweighted	3.50
10	IC1	information content index (neighborhood symmetry of 1-order)	3.40
11	DISPe	d COMMA2 value/ weighted by atomic Sanderson electronegativities	3.38
12	PC13-11	11th principal component calculated for the RDF descriptors	3.22
13	PC16-11	11th principal component calculated for the GETAWAY descriptors	3.19
14	T(O···O)	sum of topological distances between nitrogen and oxygen atoms	3.18
15	PC05-01	first principal component calculated for the information indices	3.14
16	PC12-11	11th principal component calculated for the geometrical descriptors	3.05
17	E2s	second component accessibility directionam WHIM index weighted by electrotopological states	3.04
18	PC16-10	10th principal component calculated for the GETAWAY descriptors	3.00
19	Ds	D total accessibility index/ weighted by electrotopological states	2.87
20	nHDon	number of donor atoms for H-bond (with N and O)	2.79

To have a more precise idea of the predictive abilities of this type of model, the original data set was randomly redivided into a test set of 45 molecules (Table 3) and a training set of 102 molecules.

New models were built for the training set, and the test set was used as an external test set. With a single classification tree, 84.4% of the molecules in the external test set are predicted correctly. When the boosting algorithms are applied, an improvement is obtained to 91.1% for discrete and 88.9% for real Adaboost.

In general, it can be concluded that, for the used data set, both algorithms result in a significant improvement of the predictive abilities compared to a single classification tree and slightly better results are obtained using the discrete Adaboost algorithm.

4.2.2. The Selected Descriptors. The importance of the explanatory variables in the boosting CART models were evaluated according to the variable ranking method proposed by Breiman.[10] The importance of a variable (TI) in a CART learner is defined by the decrease of impurity across the tree for all nonterminal nodes that use this variable as a split variable. The importance of the variable in boosting (BI) is given by the weighted average of its importance over the whole set of classification trees defining the boosting model.[16,31] For a variable x_i, the importance is given as

$$\mathrm{BI}(x_i) = \frac{1}{T}\sum_{t=1}^{T}\{c^t\ \mathrm{TI}[x_i, f^t(x)]\}$$

with T the number of iterations and c^t the confidence index of the learner or tree $f^t(x)$.

Table 4 shows the 20 most important variables in the boosting models obtained with discrete and real Adaboost. The variables are ranked in decreasing order of importance.

The selection of a number of these descriptors can easily be explained. One of the most important descriptors in both models is PC5, which is to be expected because principal components should contain more information than individual variables.[21] The fact that PC5 is selected and not PC1 can be explained by the fact that CART and boosting CART are local modeling techniques. In certain regions of the data space, higher principal components can be more important than the first. Hy and log *P* are related to the hydrophilicity/lipophilicity properties of molecules,[23] which as previously

mentioned are key properties for membrane passage.[27−29] T(N··O), nHDon, and T(O··O) describe properties related to the calculation of the PSA,[23,26] also an important property in membrane passage.[29,30]

It can be noticed that the majority of the selected descriptors are principal components calculated for the different classes of Dragon descriptors. Principal components for six descriptor classes are selected, from which two (2D autocorrelations and information indices) can be related to the topological structure of the molecules and four (GETAWAY, RDF, geometrical, and 3D MoRSE descriptors) to the 3D structure. The topology and geometry of a molecule is also important in determining whether it will pass biological membranes and especially the passage through the blood−brain barrier because, here, membrane transport is more specific. The remaining descriptors are all theoretically calculated variables that can somehow be related to hydrogen bonding and PSA (HyHydratE and H7u) or to the topological (IC1, IC2, Seigv, DISPe, Ds and Ss) or geometrical structure of the molecule (H7u and E2s) but that are not easy to interpret in relation to physicochemical processes such as membrane passage.

5. CONCLUSIONS

For the three types of models presented, very good predictive abilities were obtained. Good results were obtained with single classification trees. The model built with the information index as a split criterion has slightly better predictive abilities than that obtained with the Gini index. Although the percentage of correctly classified objects with the best tree is only 83.6%, a single tree can be very useful in QSAR, because the model is very simple and easy to interpret.

The use of boosting improves the predictive abilities of the models compared to a single classification tree. For this data set, the best model is obtained with discrete Adaboost. With this model, 91.1% of an external test set is predicted correctly. The results of the manual 20-bootstrap with replacement method showed a predictive ability of 94%.

When the obtained models are compared to the original model presented by Crivori et al.[6] for the same data set, it can be concluded that both the single CART and boosted CART models show higher predictive abilities. The PCA model obtained by Crivori has an overall predictive ability of 74.8% and is less transparent and, thus, more difficult to interpret than the tree-based models presented.

CART and boosting CART seem able to select the best descriptors to describe the processes of membrane passage and, more specifically, the passage through the blood−brain barrier, because the majority of the selected descriptors can easily be related to the physicochemical processes and molecular properties that play a major part in membrane passage. The use of PCA scores as extra descriptors allows the input of more information into the model, eventually resulting in better models.

Generally, it can be concluded that the use of classification trees, as a single model or as learner in a boosting model, can be a very useful tool in QSAR modeling, resulting in easily interpretable models with high predictive abilities.

ACKNOWLEDGMENT

The authors thank the late Prof. D. L. Massart for providing the possibility to start the Ph.D. study and for supporting the project. This research is financed with a specialization grant from the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT).

REFERENCES AND NOTES

(1) Kaliszan, R.; Markuszewski, M. Brain/blood distribution described by a combination of partition coefficient and molecular mass. *Int. J. Pharm.* **1996**, *145*, 9−16.
(2) Clark, D. E. Rapid calculation of polar molecular surface area and its application to the prediction of transport phenomena. 2. Prediction of blood−brain barrier penetration. *J. Pharm. Sci.* **1999**, *88*, 815−821.
(3) Platts, J. A.; Abraham, M. H.; Zhao, Y. H.; Hersey, A.; Ijaz, L.; Butina, D. Correlation and prediction of a large blood−brain distribution data set-an LFER study. *Eur. J. Med. Chem.* **2001**, *36*, 719−730.
(4) Narayanan, R.; Gunturi, S. B. In silico ADME modelling: prediction models for blood−brain barrier permeation using a systematic variable selection method. *Bioorg. Med. Chem.* **2005**, *13*, 3017−3028.
(5) Norinder, U.; Sjöberg, P.; Österberg, T. Theoretical calculation and prediction of brain−blood partitioning of organic solutes using molsurf parametrization and PLS statistics. *J. Pharm. Sci.* **1998**, *87*, 952−958.
(6) Crivori, P.; Cruciani, G.; Carrupt, P. A.; Testa, B. Predicting blood−brain barrier permeation from three-dimensional molecular structure. *J. Med. Chem.* **2000**, *43*, 2204−2216.
(7) Subramanian, G.; Kitchen, D. B. Computational models to predict blood−brain barrier permeation and CNS activity. *J. Comput.-Aided Mol. Des.* **2003**, *17*, 643−664.
(8) Dorronsoro, I.; Chana, A.; Inés Abasolo, M.; Gil, C.; Stud, M.; Martinez, A. CODES/ neural network model: a useful tool for *in silico* prediction of oral absorption and blood−brain barrier permeability of structurally diverse drugs. *QSAR Comb. Sci.* **2004**, *23*, 89−98.
(9) Hemmateenejad, B. Correlation ranking procedure for factor selection in PC-ANN modeling and application to ADMETox evaluation. *Chemom. Intell. Lab. Syst.* **2005**, *75*, 231−245.
(10) Breiman, L.; Friedman, J. H.; Olshen, R. A.; Stone, C. J. *Classification and Regression Trees*; Wadsworth & Brooks: Monterey, CA, 1984.
(11) Schapire, R. E. The strength of weak learnability. *Mach. Learning* **1990**, 5, 197−227.
(12) Deconinck, E.; Hancock, T.; Coomans, D.; Massart, D. L.; Vander Heyden, Y. Classification of drugs in absorption classes using classification and regression trees (CART)-methodology. *J. Pharm. Biomed. Anal.* **2005**, *39*, 91−103.
(13) Bai, J. P. F.; Utis, A.; Crippen, G.; He, H.; Fischer, V.; Tullman, R.; Yin, H.; Hsu, C.; Jiang, L.; Hwang, K. Use of classification regression tree in predicting oral absorption in humans. *J. Chem. Inf. Comput. Sci* **2004**, *44*, 2061−2069.
(14) Put, R.; Perrin, C.; Questier, F.; Coomans, D.; Massart, D. L.; Vander Heyden, Y. Classification and regression tree analysis for molecular descriptor selection and retention prediction in chromatographic structure−retention relationship studies. *J. Chromatogr., A* **2003**, *988*, 261−276.
(15) Caetano, S.; Aires-de-Sousa, J.; Daszykowski, M.; Vander Heyden, Y. Prediction of enantioselectivity using chirality codes and classification and regression trees. *Anal. Chim. Acta* **2005**, *544*, 315−326.
(16) Zhang, M. H.; Xu, Q. S.; Daeyaert, F.; Lewi, P. J.; Massart, D. L. Application of boosting to classification problems in chemometrics. *Anal. Chim. Acta* **2005**, *544*, 167−176.
(17) 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*, 39−47.
(18) He, P.; Fang, K. T.; Liang, Y. Z.; Li, B. Y. A generalized boosting algorithm and its application to two-class chemical classification problem. *Anal. Chim. Acta* **2005**, *54*, 181−191.
(19) Friedman, J.; Hastie, T.; Tibshirani, R. Additive logistic regression: a statistical view of boosting. *Ann. Stat.* **2000**, *28*, 337−407.
(20) Wehrens, R.; Putter, H.; Buydens, L. M. C. The Bootstrap: a tutorial. *Chemom. Intell. Lab. Syst.* **2000**, *54*, 35−52.
(21) Vandeginste, B. G. M.; Massart, D. L.; Buydens, L. M. C.; De Jong, S.; Lewi, P. J.; Smeyers-Verbeke, J. *Handbook of Chemometrics and Qualimetrics—Part B*; Elsevier Science: Amsterdam, 1997.
(22) Varmuza, K.; He, P.; Fang, K. T. Boosting applied to classification of mass spectral data. *J. Data Sci.* **2003**, *1*, 391−404.
(23) Todeschini, R.; Consonni, V. *Handbook of Molecular Descriptors*; Wiley-VCH: Weinheim, Germany, 2000.

(24) Todeschini, R.; Consonni, V.; Mauri, A.; Pavan, M. *Dragon Professional Version*, version 5.0; Milano Chemometrics and QSAR Research Group, Talete SRL: Italy, 2004.
(25) Abraham, M. H.; Ibrahim, A.; Zissimos, A. M.; Zhao, Y. H.; Corner, J.; Reynolds, D. P. Application of hydrogen bonding calculations in property based drug design. *Drug Discovery Today* **2002**, *7*, 1056−1063.
(26) Deconinck, E.; Xu, Q. S.; Put, R.; Coomans, D.; Massart, D. L.; Vander Heyden, Y. Prediction of gastro-intestinal membrane permeability using multivariate adaptive regression splines. *J. Pharm. Biomed. Anal.* **2005**, *39*, 1021−1030.
(27) Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J. Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Adv. Drug Delivery Rev.* **2001**, *46*, 3−26
(28) Poole, S. K.; Poole, C. F. Separation methods for estimating octanol−water partition coefficients. *J. Chromatogr., B* **2003**, *797*, 3−19.
(29) Yang, S.; Bumgarner, J. F.; Kruk, L. F. R.; Khaledi, M. G. Quantitative structure−activity relationships studies with micellar electrokinetic chromatography. Influence of surfactant type and mixedmicelles on estimation of hydrofobicity and bioavailability. *J. Chromatogr., A* **1996**, *721*, 323−335.
(30) Abraham, M. H.; Chadha, H. S.; Leitao, R. A. E.; Mitchell, R. C.; Lambert, W. J.; Kaliszan, R.; Nasal, A.; Haber, P. Determination of solute lipophilicity, as log *P*(octanol) and log *P*(alkane) using poly(styrene-divinylbenzene) and immobilised artificial membrane stationary phases in reversed-phase high-performance liquid chromatography. *J. Chromatogr., A* **1997**, *766*, 35−47.
(31) Put, R.; Van Gyseghem, E.; Coomans, D.; Vander Heyden, Y. The selection of orthogonal reversed-phase HPLC systems by univariate and auto-associative multivariate regression trees. *J. Chromatogr., A* **2005**, *1096*, 187−198.

CI050518S