Journal of Statistical Research *ISSN 0256 - 422 X*
2012, Vol. 46, No. 2, pp. 157-186

THE INTERACTIVE DECISION COMMITTEE FOR CHEMICAL TOXICITY ANALYSIS

CHAERYON KANG

Fred Hutchinson Cancer Research Center, Seattle, WA 98109-1024, USA

Email: ckang2@fhcrc.org

HAO ZHU

Rutgers University, Camden, NJ 08102-1519, USA

Email: hao.zhu99@rutgers.edu

FRED A. WRIGHT, FEI ZOU, AND MICHAEL R. KOSOROK

University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-7420, USA

Email: fwright@bios.unc.edu, fzou@bios.unc.edu, kosorok@email.unc.edu

SUMMARY

We introduce the Interactive Decision Committee method for classification when high-dimensional feature variables are grouped into feature categories. The proposed method uses the interactive relationships among feature categories to build base classifiers which are combined using decision committees. A two-stage or a single-stage 5-fold cross-validation technique is utilized to decide the total number of base classifiers to be combined. The proposed procedure is useful for classifying biochemicals on the basis of toxicity activity, where the feature space consists of chemical descriptors and the responses are binary indicators of toxicity activity. Each descriptor belongs to at least one descriptor category. The support vector machine, the random forests, and the tree-based AdaBoost algorithms are utilized as classifier inducers. Forward selection is used to select the best combinations of the base classifiers given the number of base classifiers. Simulation studies demonstrate that the proposed method outperforms a single large, unaggregated classifier in the presence of interactive feature category information. We applied the proposed method to two toxicity data sets associated with chemical compounds. For these data sets, the proposed method improved classification performance for the majority of outcomes compared to a single large, unaggregated classifier.

Keywords and phrases: Chemical toxicity; Decision committee method; Ensemble; Ensemble feature selection; QSAR modeling; Statistical learning

1 Introduction

The assessment of potential toxicity associated with drugs and commercial chemicals is an important topic in medicinal chemistry and toxicology. Standard toxicity assessment requires *in vivo* testing in animals, which is expensive, time consuming, and raises ethical concerns. For these reasons, only a small fraction of commercial chemicals have been tested extensively. Thus, there is increasing interest in developing models for accurate toxicity prediction, to better prioritize chemicals for testing, with an ultimate goal of purely computational toxicity prediction. Quantitative Structure-Activity Relationship (QSAR) modeling is one of the most popular approaches to develop computational toxicity models (Richard, 2006). QSAR approaches model the relationship between chemical structures and target biological activities and the resulting models are used to predict the target biological activities using the chemical descriptors of new compounds. External prediction accuracy is one of the most important issues in QSAR modeling. However, most currently available QSAR toxicity models have relatively low prediction ability for new compounds (Stouch et al., 2003; Johnson, 2008).

The goal of this paper is to develop a new modeling procedure to improve computational models of animal toxicity. We propose an extension of the decision committee method to use functional information of existing feature categories to improve accuracy of the classification model. Simulation studies demonstrate that the proposed method outperforms a single classifier in the presence of interactive association between feature categories. To the best of our knowledge, our paper is the first study to consider the interactive relationship among existing feature categories to combine base classifiers for more accurate classification in the decision committee context. To set the stage for our contribution, we briefly introduce the basic idea of the decision committee method and related issues.

The decision committee method, sometimes called *ensemble* or *classifier fusion*, is known to perform better than a single classifier by integrating multiple base classifiers which are individually trained by a deterministic *inducer* (a mapping from a training sample to a classifier) into the combined classification system (Opitz and Maclin, 1999; Assareh et al., 2008). The basic idea of the decision committee method is that each base classifier can provide complementary information about the pattern to be classified, which may lead to better performance in the classification task (Vale et al., 2008).

In addition, aggregating multiple predictions from different base classifiers can resolve the problem of overtraining (Shin and Markey, 2006). Aggregation is the procedure by which multiple classifiers are combined into a single large classifier. A good choice of the aggregation rule can improve the classification accuracy. There are many different aggregation rules in the decision committee method literature (Clemen, 1989; Ali and Pazzani, 1996; Dietterich, 1997). Researchers have focused on three methods to aggregate base classifiers: selecting the best one (winner takes all), voting for the most popular class, and stacking with some other learning algorithm. The winner takes all strategy selects one best base classifier. Voting for the most popular class takes an average over outputs from the base classifiers with or without weighting, and classifies the examples into the class that has the most votes. Stacking (or stacked generalization), introduced by Wolpert (1992), is a general method of learning with meta-level classifiers using predictions from the base classifiers as inputs (Sigletos et al., 2005). In this study, we combine base classifiers by using either voting (majority vote) or stacking. We discuss aggregation rules in detail later in this paper.

In the decision committee method, *diversity* among base classifiers is one of the key factors to improve classification performance, and can be even more important than the aggregation rule (Assareh et al., 2008). Lam (2000) and Shipp and Kuncheva (2002) characterized diversity by independence among the base classifiers (independency), tendency to make different decisions (orthogonality), and complementary effects (complementarity) among base classifiers. Krogh and Vedelsby (1995) define diversity as disagreement among the base classifiers on feature variables. It is obvious that there would be no accuracy gained by aggregating multiple classifiers which provide identical information about the classification pattern. Diverse classifiers provide varied information for the classification patterns.

One can increase the diversity among base classifiers through resampling individuals as training sets (for example, boosting (Freund and Schapire, 1997) and bagging (Breiman, 1996)), selecting different subsets of feature variables (for example, the random forests (Breiman, 2001) and the random subspace algorithm (Ho, 1998)), or using different types of learning algorithms to build base classifiers. Recent research has improved classification performance further by integrating boosting or bagging with feature selection (for example, (Stefanowski, 2005) and (Assareh et al., 2008)).

In ensemble feature selection, each base classifier is trained based on different subsets of feature variables. See Opitz (1999), Abeel et al. (2009), Vale et al. (2008), and Tuv et al. (2009) for more details. Many recent studies of chemical toxicity have utilized the ensemble feature selection method to develop QSAR models. Budka and Gabrys (2010) applied a ridge regression ensemble in which base classifiers were trained using different feature subsets selected by the “plus-L-takeaway-R” method (van der Heijden et al., 2004). Dutta et al. (2007) proposed an ensemble feature selection method to identify an optimal subset of chemical descriptors based on different types of learning algorithms applied simultaneously. Neither study, however, considered the potential interactive relationship between existing categories of descriptors. Including these two studies, most published articles in the decision committee method literature have focused on finding better aggregation rules or on feature selection using marginal prediction ability (Bauer and Kohavi, 1999; Assareh et al., 2008; Tuv et al., 2009).

When feature variables belong to some informative categories, base classifiers with feature variables belonging to each category as input covariates yield different predictions of outcome due to fundamental differences in the information contained in the variables. Eventually, this increases the diversity among base classifiers. Each category might provide important insight into the data structure by itself (the univariate method) or via association with other categories (the interactive method). It is scientifically reasonable to assume that different feature categories may be interactively associated, and that such relationships could affect the classification task.

In this paper, we propose the interactive decision committee (IDC) method to improve prediction accuracy in binary classification problems when high-dimensional feature variables are grouped into feature categories. The method uses the interactive relationships between existing feature categories to build base classifiers in the decision committee context. This is our first contribution. Our second contribution is to utilize a two-stage 5-fold cross-validation (CV) technique to choose the number of base classifiers to be combined using the Support Vector Machine (SVM) algorithm. This technique reduces problems on overtraining by controlling the size of the decision committee method.

The rest of this article is organized as follows. In Section 2, we provide detailed information on two real toxicity data sets of chemical compounds. Next, we describe the IDC method and setup for the IDC method. This includes a general setting for the decision committee method and aggregation rules, and a brief introduction to the three classification inducers: Support Vector Machines, Random forests, and AdaBoost. We provide simulation studies in Section 5 and numerical results on the toxicity data associated with chemical compounds in Section 6. Finally, we conclude with a discussion of some of the limitations of the proposed method, and further research topics to pursue.

2 Toxicity Data sets

In this study, two sets of toxicity data sets associated with chemical compounds are used for model development. Each data set consists of binary *in vivo* endpoints based on animal experiments.

2.1 ToxRefDB data

Historical animal toxicity data for 320 compounds are stored in the Toxicity Reference Database (ToxRefDB), developed by the National Center for Computational Toxicology in the US Environmental Protection Agency (US EPA) (Martin et al., 2009,?). Up to 78 *in vivo* toxicity endpoints are available for each compound. These *in vivo* toxicity endpoints were based on chronic, sub-chronic, developmental, and reproductive toxicity experiments. We used a subset of the original data for this study, due to the relatively low ratio of active compounds for most animal toxicity testings. Eighteen endpoints with the highest activity ratios were selected for model development. Also, we excluded duplicates, and those compounds that could not be handled by our descriptor generating software. Across the eighteen endpoints, the number of compounds in each endpoint subset ranged from 237 to 249 (Table 1). Toxicity results were coded as 1 (active, toxic), or -1 (inactive, non-toxic).

2.2 Rat LD$_{50}$ data

The acute toxicity data of organic chemical compounds in the rat caused by oral exposure to chemicals, described in Zhu et al. (2009), were utilized. The data consist of 5,917 chemical compounds with toxicity activity, originally collected from different sources (National Library of Medicine database (2008)). The acute toxicity activity presents the median lethal dose of a toxic substance in the negative log scale ($-\log LD_{50}$). LD_{50} is the dose level required to kill 50% of the animals of a tested population. Sedykh et al. (2011) categorized chemical compounds into three activity categories using the acute toxicity guidelines (OECD, 1996; Walum, 1998) and used only two categories for QSAR modeling: 'toxic' compounds $(-\log LD_{50} > 3)$ and 'non-toxic' compounds $(-\log LD_{50} < 2)$. Chemical compounds with $2 \leq -\log LD_{50} \leq 3$ were not used for analysis. Following Sedykh et al. (2011), 3,404 chemical compounds classified in either toxic or non-toxic activity categories are used in our study. Each toxicity result was coded 1 (active, toxic), or -1 (inactive, non-toxic).

2.3 Chemical descriptors computed by DRAGON

For each toxicity data set introduced in 2.1 and 2.2, a large set of theoretical molecular descriptors were computed by DRAGON software DRAGON (2006). 2,489 chemical descriptors and 521 chemical descriptors were available for compounds in the ToxRefDB data and in the Rat LD_{50} data, respectively, after removing descriptors which showed almost no variation in the data set (hereafter "invariant"). The selected chemical descriptors belong to one of the following ten descriptor categories: 2D-autocorrelation (calculated from topological and atomic mass), 1D-functional group counts, 2D-eigenvalue-based indices (all 2D-descriptors based on eigenvalues), 2D-molecular properties (measures of certain physical properties), 2D-atom-centered fragments, 2D-topological descriptors (a number of topological patterns), 2D-connectivity indices (number of indices), 0D-constitutional descriptors (number of atoms), 2D-walk and path counts, and 2D-fingerprints. These categories are different logical blocks of molecular descriptors computed by DRAGON. For each data set, the categories of the chemical descriptors and the number of chemical descriptors belonging to each category after removing invariant descriptors are given in Table 2. Since different feature categories are associated with different theoretical molecular structure, it is reasonable to build base classifiers using the various feature categories and to combine base classifiers using a decision committee method. For the Rat LD_{50} data set, chemical descriptors belonging to 2D-fingerprints were not provided, so nine categories were used to build a classification model.

3 Methods and experimental setup

3.1 Background methods

3.1.1 General setting for the decision committee method

Suppose we have training data consisting of n pairs $\{(y_i, x_i)\}_{i=1}^n$, where $y_i \in \{-1, 1\}$ is a binary outcome for class level, and $x_i \in \mathbb{R}^p$ is a p-dimensional feature vector. Following the presentation in Kuncheva et al. (2001), we define a *classifier* as a map $C : x \in \mathbb{R}^p \mapsto \{-1, 1\}$. Let $\mu(C(\mathbf{x}))$ denote the output, either class label of C or continuous decision value of C that will be introduced in Section 3.1.3 below. During the construction phase, multiple base classifiers $\mathbf{C}(x) = \{C_1(x), \dots, C_L(x)\}$ are trained, and a collection of first-level outputs $\mu(\mathbf{C}(x)) = \{\mu(C_1(x)), \dots, \mu(C_L(x))\}$ are obtained. Then the final class label $\hat{C}$ can be obtained by aggregating the base classifiers through the aggregation rule $\mathcal{F}(\mathbf{C}(x))$ defined in the next section.

3.1.2 Aggregation rules: Voting and stacking

We first introduce two voting schemes that were used in this study. Suppose we have outputs $\{\mu(C_1(x)), \dots, \mu(C_L(x))\}$, where $\mu(C_l(x))$ denotes the first-level output obtained from a first-

Table 1: All endpoints for the chemical toxicity and the total number of available chemical compounds for each endpoint are given for both chemical toxicity data sets. The numbers in parentheses denote the percentage of active compounds for each endpoint.

Data	Endpoints	Toxicity category	Test species	Total number of compounds (% of active compounds)
ToxRefDB	CHR: Mouse: Liver Hypertrophy (Y1)	chronic	mouse	239 (27.62 %)
	CHR: Mouse: Liver Proliferative Lesions (Y2)	chronic	mouse	239 (38.91 %)
	CHR: Mouse: Liver Tumors (Y3)	chronic	mouse	239 (30.13 %)
	CHR: Mouse: Tumorigen (Y4)	chronic	mouse	239 (38.49 %)
	CHR: Rat: Liver Hypertrophy (Y5)	chronic	rat	247 (26.32 %)
	CHR: Rat: Liver Proliferative Lesions (Y6)	chronic	rat	247 (26.32 %)
	CHR: Rat: Tumorigen (Y7)	chronic	rat	247 (39.27 %)
	DEV: Rabbit: General Fetal Weight Reduction (Y8)	developmental	rabbit	237 (20.68 %)
	DEV: Rabbit: Pregnancy Related Embryo Fetal Loss (Y9)	developmental	rabbit	237 (29.54 %)
	DEV: Rabbit: Pregnancy Related Materl Preg Loss (Y10)	developmental	rabbit	237 (45.99 %)
	DEV: Rabbit Skeletal Axial (Y11)	developmental	rabbit	237 (23.21 %)
	DEV: Rat: General Fetal Weight Reduction (Y12)	developmental	rat	249 (34.94 %)
	DEV: Rat: Pregnancy Related Embryo Fetal Loss (Y13)	developmental	rat	249 (22.09 %)
	DEV: Rat: Pregnancy Related Materl Preg Loss (Y14)	developmental	rat	249 (19.68 %)
	DEV: Rat: Skeletal Axial (Y15)	developmental	rat	249 (44.58 %)
	MGR:Rat: Kidney (Y16)	reproductive	rat	244 (30.33 %)
	MGR: Rat: Liver (Y17)	reproductive	rat	244 (42.62 %)
	MGR: Rat: Viability PND4 (Y18)	reproductive	rat	244 (27.87 %)
Rat LD_{50}	Acute toxicity activity			3,404 (45.92 %)

Table 2: Ten categories of chemical descriptors. The number of descriptors belonging to each category was obtained after removing invariant descriptors. These molecular descriptors were derived using DRAGON software.

Category of variables	Number of variables	
	ToxRefDB	Rat LD_{50}
0D-constitutional descriptors	40	37
2D-topological descriptors	98	75
2D-walk and path counts	47	10
2D-connectivity indices	33	15
2D-autocorrelations	96	61
2D-eigenvalue-based indices	235	60
1D-functional group counts	64	131
2D-atom-centered fragments	81	104
2D-molecular properties	28	27
2D-fingerprints	382	-
Total	1,104	520

level classifier C_l, $l = 1, \dots, L$. The simplest aggregation rule is to take the average of the outputs:

$$\mathcal{F}_1(\mathbf{C}(x)) \quad = \quad \sum_{l=1}^{L} \mu(C_l(x))/L.$$

A second aggregation rule is

$$\mathcal{F}_2(\mathbf{C}(x)) \quad = \quad \beta_t^T R(\mathbf{C}), \text{ where } \beta_t = (R_t(\mathbf{C})^T R_t(\mathbf{C}))^{-1} R_t(\mathbf{C})^T y_t.$$

Here, $R(\mathbf{C}) = (1, \mu(C_1(x)), \dots, \mu(C_L(x)))$ and $R_t(\mathbf{C}) = (1, \mu(C_{1,t}(x_t)), \dots, \mu(C_{1,t}(x_t)))$ denote a collection $\{\mu(C_l(x))\}_{l=1}^{L}$ for the test set and the training set, respectively. x_ts and y_ts are the covariates and known class labels for the training set. Then, the final decision rule is

$$\hat{C} = \begin{cases} -1, & \text{if } \mathcal{F}(\mathbf{C}(x)) < c^*, \\ +1, & \text{if } \mathcal{F}(\mathbf{C}(x)) \geq c^*, \end{cases}$$

where $\mathcal{F}(\mathbf{C}(x))$ can be either $\mathcal{F}_1(\mathbf{C}(x))$ or $\mathcal{F}_2(\mathbf{C}(x))$ and c^* is a pre-determined threshold value.

Second, we employed a special type of stacked generalization which is slightly different from the procedure proposed by Wolpert (1992). Instead of using cross-validation, we split the data into a training set (X_t, Y_t), validation set (X_v, Y_v), and testing set (X_s, Y_s). Let $\tilde{C}_l^1, l = 1 \dots L$ denote level-1 classifiers, where L is the number of base classifiers, and $\tilde{C}^2$ denote a level-2 classifier. In the stage-1, training set (X_t, Y_t) is used as an input to train for the classification task, and L learning rules are obtained. Next, we apply each learning rule to the validation set and obtain L sets of level-1 outputs $\{\mu(\tilde{C}_l^1(X_v))\}_{l=1}^L$. Now $\{\mu(\tilde{C}_l^1(X_v)), Y_v\}_{l=1}^L$ are used as level-2 inputs for the stage-2 learning algorithm to learn stage-2 aggregation rule $\tilde{C}^2 : \{\mu(\tilde{C}_2^1(x))\}_{l=1}^L \in \{-1,1\}^L \mapsto \{-1,1\}$. For example, the SVM algorithm could be used as a level-1 classification inducer, and the logistic regression model could be a stage-2 learning algorithm.

3.1.3 Classification inducer: C-BSVM, Random forests, AdaBoost

In this study, we employed three different learning algorithms as classification inducers: the Support Vector Machine (SVM), AdaBoost (AdaBoost.M1, tree), and Random forests. In this section, we provide a brief review of these three learning algorithms.

Support Vector Machines (Vapnik, 1998) are among the most popular machine learning algorithms based on the kernel method. In the classification problem, SVMs find a decision function f for a given set of attributes x, and predict the class label b of target y according to the sign of $f(x)$ as follows:

$$b(x) = sign(f(x)) = \begin{cases} +1, & \text{if } f(x) \geq 0, \\ -1, & \text{if } f(x) < 0. \end{cases}$$

SVMs provide multiple types of outputs, including a decision value $f(x) \in \mathbb{R}^1$ and a class label $b(x) \in \{-1,1\}$. Many different types of SVMs have been developed, and we utilize a bound constraint version of the C classification (C-BSVM) algorithm as a base classifier. To implement the C-BSVM algorithm, the **ksvm** function in the **libsvm** library (Chang and Lin, 2001) in the **R** package R Development Core Team (2010) is utilized. In C-BSVM, the successive overrelaxation (SOR) algorithm for quadratic programs is used to train SVMs by the modified **TRON** QP solver (Lin et al., 1999; Karatzoglou et al., 2006). For more details concerning the C-BSVM algorithm, we refer the reader to Mangasarian and Musicant Mangasarian and Musicant (1999). We use linear and radial basis kernels for all SVM models in data analysis, and the quadratic polynomial kernel was added for simulation studies:

$$\begin{aligned} \text{Linear kernel} &: \; k(x, x') := < x, x' > \\ \text{Quadratic polynomial kernel} &: \; k(x, x') = (\text{scale} \cdot < x, x' > + \text{offset})^2 \\ \text{Radial basis function kernel (RBF)} &: \; k(x, x') := \exp(-\sigma \|x - x'\|^2), \end{aligned}$$

where $< \cdot, \cdot >$ denotes the inner product of two vectors, and k is a kernel function. Most internal parameters of SVM learning are obtained by the internal 5-fold CV. For the regularization margin in the Lagrange formulation (C value), we use a default setup of 1 for a relatively simple but robust prediction function. For the two example data sets in this study, we did not observe any marked differences in prediction accuracy by using different values (1, 50, and 100) for the regularization margin.

Random forests Breiman (2001) increase diversity among base classifiers by using bootstrap samples and a random selection of features. A large number of trees are then combined by majority voting without pruning for the individual trees. Breiman (2001) proved that random forests can improve classification performance by reducing the correlation between individual trees and improving each individual tree's performance. In this study, the prediction of class label $h_f(x) \in \{-1, 1\}$ is used. We implemented a random forests algorithm by using the **R** package **randomForests** (Liaw and Wiener, 2002). In this study, the number of variables randomly sampled at each split was $\sqrt{p}$, where $X \in \mathbb{R}^p$ (default set-up) and the number of minimum observations for the node was 1. For each forest, 500 individual tree were grown.

AdaBoost (*Adaptive Boosting*; Freund and Schapire (1997)) generates a set of classifiers sequentially and then aggregates them by a weighted majority voting method. In the first step of AdaBoost, n observations in the training set have a weight equal to $1/n$, and at each step, the procedure updates the weight according to the classification performance in the previous step. AdaBoost aggregates outcomes from the base classifiers by summing their probabilistic predictions, and then selecting the best prediction performance (weighted majority voting). In this study, the prediction of class label $h_f(x) \in \{-1, 1\}$ is used. We implement AdaBoost by using the **R** package **adabag**. In this study, the minimum number of observations that must exist in a node in order for a split to be attempted was 5 and the maximum depth of any node of the final tree was 5 (the root note counted as depth 0). The complexity parameter was C_P, and the weight updating coefficient was calculated by $\frac{1}{2}\log\left((1 - \text{error})/\text{error}\right)$. For each forest, 100 individual trees were grown with computational cost included as a consideration.

3.2 Proposed method

3.2.1 Two-stage cross-validation

As discussed in Hansen and Salamon (1990) and Opitz and Maclin (1999), the decision committee method can reduce test-set error sufficiently by aggregating a few base classifiers, instead of combining all base classifiers. Higher prediction accuracy can be achieved by eliminating some irrelevant or noisy base classifiers. During this selection phase, the forward selection approach is adopted to find optimal combinations of base classifiers similar to Breiman (1996). In the first step, the best base classifier based on the given prediction accuracy is selected, and denoted by $C^1(x)$. In the second step, each of the remaining base classifiers $\{C_l^{(2)}(x)\}_{l=1}^{L-1}$ is integrated with $C^1(x)$ by a given aggregation rule $\mathcal{F}(\mathbf{C}(x))$. The best pair of base classifiers is picked up, and denoted by $C^2(x)$. For each step of the forward selection approach, the prediction accuracy is assessed, and only one best base classifier is added. The forward selection procedure will ensure that the matrix of first-level outcomes from selected base classifiers has full rank by removing redundant base classifiers.

In this study, we propose a 5-fold cross-validation (CV) method to decide the total number of base classifiers K to be incorporated in the final classifier. The training set is randomly split into five subsets, and four out of five subsets are used to train base classifiers. For given L subsets of feature variables $\{Z_1, \ldots, Z_L\}$, let $\mathbf{C}_{cv,i} = \{C_{cv,i}(Z_1), \ldots, C_{cv,i}(Z_L)\}$, $i = 1, \ldots, 5$ denote the set of base classifiers for the remaining set which is not used for the ith training. In this phase, we continue the forward selection procedure until all base classifiers are combined. At each step, prediction accuracy is assessed for each of the five sets $\{\mathbf{C}_{cv,i}\}_{i=1}^{5}$. Then, we take the average of the prediction accuracies over five sets, and K is decided by the number of combined base classifiers in which the highest average prediction accuracy is achieved. Since another internal 5-fold CV is conducted to determine internal parameters of the SVM learning, a two-stage 5-fold CV is used for SVM in this phase. To the best of our knowledge, the proposed two-stage CV is novel. Note that a single-stage CV is utilized to decide the number of base classifiers using AdaBoost and random forests as classification inducers.

3.2.2 Interactive feature space

Suppose that we have the same training data $\{(y_i, x_i)\}_{i=1}^{n}$ as described in Section 3.1.1, and testing data $\{x_i\}_{i=n+1}^{I}$. Suppose each feature variable belongs to at least one feature category m, and $X_m = \{x_{i,j}\}_{i=1,\ j=1}^{n\quad p_m} \in \mathbb{R}^{n\times p_m}$ denotes a feature matrix for the category m, where p_m is the number of feature variables belonging to the category m, $m = 1, \ldots, M$ and $\sum_{m=1}^{M} p_m = p$. Good examples of these categories would be the blocks of chemical descriptors in chemical toxicity data presented in this study, or gene ontology terms in gene expression profiles (Ashburner et al., 2000).

First, we generate a univariate feature space consisting of M feature categories of $\mathbf{X} = \{X_m\}_{m=1}^{M}$ and a bivariate feature space $\mathbf{X}^* = \{X_q^* = (X_m, X_{m'}),\ m, m' = 1, \ldots, M,\ m \neq m',\ q = 1, \ldots, Q = \binom{M}{2}\}$. We then construct the interactive feature space $\mathbf{Z} = \{\mathbf{X} \cup \mathbf{X}^*\} = \{Z_l\}_{l=1}^{L=M+Q}$, where Z_l would be either X_m or $X_m \cup X_{m'}$. By doing this, the interactive feature space allows us to use the information of the feature categories both marginally and interactively. To the best of our knowledge, this study is the first to use the interactive relationship between feature categories to construct base classifiers for decision committees.

3.2.3 IDC with different aggregation rules

Our proposed method can be summarized in two steps: first, during the construction phase, each base classifier is trained using Z_l from the interactive feature space. $C(Z_l)$ and $\mu(C(Z_l))$ denote the base classifiers and the first-level outputs by using the interactive feature space, respectively. Next, by using 5-fold CV as described in Section 3.2.1, the number of base classifiers K to be aggregated is determined. Once K is decided from the training set, the same forward selection procedures are repeated until we find K base classifiers with the best performance. We call the above decision committee system the Interactive Decision Committee (IDC). The proposed IDC method is new.

Figure 1 illustrates the basic framework for the IDC method. In this flowchart, X_ms denote the $n \times p_m$ matrix for feature categories $m, m = 1, \ldots, M$, and p_m is the number of variables belonging to feature category m. Z_ls are elements of the interactive feature space, so Z_l is either feature category X_m or a pair of two feature categories $X_m \cup X_{m'}$, $m \neq m'$ as explained in 3.2.2. Each base classifier $T(Z_l)$ is trained using feature category Z_l in the training set. The total number of base classifiers K to be combined for the final classifier is determined in this phase by the use of 5-fold CV. Then, first-level predicted outputs can be obtained from the base classifier $C(Z_l)$s for the test individuals. Finally, the final decision $\hat{C}$ is made by aggregating K base classifiers through using the aggregation rule. In practice, however, we do not have outputs for the new examples. Therefore, we use first-level class predictions from the validation set to find the best K base classifiers to be combined. Then, the selected K base classifiers are combined to predict class labels for new examples.

We utilized voting and stacking to combine base classifiers. For the voting method (denoted by IDC), the two aggregation rules $\mathcal{F}_1(\mathbf{C}(x))$ and $\mathcal{F}_2(\mathbf{C}(x))$ described in Section 3.1.2 are utilized to combine base classifiers after the system size K is determined by 5-fold CV. First, we use the aggregation rule of $\mathcal{F}_1(\mathbf{C}(x))$ having $\mu(C_l(Z)) = \mu(C(Z_l)) = b(Z_l) = sign(f(Z_l)) \in \{-1, 1\}$ in SVM, and $\mu(C_l(Z)) = \mu(C(Z_l)) = h_f(Z_l) \in \{-1, 1\}$ in AdaBoost and random forests as the first-level output. The IDC method with this aggregation rule is denoted by $IDC_{\mathcal{F}_1}$. Second, we use the aggregation rule of $\mathcal{F}_2(\mathbf{C}(x))$ having $\mu(C_l(Z)) = \mu(C(Z_l)) = f(Z_l) \in \mathbb{R}^1$ as the first-level output, and the IDC method with this rule is denoted by $IDC_{\mathcal{F}_2}$. We apply $\mathcal{F}_2(\mathbf{C}(x))$ for the base classifiers obtained by SVM only. In voting, we set the threshold value $c^* = 0$ for the final decision rule. Therefore, $\mathcal{F}_1(\mathbf{C}(x))$ is equivalent to the majority voting method, and $\mathcal{F}_2(\mathbf{C}(x))$ yields the same result as linear discriminant analysis (LDA) using R as a new feature variable set.

For the IDC method with a stacked generalization (denoted by IDC stacking), two different learning algorithms at stage-2 were adopted separately in order to learn a combining method: L_2 penalized logistic regression (Park and Hastie, 2008) and a single-hidden layer neural network (Ripley, 2008). L_2 penalized logistic regression (L_2-logit) was implemented through the **R** package **stepPlr** using BIC (Bayesian Information Criterion) as complexity parameters to compute the score and selecting base classifiers through the forward stepwise (forward select first, then backward deletion follows) selection. A single-hidden layer neural network (NN) was implemented through the **R** package **nnet** with one unit in the hidden layer (single layer), initial random weights on [-0.1, 0.1], a parameter of 0.0005 for weight decay, and a maximum iteration of 300. IDC methods stacking with L_2 penalized logistic regression and NN are denoted by IDC_{LR} and IDC_{NN}, respectively. Note that we do not have to decide on the system size of K for the IDC stacking method although base classifiers are trained by the same IDC method. Therefore, four different types of aggregation methods are applied to combine base classifiers that are trained by the proposed IDC method.

4 Evaluation measure and other methods

4.1 Prediction accuracy measurement

For the ToxRefDB data, we have fewer active compounds compared to inactive compounds, which is imbalanced for all binary endpoints. Thus we chose to use both sensitivity and specificity to reflect performance on the classification task following Assareh, Assareh et al. (2008). Regarding an active (+1) as positive while an inactive (-1) as negative, sensitivity and specificity are calculated as follows:

$$\begin{aligned}\text{Sensitivity} &= \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false negatives}},\\ \text{Specificity} &= \frac{\text{number of true negatives}}{\text{number of true negatives} + \text{number of false positives}}.\end{aligned}$$

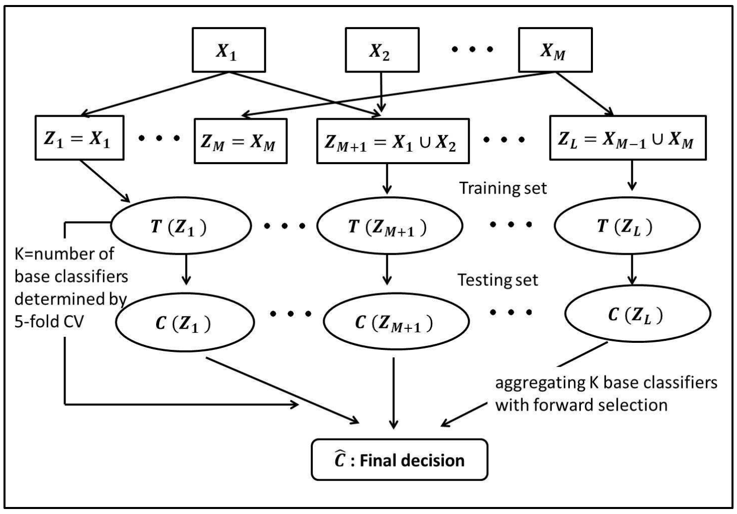


Figure 1: Flowchart of the IDC method with M feature categories. $T(Z_l)$ and $C(Z_l)$ denote the base classifiers which are trained using feature set Z_l for the training set and the testing set, respectively.

Therefore, sensitivity is the proportion of actual active compounds that are correctly classified as active compounds. Similarly, specificity is the proportion of the true inactive compounds which are correctly classified as inactive compounds. The average of sensitivity and specificity was used as a prediction accuracy measure to select base classifiers in forward selection, to decide the number of base classifiers, and to compare the performances on the classification task among different methods:

$$\text{Accuracy} \quad = \quad \frac{\text{sensitivity} + \text{specificity}}{2}.$$

To compare improvement in prediction accuracy relative to a single large, unaggregated classifier, relative percent improvement (RI) of the classification model M was calculated as follows:

$$\text{RI(M)} \quad = \quad \frac{\text{accuracy of model M} - \text{accuracy of a single large classifier}}{\text{accuracy of a single large classifier}} \times 100.$$

4.2 Classification methods

First, a single large, unaggregated classifier was used as a reference model (Single). Although random forests and tree-based AdaBoost are already decision committee methods rather than single classifiers, what we really mean by denoting a “single random forest” or a “single AdaBoost” are the usual random forests or AdaBoost without using the IDC method. For a single classifier, we combine the training set and validation set for training. By doing this, we have a larger training set for single classifiers compared to the IDC method and the IDC stacking method.

Second, we apply the proposed IDC method (i.e., determining K and selecting the best K base classifiers by 5-fold CV and forward selection) to each classifier inducer (IDC). We find a training rule and the number of base classifiers to be combined using the training set and decide on the best K base classifiers based on the validation set. Instead of using the training and validation sets separately, one can do another cross-validation by combining the training and validation sets. In our study, using the validation set (a data set that does not contribute to training at all) seems to be slightly better with respect to the performance on the test set than using another cross-validation on the combined training and validation sets (for both training and finding the best K classifiers).

Last, we applied stacked generalization with L_2 penalized logistic regression and NN (IDC stacking). IDC and IDC stacking are the same in the first stage, but they combine base classifiers differently.

The primary goal of this study was to compare the interactive decision committee method to single, unaggregated classification methods rather than to find optimal subsets of features or the optimal classifier inducer. Therefore, comparison between different classification inducers was not done in this study.

5 Simulation study

We have empirically evaluated the proposed IDC and IDC stacking methods compared to a single classifier in three classification inducers: SVM, random forests, and tree-based AdaBoost algorithms. Ten random sets of data were generated for the binary classification task, and we used 60% of the data for training, 20% for validation and the remaining set for testing. Again, combined training and validation sets (80%) were used for training in a single classifier.

5.1 Simulation set-up

Four feature categories $X = \{X_i\}_{i=1}^4$ were randomly generated, and each category X_i consists of three feature variables $\{x_{ij}\}_{j=1}^3$ from the standard multivariate normal distribution $\mathbf{N}_3(0, I)$, where I denotes a 3×3 identity matrix. New variables Z were generated by combining variables in four feature categories differently so that the effect by univariate or bivariate feature categories could be added to the individual feature variables. Then, we simulated logistic regression models under six different scenarios. A binary outcome was obtained by $Y = \mathbf{1}\{U < p_0\}$, where U is uniformly distributed in $(0, 1)$, and $p_0 = \exp(\eta)/(1 + \exp(\eta))$, where η is computed by six different scenarios. The sample size for each run was 300.

Simulation 1: Two new variables $Z = (z_1, z_2)$ were generated, where $z_1 = (x_{11} + x_{12}) + (x_{21} + x_{22} + x_{23})$, $z_2 = (x_{31} + x_{32} + x_{33}) + (x_{41} + x_{42} + x_{43})$. For logistic regression, $\eta = X\beta + Z\gamma$, where $\beta = (\beta_1, \beta_2, \beta_3, \beta_4)^T$, $\beta_1 = (0.12, 0.5, 0.5)^T$, $\beta_2 = (0.15, 0.18, 0.7)^T$, $\beta_3 = (0.11, 0.8, 0.8)^T$, $\beta_4 = (0.5, 0.15, 0.15)^T$, and $\gamma = (1.5, 1.8)^T$.

Simulation 2: Three new variables $Z = (z_1, z_2, z_3)$ were generated, where $z_1 = (x_{11} + x_{12}) \times (x_{21} + x_{22} + x_{23})$, $z_2 = (x_{11} + x_{12} + x_{13}) \times (x_{31} + x_{32} + x_{33})$, and $z_3 = (x_{31} + x_{32} + x_{33}) \times (x_{41} + x_{42} + x_{43})$. For logistic regression, $\eta = x_{12}\beta_{12} + Z\gamma$, where $\beta_{12} = 0.5$ and $\gamma = (1.5, -1.5, 1.8)^T$.

Simulation 3: No new variable for feature categories was derived. $\eta = x_{11}\beta_{11} + x_{23}\beta_{23} + x_{31}\beta_{31} + 1.2x_{11} \times x_{31}$, where $\beta_{11} = 1.2$, $\beta_{23} = -1.8$, $\beta_{31} = 1.2$.

Simulation 4: No new variable for feature categories was derived. $\eta = X\beta$, where $\beta_1 = (0.12, 0.1, 0.1)^T$, $\beta_2 = (-0.12, -0.1, -0.1)^T$, $\beta_3 = (0.11, 0.1, 0.1)^T$, and $\beta_4 = (-0.11, -0.1, -0.1)^T$.

Simulation 5: Two new variables $Z = (z_1, z_2)$ were generated, where $z_1 = (x_{11} + x_{12}) + (x_{12} + x_{22} + x_{23})$, $z_2 = (x_{31} + x_{32} + x_{33}) + (x_{41} + x_{42} + x_{43})$. For logistic regression, $\eta = x_{12}\beta_{12} + Z\gamma + 5.0\epsilon$, where $\beta_{12} = 0.5$, $\gamma = (0.1, 0.1)^T$, and random noise $\epsilon \sim \mathbf{N}(0, 1)$.

Simulation 6: Two new variables $Z = (z_1, z_2)$ were generated, where $z_1 = (x_{11} + x_{12}) + (x_{12} + x_{22} + x_{23})$, $z_2 = (x_{31} + x_{32} + x_{33}) + (x_{41} + x_{42} + x_{43})$. For logistic regression, $\eta = x_{12}\beta_{12} + Z\gamma$, where $\beta_{12} = 0.5$, $\gamma = (0.1, 0.1)^T$.

In Simulation 1, two new variables generated by bivariate linear combination have larger effects compared to individual variables, so Simulation 1 would be favorable to the IDC methods. In Simulation 2, three new variables were derived by combining feature variables belonging to different feature categories non-linearly, and they have a greater effect than the individual variables. In Simulation 3, three individual variables and one second-order interaction effect by individual variables exist while no intended effect by feature categories exists. In Simulation 4, variables in feature category 1 and variables in feature category 2 have equal effects but with opposite signs. The same is true for the variables in feature category 3 and feature category 4. All variables have small positive effects. In Simulation 5, one weak individual effect and two weak categorical effects exist while a large random error effect exists. Simulation 6 is similar to Simulation 1 except that there is no intended large noise effect.

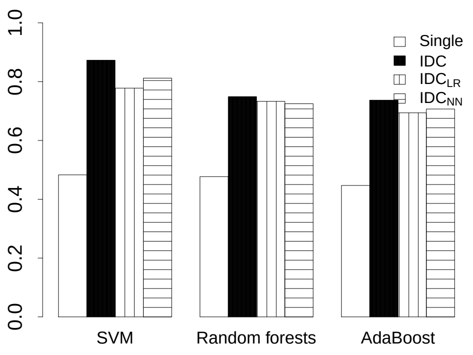
5.2 Main results

5.2.1 Prediction accuracy

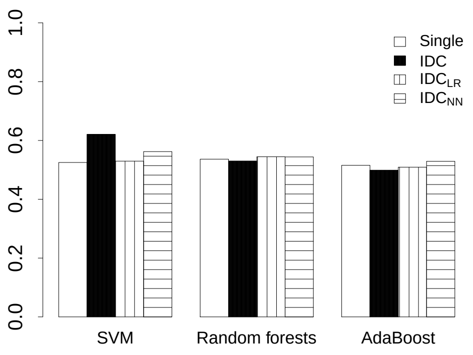
Figure 2 presents experimental results under six scenarios. We first focus on the prediction accuracies. In Simulation 1, both the IDC and the IDC stacking methods outperformed single classifiers, especially for the IDC method regardless of classifier inducer (RI: 80.75% for SVM; 57.02% for random forests; 64.88% for AdaBoost). This is not a surprising result since there are greater effects by feature categories.

In Simulation 2, both the IDC and the IDC stacking methods performed similarly to the single classifier except for the IDC with SVM (18.29%). This indicates that the IDC method might be able to catch a non-linear bivariate structure among feature categories better than a single classifier, but the performance can depend on the classification inducer.

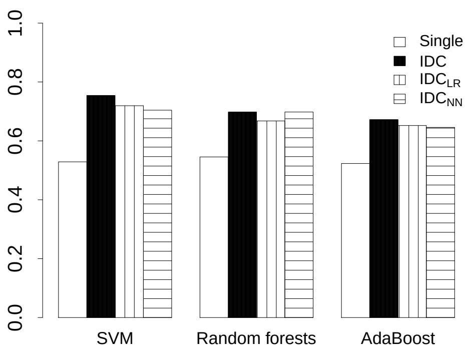
In Simulation 3, both IDC and IDC staking outperformed single classifiers regardless of classification inducers. IDC (42.53% for SVM; 28.49% for AdaBoost) performed slightly better than IDC stacking (35.92% for SVM; 24.67% for AdaBoost) in SVM and AdaBoost, but the results are comparable to random forests (28.06% for IDC; 28.07% for IDC stacking).



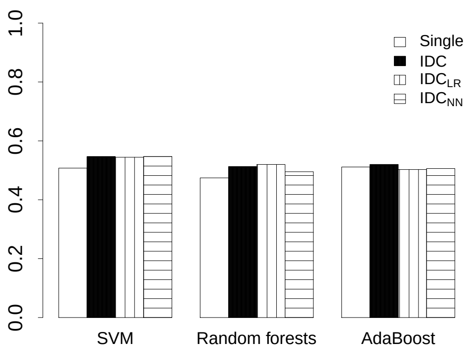
(a) Simulation 1



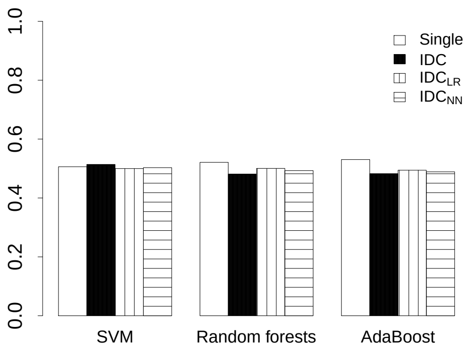
(b) Simulation 2



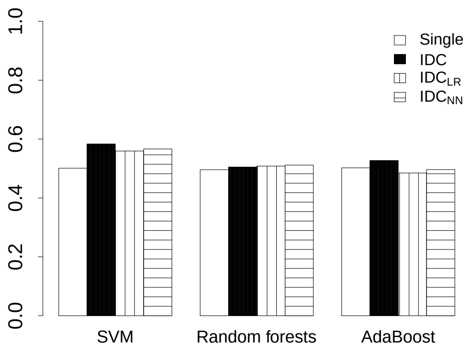
(c) Simulation 3



(d) Simulation 4



(e) Simulation 5



(f) Simulation 6

Figure 2: Average prediction accuracies over ten replications of a single classifier, IDC, and IDC with stacking applied using three classifiers, SVM, Random forests, and AdaBoost (tree), are compared. For SVM, the best result among three kernel functions and two aggregation rules are presented. For each classifier inducer, the first bar denotes the prediction accuracy of a single classifier (Single), the second bar is for the IDC method (IDC), the third bar is for the IDC stacking with L_2 logistic regression (IDC$_{LR}$), and the last bar is for the IDC stacking with NN (IDC$_{NN}$).

In Simulation 4, both IDC (7.89% for SVM; 8.23% for random forests) and IDC stacking (7.89% for SVM; 9.7% for random forests) performed slightly better than single classifiers in SVM and random forests but similar to or slightly worse than single classifiers in AdaBoost (1.76% for IDC; -0.98% for IDC stacking).

In Simulation 5, single classifiers performed slightly better than IDC (-7.68% for random forests; -9.06% for AdaBoost) and IDC stacking (-4.03% for random forests; -6.79% for AdaBoost) in random forests and AdaBoost, and slightly worse than or similar to IDC (1.58%) and IDC stacking (-0.59%) in SVM.

In Simulation 6, both IDC (16.57% for SVM; 1.81% for random forests) and IDC stacking (12.97% for SVM; 3.23% for random forests) performed slightly better than single classifiers in SVM and random forests. In AdaBoost, IDC (4.98%) performed slightly better than a single AdaBoost, but slightly worse than IDC stacking (-1.20%).

Based on the empirical results from Simulation 5 and Simulation 6, there appears to be more degradation of performance for the IDC and IDC stacking methods compared to a single classifier, as effects by random noise increases. Simulation 3 shows that the IDC method can improve prediction accuracy compared to a single classifier when no intended categorical information exists, but a large interaction effect between feature variables belonging to different feature categories exists. Simulation 2 and Simulation 4 show the possibility that IDC may not be able to capture non-linearly associated feature category information or opposite effects between categories well, but it still performs well compared to single classifiers. Also, IDC and IDC stacking can show different behavior depending on the classification inducers and data characteristics.

5.2.2 Standard error estimates in prediction accuracy

In Simulation 1 and Simulation 3, the standard error estimates of the IDC method were smaller than those from single classifiers (overall less than half of the estimates from single classifiers) except for random forests in Simulation 1 (0.025 for single random forests vs. 0.026 for IDC random forests). The standard error estimates of the IDC stacking method were smaller than those from single classifiers, but larger than or similar to those from IDC overall. In Simulation 2, the standard error estimates of the IDC method were slightly smaller than those from single classifiers in SVM and AdaBoost, but larger than in random forests (0.017 for single vs. 0.023 for IDC). In Simulations 4, 5, and 6, the standard error estimates of the IDC method were greater than those of single classifiers in SVM, but smaller than or similar to the other two classifier inducers.

Overall, the standard error estimates of the IDC methods were smaller than or similar to those of the IDC stacking methods as well as those of single classifiers. The standard error estimates of the IDC stacking with L_2 penalized logistic regression were smaller than or similar to those of the IDC stacking method with NN except for SVM (0.038 for IDC_{LR}; 0.02 for IDC_{NN}) and AdaBoost (0.021 for IDC_{LR}; 0.011 for IDC_{NN}) in Simulation 6. The empirical results show that the IDC method can perform better than single classifiers with lower variation when large but relatively simple bivariate feature categorical effects exist or a large interaction effect of individual variables belonging to two feature categories exists. The IDC methods (with voting) performs better than or compares favorably with the IDC stacking method in the current setting.

5.2.3 System size: The number of base classifiers to be combined

For the IDC method, a small investigation was carried out to determine whether the system size K differed by aggregation method or by classification inducer. LDA-type aggregation tends to select a smaller number of base classifiers to be combined compared to the unweighted average (majority voting) in SVM. Overall, three classification inducers have a similar system size K, but AdaBoost tends to have smaller number of base classifiers to be combined compared to SVM and random forests except for Simulation 6.

6 Example: toxicity data analysis

In this section, we describe the empirical results of applying the IDC and the IDC stacking methods as well as a single classifier to two toxicity data sets.

6.1 ToxRefDB data

Three classifier inducers were explored: SVM, random forests, and tree-based AdaBoost. For stacked generalization, L_2 penalized logistic regression with stepwise selection and a single layer NN were adopted. Ten replications were obtained randomly. In each run, the data set was randomly split into three sets, and we used 60% of the data for training, 20% for validation, and the remaining set for testing. The average prediction accuracies of the ten replications were compared. In SVM, linear and radial basis functions were utilized without optimizing any other parameters, considering computational cost. The data analysis was conducted under the same set-up in the simulation studies except for the polynomial kernel in SVM.

6.1.1 Prediction accuracy

Table 3 displays the average of the prediction accuracies computed by using the test set in ToxRefDB. In SVM, the highest accuracies between linear and radial basis kernel are presented. For the IDC method, the best prediction accuracies between $IDC_{\mathcal{F}_1}$ and $IDC_{\mathcal{F}_2}$ are reported. In SVM, the prediction accuracy of the IDC method achieved the highest prediction accuracies for 14 endpoints (Y1, Y2, Y3, Y5, Y6, Y8, Y9, Y10, Y11, Y12, Y13, Y14, Y16, Y17), especially for Y10 (RI: 7.24%), Y13 (8.2%), and Y17 (7.26%). IDC stacking performed best for three endpoints (Y4 and Y18 for IDC stacking with L_2-logit and Y7 for IDC stacking with NN). In Y15, a single SVM achieved the highest prediction accuracy.

With random forests, the IDC method achieved the highest prediction accuracies for 7 endpoints (Y1, Y8, Y9, Y10, Y13, Y15, and Y17), especially for Y8 (RI: 9.96%), Y10 (16.36%) and Y15 (7.62%). IDC stacking performed better than the IDC methods as well as single classifiers for 7 endpoints (Y7, Y11, and Y14 for IDC stacking with L_2-logit and Y3, Y4, Y5, and Y12 for IDC stacking with NN).

Using AdaBoost, the IDC method performed best for 8 endpoints (Y1, Y5, Y8, Y9, Y10, Y12, Y15, and Y17), especially for Y10 (RI: 9.88%). IDC stacking achieved the highest prediction

Table 3: Averages (ACC) and standard error estimates (SEE) of prediction accuracies over ten replications for ToxRefDB data. For SVM, the best results among three kernel functions are presented indicating which kernel function is the best [l=linear and r=radial basis functions]. The best method for each classification inducer is marked in **bold**.

Endpoints		SVM					Random forests				AdaBoost (tree)			
		Single	$IDC_{\mathcal{F}_1}$	$IDC_{\mathcal{F}_2}$	IDC_{LR}	IDC_{NN}	Single	IDC	IDC_{LR}	IDC_{NN}	Single	IDC	IDC_{LR}	IDC_{NN}
Y1	ACC	0.509^r	0.518^l	**0.530^l**	0.500^r	0.528^l	0.489	0.480	0.498	**0.515**	0.502	**0.524**	0.499	0.511
	SEE	0.009	0.022	0.026	0.000	0.020	0.011	0.019	0.008	0.022	0.015	0.014	0.015	0.019
Y2	ACC	0.505^r	**0.535^l**	0.508^r	0.499^r	0.517^l	**0.498**	0.482	0.487	0.488	0.515	0.492	0.544	**0.561**
	SEE	0.020	0.029	0.021	0.016	0.021	0.023	0.019	0.020	0.023	0.022	0.023	0.017	0.028
Y3	ACC	0.494^r	**0.515^r**	0.491^l	0.507^r	0.497^r	0.471	0.504	0.515	**0.520**	0.474	0.485	**0.507**	0.488
	SEE	0.003	0.029	0.031	0.007	0.015	0.011	0.019	0.020	0.018	0.018	0.017	0.028	0.023
Y4	ACC	0.509^r	0.488^l	0.524^r	**0.528^l**	0.502^l	0.510	0.515	0.515	**0.530**	0.500	0.486	0.511	**0.526**
	SEE	0.016	0.013	0.022	0.026	0.029	0.017	0.014	0.020	0.026	0.013	0.028	0.017	0.026
Y5	ACC	0.496^r	**0.522^l**	0.520^l	0.502^r	0.502^l	0.491	0.498	0.499	**0.520**	0.481	**0.509**	0.487	0.490
	SEE	0.002	0.026	0.026	0.003	0.010	0.010	0.016	0.012	0.020	0.011	0.019	0.012	0.022
Y6	ACC	0.501^r	**0.525^r**	0.500^r	0.500^r	0.489^r	**0.515**	0.507	0.499	0.469	**0.515**	0.505	0.507	0.509
	SEE	0.003	0.026	0.027	0.000	0.006	0.010	0.024	0.005	0.016	0.010	0.021	0.015	0.021
Y7	ACC	0.487^r	0.518^r	0.498^l	0.525^r	**0.541^l**	0.499	0.489	**0.504**	0.498	0.503	0.481	**0.509**	0.476
	SEE	0.009	0.022	0.025	0.012	0.018	0.018	0.019	0.023	0.019	0.015	0.021	0.020	0.023
Y8	ACC	0.500^r	0.506^r	**0.525^l**	0.511^r	0.519^r	0.502	**0.552**	0.501	0.518	0.505	**0.528**	0.496	0.490
	SEE	0.000	0.022	0.023	0.008	0.009	0.009	0.022	0.015	0.020	0.010	0.021	0.011	0.019
Y9	ACC	0.495^r	**0.528^r**	0.510^l	0.507^l	0.518^l	0.493	**0.495**	0.492	0.484	0.492	**0.503**	0.487	0.496
	SEE	0.002	0.034	0.023	0.014	0.023	0.013	0.023	0.008	0.015	0.015	0.016	0.016	0.020
Y10	ACC	0.497^r	0.515^r	**0.533^l**	0.496^l	0.500^r	0.489	**0.569**	0.502	0.539	0.496	**0.545**	0.536	0.530
	SEE	0.018	0.011	0.026	0.027	0.036	0.022	0.016	0.019	0.022	0.022	0.014	0.021	0.020
Y11	ACC	0.505^r	**0.512^l**	0.506^l	0.498^r	0.493^r	0.491	0.494	**0.513**	0.508	0.484	0.486	**0.505**	0.490
	SEE	0.005	0.011	0.019	0.003	0.004	0.011	0.018	0.016	0.018	0.010	0.023	0.013	0.019
Y12	ACC	0.502^r	**0.512^l**	0.496^r	0.503^r	0.501^r	0.493	0.512	0.485	**0.519**	0.485	**0.513**	0.510	0.479
	SEE	0.006	0.017	0.022	0.014	0.018	0.022	0.021	0.015	0.026	0.019	0.020	0.022	0.021
Y13	ACC	0.500^r	0.504^r	**0.541^r**	0.497^r	0.503^r	0.491	**0.508**	0.504	0.506	0.491	0.507	0.517	**0.539**
	SEE	0.000	0.031	0.030	0.003	0.005	0.007	0.004	0.007	0.023	0.010	0.019	0.021	0.020
Y14	ACC	0.500^r	**0.505^l**	0.503^l	0.500^1	**0.499^2**	0.494	0.483	0.499	0.495	0.484	0.491	0.494	**0.495**
	SEE	0.005	0.008	0.022	0.016	0.001	0.011	0.017	0.014	0.011	0.006	0.016	0.010	0.015
Y15	ACC	**0.534^r**	0.524^l	0.511^r	0.517^l	0.497^r	0.512	**0.551**	0.492	0.513	0.512	**0.535**	0.526	0.528
	SEE	0.021	0.025	0.025	0.021	0.027	0.031	0.023	0.026	0.015	0.021	0.020	0.017	0.016
Y16	ACC	0.504^r	**0.513^r**	0.480^r	0.504^r	0.511^r	**0.522**	0.503	0.500	0.489	0.515	0.472	0.510	**0.518**
	SEE	0.004	0.010	0.031	0.003	0.008	0.019	0.018	0.010	0.017	0.011	0.014	0.019	0.018
Y17	ACC	0.496^r	**0.532^l**	0.514^l	0.515^r	0.517^l	0.495	**0.520**	0.510	0.513	0.501	**0.528**	0.498	0.460
	SEE	0.012	0.019	0.017	0.015	0.017	0.022	0.021	0.022	0.021	0.017	0.020	0.024	0.029
Y18	ACC	0.506^r	0.495^l	0.484^l	**0.506^l**	0.500^l	**0.509**	0.498	0.508	0.490	0.503	0.488	0.485	**0.509**
	SEE	0.006	0.013	0.021	0.024	0.026	0.014	0.024	0.015	0.018	0.013	0.027	0.015	0.022

Table 4: Averages (AVG) and standard error estimates (SEE) of the number of base classifiers to be combined which is determined by 5-fold CV over ten replications for ToxRefDB and Rat LD_{50} data. For SVM, the best results among linear and radial basis function kernels are presented.

Data	Endpoints	SVM				Random forests		AdaBoost (tree)	
		$IDC_{\mathcal{F}_1}$		$IDC_{\mathcal{F}_2}$					
		AVG	SEE	AVG	SEE	AVG	SEE	AVG	SEE
ToxRefDB	Y1	9.4	0.933	14.5	2.377	2.8	0.442	2.6	0.427
	Y2	9.7	1.075	14.4	2.212	5	0.683	4.6	0.306
	Y3	8.3	0.367	13.2	1.569	3.2	0.533	2.8	0.327
	Y4	3.2	0.800	17.7	2.556	3.4	0.521	4.8	0.533
	Y5	6.2	0.533	13.2	1.919	2	0.000	2.6	0.427
	Y6	7.7	1.096	18.7	2.246	2.4	0.267	2.4	0.267
	Y7	3	0.333	11.8	2.215	4.9	0.482	5.6	1.327
	Y8	9.3	1.033	17.6	1.500	1.9	0.100	2	0.000
	Y9	7.3	0.667	9.9	1.581	2.2	0.200	2.2	0.200
	Y10	11.1	1.169	19.2	2.764	5.2	0.533	8	0.667
	Y11	1.4	0.163	13.7	1.627	2.4	0.267	2	0.000
	Y12	9.2	0.964	14.6	1.675	3.2	0.533	4.4	0.581
	Y13	10.1	2.163	18.3	3.461	2	0.000	2	0.000
	Y14	1.1	0.100	16.6	3.557	2	0.000	2	0.000
	Y15	5.8	1.052	13.9	1.748	5.2	0.442	6.8	0.854
	Y16	1.6	0.163	17	2.186	2.2	0.200	2.4	0.267
	Y17	2.8	0.442	14.8	1.692	4	0.422	6.2	0.757
	Y18	2	0.000	15.7	1.862	2.8	0.327	2.8	0.327
Rat LD_{50}		3.72	0.549	8.56	1.036	-	-	-	-

Table 5: Averages (AVG) and standard error estimates (SEE) of the standard deviation in the total number of base classifiers within 5-fold CV in IDC applying SVM for ToxRefDB data.

Endpoints	$IDC_{\mathcal{F}_1}$				$IDC_{\mathcal{F}_2}$			
	Linear		RBF		Linear		RBF	
	AVG	SEE	AVG	SEE	AVG	SEE	AVG	SEE
Y1	3.30	0.49	1.80	1.27	10.03	1.28	12.05	1.63
Y2	3.99	0.40	1.08	0.17	9.94	1.37	9.51	1.26
Y3	3.97	0.85	0.93	0.30	6.77	0.86	11.31	1.20
Y4	5.02	0.47	1.51	0.50	11.32	1.62	11.30	1.82
Y5	4.15	0.31	0.36	0.14	9.07	1.38	11.99	1.21
Y6	3.14	0.29	0.30	0.08	8.68	1.64	11.14	1.37
Y7	4.22	0.62	1.25	0.14	7.83	1.06	10.28	1.06
Y8	3.85	0.56	0.09	0.06	10.13	1.44	9.88	2.11
Y9	3.97	0.38	0.48	0.07	9.87	1.65	9.80	1.25
Y10	4.06	0.76	2.87	0.44	10.96	1.11	9.98	1.28
Y11	5.71	0.56	0.51	0.26	9.14	1.39	11.23	1.77
Y12	5.11	0.59	0.47	0.13	10.94	1.04	10.69	1.25
Y13	5.07	1.06	0.28	0.13	10.94	1.56	10.16	1.03
Y14	3.75	0.58	0.04	0.04	10.73	1.45	13.22	1.66
Y15	3.32	0.36	3.14	0.72	7.64	1.10	11.05	1.15
Y16	2.94	0.29	0.51	0.30	10.62	1.61	9.39	1.22
Y17	2.99	0.36	1.87	0.49	9.47	1.34	10.17	1.26
Y18	3.63	0.44	0.73	0.09	8.01	0.79	12.47	1.27

accuracies for 9 endpoints (Y3, Y7, Y11 for IDC stacking with L_2-logit and Y2, Y4, Y13, Y14, Y16, and Y18 for IDC stacking with NN). A single AdaBoost tree performed best for Y6.

The empirical results show that the IDC method was the best choice for 10 endpoints: Y1 ($IDC_{\mathcal{F}_2}$ applying SVM with linear kernel), Y5 ($IDC_{\mathcal{F}_1}$ applying SVM with linear kernel), Y6 ($IDC_{\mathcal{F}_1}$ applying SVM with RBF kernel), Y8 (IDC applying random forests), Y9 ($IDC_{\mathcal{F}_1}$ applying SVM with RBF kernel), Y10 (IDC applying random forests), Y13 ($IDC_{\mathcal{F}_2}$ applying SVM with RBF kernel), Y14 ($IDC_{\mathcal{F}_1}$ applying SVM with linear kernel), Y15 (IDC applying random forests), and Y17 ($IDC_{\mathcal{F}_1}$ applying SVM with linear kernel). IDC stacking was the best method for 6 endpoints: Y2 (IDC stacking with L_2 penalized logistic applying AdaBoost tree), Y3, Y4 (IDC stacking with NN applying random forests), Y7 (IDC stacking with NN applying SVM with linear kernel), Y11 (IDC stacking with L_2 penalized logistic applying random forests), and Y12 (IDC stacking with NN applying random forests). Both IDC and IDC stacking methods failed to improve prediction accuracies compared to a single classifier for Y16 and Y18, and single random forests achieved the best performance in the current experimental setting. Overall, the classification performance was not very good, and the IDC or the IDC stacking methods are not always better than a single classifier. The experimental results, however, show that the IDC method and the IDC stacking method perform as well or better than single classifiers for the majority of endpoints in the ToxRefDB data, especially applying the SVM method which is kernel based, and base classifiers are not trained by a decision committee method.

6.1.2 System size: The number of base classifiers to be combined

Table 4 provides the mean and standard error estimates of the number of base classifiers to be aggregated by the IDC method over ten replications in ToxRefDB, which was decided by two-stage 5-fold CV. Applying SVM, $IDC_{\mathcal{F}_2}$ tends to have more base classifiers (on average, across all 18 endpoints, 15.27 base classifiers) than the unweighted average $IDC_{\mathcal{F}_1}$ (on average, 6.07) with greater variation, especially for the radial basis kernel (on average, across 18 endpoints, standard error estimates were 0.702 and 2.153 for $IDC_{\mathcal{F}_1}$ and $IDC_{\mathcal{F}_2}$, respectively). However, the number of base classifiers was still less than half of all base classifiers for all four models. The IDC method applying random forests and AdaBoost tree tend to combine a smaller number of base classifiers compared to the IDC method applying SVM (on average, across 18 endpoints, 3.16 and 3.68 base classifiers for random forests and AdaBoost tree, respectively) with less variation (on average, standard error estimates were 0.331 and 0.404 for random forests and AdaBoost tree, respectively).

Since the number of base classifiers for the final classifier was determined by 5-fold CV, we explored variation in the total number of base classifiers within 5-fold CV. This investigation was limited to the SVM classifier. Table 5 provides the average and standard error estimates of the standard deviation of the number of base classifiers within 5-fold CV over ten replications in ToxRefDB. The standard deviation was computed by using the fifth part of the training set which was not used for training. The smaller variation within CV was observed when the unweighted average aggregation rule $\mathcal{F}_1$ was applied (average standard deviation was 4.00 and 1.01 for the linear and the radial basis kernels, respectively) compared to the LDA-type aggregation rule $\mathcal{F}_2$ (average standard deviation was 9.56 and 10.87 for the linear and the radial basis kernels, respectively). In this study,

Table 6: Best-performing kernel functions and C values selected by 5-fold CV using pre-training set for Rat LD_{50} data [Linear=linear and RBF=radial basis functions].

Category	Kernel	C value	Category	Kernel	C value	Category	Kernel	C value
(1)	RBF	2^1	(1,8)	RBF	2^5	(4,5)	Linear	2^{-1}
(2)	RBF	2^7	(1,9)	RBF	2^1	(4,6)	RBF	2^1
(3)	RBF	2^5	(2,3)	RBF	2^1	(4,7)	RBF	2^5
(4)	RBF	2^1	(2,4)	RBF	2^7	(4,8)	RBF	2^{10}
(5)	RBF	2^1	(2,5)	RBF	2^7	(4,9)	RBF	2^1
(6)	RBF	2^1	(2,6)	RBF	2^{10}	(5,6)	RBF	2^1
(7)	RBF	2^1	(2,7)	RBF	2^7	(5,7)	RBF	2^5
(8)	RBF	2^7	(2,8)	RBF	2^7	(5,8)	Linear	2^{-5}
(9)	RBF	2^1	(2,9)	RBF	2^7	(5,9)	RBF	2^1
(1,2)	RBF	2^{10}	(3,4)	RBF	2^1	(6,7)	RBF	2^7
(1,3)	RBF	2^1	(3,5)	RBF	2^1	(6,8)	Linear	2^{-5}
(1,4)	RBF	2^1	(3,6)	RBF	2^1	(6,9)	RBF	2^1
(1,5)	RBF	2^{10}	(3,7)	RBF	2^5	(7,8)	RBF	2^{10}
(1,6)	RBF	2^5	(3,8)	RBF	2^{10}	(7,9)	RBF	2^7
(1,7)	RBF	2^7	(3,9)	RBF	2^1	(8,9)	RBF	2^5

Note: Numbers in parentheses indicate feature categories as follows:

1: 0D-constitutional descriptors	2: 2D-topological descriptors	3: 2D-walk and path counts
4: 2D-connectivity indices	5: 2D-autocorrelations	6: 2D-eigenvalue-based indices
7: 1D-functional group counts	8: 2D-atom-centered fragments	9: 2D-molecular properties

5-fold CV was selected considering the small sample size and computational cost.

6.2 Rat LD_{50} data

Since we have a large sample size for Rat LD_{50} data, we considered the best parameter configurations as suggested by a referee, focusing on the SVM. First, the data set was randomly split into two sets ($20\% : 80\%$) and the 20% portion (pre-training) was used to find the best-performing configuration. Considering the possibility of different data structures for each feature category, the best-performing configuration for each feature category was investigated in the IDC and the IDC stacking methods. Being able to apply a heterogeneous parameter set-up for different feature categories is an additional advantage of the IDC method. Linear and radial basis functions were investigated by

Table 7: Averages (ACC) and standard error estimates (SEE) of prediction accuracies over ten replications for Rat LD_{50} data.

Method	ACC	SEE
Single SVM (best-performing set-up)	0.763	0.006
Bootstrap SVM combined by $\mathcal{F}_1$ (best-performing set-up)	0.764	0.005
Bootstrap SVM combined by $\mathcal{F}_2$ (best-performing set-up)	0.599	0.047
Stacked Bootstrap SVM (best-performing set-up) combined by LR	0.740	0.005
Stacked Bootstrap SVM (best-performing set-up) combined by NN	0.736	0.006
$\text{IDC}_{\mathcal{F}_1}$ (best-performing set-up for each feature category)	0.844	0.005
$\text{IDC}_{\mathcal{F}_2}$ (best-performing set-up for each feature category)	**0.848**	0.005
IDC_{LR} (best-performing set-up for each feature category)	0.842	0.005
IDC_{NN} (best-performing set-up for each feature category)	0.837	0.006

nested 5-fold cross-validation over C values $(2^{-5}, 2^{-1}, 2^{1}, 2^{5}, 2^{7}, 2^{10})$. The hyperparameter σ in the radial basis function was determined by default ("automatic") which utilizes the heuristic approach to calculating a good σ value. The single SVM classifier performed best with the radial basis functions and C value of 2^7. The best-performing kernel function and C value for each feature category are given in Table 6. Once the best parameter set-up was determined, the remaining data set was randomly split into three sets: 50% for training, 25% for validation, and 25% for testing. Ten replications were obtained by random splitting and the average prediction accuracies of the ten replications were compared.

We compared the IDC and IDC stacking methods with the bootstrap SVM and the stacked bootstrap SVM methods. Suppose n_T and n_V denote sample size for the training and validation data sets, respectively. For bootstrap SVM, we drew 100 random samples of $n_T + n_V$ with replacement from the combined training and validation set. For each bootstrap sample, a base SVM classifier was built with the best-performing parameter set-up obtained for the single SVM. To combine 100 bootstrap base classifiers, $\mathcal{F}_1$ and $\mathcal{F}_2$ were applied. For stacked bootstrap SVM, we drew 100 random samples of n_T with replacement from the training data set and built 100 base classifiers such as bootstrap SVM. Again, L_2 penalized logistic regression with stepwise selection and a single layer NN were utilized to learn the aggregation rule from the validation set. By comparing the IDC methods with the bootstrap SVM methods where diversity among base classifiers was obtained by bootstrap resampling, we can see the effect of using the interactive relationship between feature categories with the forward selection procedure in the decision committee method.

Table 7 summarizes the analysis results for the Rat LD_{50} data. The IDC method with the best-performing set-up for each feature category achieved marked improvement in the average prediction accuracies (RI: 11.14%) with similar standard error estimates with the single SVM built in the best-performing set-up. We observed that both the IDC and the IDC stacking methods outperformed the

Table 8: Top 50 chemicals predicted to be toxic by the best IDC method with SVM for Rat LD_{50} data. $\hat{P}(\text{toxicity})$ denotes the average probability of toxicity over ten replications.

Rank	SOURCE_NAME_SID	$\hat{P}(\text{toxicity})$	Rank	SOURCE_NAME_SID	$\hat{P}(\text{toxicity})$
1	89426971	0.848	26	32527552	0.732
2	89427032	0.839	27	90220147	0.728
3	89427009	0.834	28	89427485	0.727
4	88909960	0.824	29	21270031	0.723
5	89427134	0.822	30	32358080	0.722
6	89427236	0.821	31	28548085	0.714
7	89427178	0.796	32	21327311	0.713
8	89427463	0.785	33	91893640	0.708
9	89427247	0.784	34	5954905	0.707
10	89457090	0.774	35	123252993	0.697
11	33399007	0.772	36	54504700	0.695
12	89457103	0.769	37	16499755	0.694
13	89427350	0.767	38	39184593	0.692
14	3309771	0.760	39	66215278	0.691
15	89427430	0.757	40	40693047	0.689
16	86811463	0.753	41	3696239	0.686
17	42576023	0.750	42	52549174	0.686
18	85977497	0.750	43	50765894	0.686
19	55391349	0.749	44	73561963	0.682
20	35944833	0.749	45	74124019	0.681
21	90293508	0.748	46	40596698	0.680
22	64050562	0.745	47	35317794	0.680
23	92065833	0.741	48	18877899	0.680
24	35972500	0.738	49	66232288	0.677
25	22224926	0.733	50	50497	0.676

bootstrap SVM and the stacked bootstrap SVM methods. This result demonstrates that the decision committee method can improve the classification performance by considering the interactive relationship between feature categories. Averages and standard error estimates of the number of base classifiers to be combined over ten replications are given in Table 4. Aggregation rule $\mathcal{F}_1$ combined a smaller number of base classifiers with less variation than the LDA-type aggregation rule $\mathcal{F}_2$ which is similar to the results in the ToxRefDB data set. As pointed out by a referee, presenting risk scores would be more informative for prioritizing compounds for further experimental assays. Table 8 provides chemical compound ID in Rat LD_{50} data and the average probability of toxicity of the top 50 chemicals predicted by $IDC_{\mathcal{F}_2}$ as an example.

In summary, we observed that the IDC and IDC stacking methods can improve prediction accuracy by considering interactive effects among categories of feature variables in both data sets. It is interesting to note that both the IDC and IDC stacking methods failed to improve classification performance for a few endpoints. As Shipp and Kuncheva (2002) noted, the decision committee method can perform worse than a single classifier due to dependency among base classifiers. Wang et al. (2009) also argued that the performance of the decision committee method depends on the data characteristics and showed through empirical experiments that the decision committee methods are not always better than a single classifier applying SVM. Due to the complicated aggregation mechanism of the decision committee methods, it is not obvious why the IDC methods or the IDC stacking methods performed worse than single classifiers for a few endpoints. However, simulation studies in the previous section already showed that the IDC method can perform similar to a single classifier in some cases. Also, it is not surprising that selecting base classifiers through forward selection with 5-fold CV works better than stacked generalization in many endpoints as shown in this data example. It is possible that we can improve the classification performance of the IDC stacking method by finding a more sophisticated, optimized learning algorithm to learn an aggregation rule, as suggested by Wolpert (1992).

7 Discussion

In this article, we proposed an interactive decision committee method that relies on different pairs of existing categories of feature variables as well as marginal feature categories and two-stage (SVM) or single stage (random forests and AdaBoost) 5-fold cross-validation with forward selection. The IDC method was applied to two sets of chemical toxicity data, ToxRefDB and Rat LD_{50}, consisting of binary endpoints and a set of feature variables from ten chemical descriptor blocks. For simple comparison purposes, a stacked generalization with the IDC method as well as a single unaggregated classifier were applied to the same data set. The basic idea and computation of the IDC method is very simple, but the IDC method and the stacked generalization IDC method can improve prediction accuracies compared to a single classifier in the chemical toxicity data sets applying SVM, random forests, and AdaBoost. Although the basic idea of the IDC method does not depend on the type of classifier inducer, the numerical examples show that the performance and the selection of the base classifiers can differ by learning algorithms.

Our method is in the early stages of development, and there are many possible ways to improve it. First, the current IDC method can be extended to resolving multiclass classification problems (Hsu and Lin, 2002) or to predict continuous outcomes (Budka and Gabrys, 2010). Second, it would be helpful to determine whether the improvement achieved by the IDC method in this paper can be observed in other types of data, such as gene expression data with gene categories. Liu et al. (2004) showed that a combinational feature selection with an ensemble neural network based on individual genes improved a classification task. Since we searched for all second order interaction terms between feature categories, the current IDC method would be inefficient for a large number of categories. Gene pathways are numerous, so we would need a more efficient way to select second order interaction terms between gene pathways. When feature categories can be defined in multiple ways, the best choice of feature categories is an open problem. When we have a data set with multivariate binary outcomes which might be associated with each other like ToxRefDB, applying the multivariate IDC method could be interesting. However, it is not clear how to implement multivariate modeling in the decision committee system, and we leave the study of the IDC method for multivariate outcomes for future research. One can consider higher order interactive relationships between feature categories for increasing diversity, although we only considered second order interactive feature spaces in this article. As pointed out by a referee, adding higher order interactive relationships should be carefully investigated in terms of overall performance since the additional benefit by adding higher order interactive relationships may be little relative to computational cost. Finally, it would also be interesting to integrate bootstrap resampling techniques with the IDC method in order to increase diversity, thus potentially achieving better prediction performance similar to Assareh et al. (2008) and Stefanowski (2005).

Like other decision committee methods, the IDC method provides little insight into the decision-making process, and thus limited interpretation of the results could be made (Dietterich, 1997). Despite this limitation, our work in this paper demonstrated that the proposed method improves classification performance compared to a single, unaggregated classifier. This study suggests that the proposed IDC method with two-stage or single-stage 5-fold CV could be useful for classification problems when high-dimensional feature variables are grouped into feature categories. Also, the proposed method could be very useful in improving the QSAR classification models, providing a useful tool for predicting hazards of chemicals, and prioritizing compounds for experimental assays.

Acknowledgements

Chaeryon Kang was supported in part by NIH grant RR025747 which partly funds the North Carolina Translational and Clinical Sciences Institute. Hao Zhu was supported in part by grants from The Society of Toxicology (11-0897) and The Johns Hopkins Center for Alternatives to Animal Testing (2010-17). Fred A. Wright was supported in part by NIH grant CA142538 and by the Carolina Environmental Bioinformatics Center (EPA STARRD832720). Michael R. Kosorok was supported in part by NIH grant RR025747, which partly funds the North Carolina Translational and Clinical Sciences Institute, and by NIH grant CA142538. The authors thank the editor and referees for helpful comments that led to a significantly improved paper.

References

Abeel, T., T. Helleputte, Y. Van de Peer, P. Dupont, and Y. Saeys (2009). Robust biomarker identification for cancer diagnosis using ensemble feature selection methods. *Bioinformatics 23*(19), 2507–2517.

Ali, K. and M. Pazzani (1996). Error reduction through learning multiple descriptions. *Machine Learning 24*(3), 173–202.

Ashburner, M., C. Ball, J. Blake, D. Botstein, H. Butler, J. Cherry, A. Davis, K. Dolinski, S. Dwight, J. Eppig, et al. (2000). Gene ontology: tool for the unification of biology. *Nature genetics 25*(1), 25–29.

Assareh, A., M. Moradi, and L. Volkert (2008). A hybrid random subspace classifier fusion approach for protein mass spectra classification. *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, 1–11.

Bauer, E. and R. Kohavi (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning 36*(1), 105–139.

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

Breiman, L. (2001). Random forests. *Machine learning 45*(1), 5–32.

Budka, M. and B. Gabrys (2010). Ridge regression ensemble for toxicity prediction. *Procedia Computer Science 1*(1), 193–201.

Chang, C.-C. and C.-J. Lin (2001). *LIBSVM: a library for support vector machines*.

Clemen, R. (1989). Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting 5*(4), 559–583.

database, C. (2008). National Library of MEdicine.

Dietterich, T. (1997). Machine-learning research. *AI magazine 18*(4), 97.

DRAGON (2006). *DRAGON for Windows (Software for Molecular Descriptor Calculations). Talete s.r.1., Milan (Italy)*.

Dutta, D., R. Guha, D. Wild, and T. Chen (2007). Ensemble feature selection: consistent descriptor subsets for multiple QSAR models. *J. Chem. Inf. Model 47*(3), 989–997.

Freund, Y. and R. Schapire (1997). A desicion-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences 55*, 119–139.

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

Ho, T. (1998). The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*(8), 832–844.

Hsu, C. and C. Lin (2002). A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on 13*(2), 415–425.

Johnson, S. (2008). The trouble with QSAR (or how I learned to stop worrying and embrace fallacy). *J. Chem. Inf. Model 48*(1), 25–26.

Karatzoglou, A., D. Meyer, and K. Hornik (2006). Support vector machines in R. *Journal of Statistical Software 15*(9), 1–28.

Krogh, A. and J. Vedelsby (1995). Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 231–238.

Kuncheva, L., J. Bezdek, and R. Duin (2001). Decision templates for multiple classifier fusion: an experimental comparison. *Pattern Recognition 34*(2), 299–314.

Lam, L. (2000). Classifier combinations: implementations and theoretical issues. *Multiple classifier systems*, 77–86.

Liaw, A. and M. Wiener (2002). Classification and regression by randomforest. *R News 2*(3), 18–22.

Lin, C., J. Moré, et al. (1999). Newton’s method for large bound-constrained optimization problems. *SIAM Journal on Optimization 9*(4), 1100–1127.

Liu, B., Q. Cui, T. Jiang, and S. Ma (2004). A combinational feature selection and ensemble neural network method for classification of gene expression data. *BMC bioinformatics 5*(1), 136.

Mangasarian, O. and D. Musicant (1999). Successive overrelaxation for support vector machines. *IEEE Transactions on Neural Networks 10*(5), 1032–1037.

Martin, M., R. Judson, D. Reif, R. Kavlock, and D. Dix (2009). Profiling chemicals based on chronic toxicity results from the US EPA ToxRef Database. *Environmental health perspectives 117*(3), 392.

Martin, M., E. Mendez, D. Corum, R. Judson, R. Kavlock, D. Rotroff, and D. Dix (2009). Profiling the reproductive toxicity of chemicals from multigeneration studies in the Toxicity Reference Database (ToxRefDB). *Toxicological Sciences*.

OECD (1996). Acute Oral Toxicity-Acute Toxic Class Method. *OECD (Organisation for Economic Co-operation and Development) Guideline for Testing of Chemicals* (No. 423).

Opitz, D. (1999). Feature selection for ensembles. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI)*, pp. 379–384. John Wiley & Sons LTD.

Opitz, D. and R. Maclin (1999). Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research 11*(1), 169–198.

Park, M. and T. Hastie (2008). Penalized logistic regression for detecting gene interactions. *Biostatistics 9*(1), 30.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. Vienna, Austria.

Richard, A. (2006). Future of ToxicologyPredictive Toxicology: An Expanded View of Chemical Toxicity. *Chem. Res. Toxicol 19*(10), 1257–1262.

Ripley, B. (2008). *Pattern recognition and neural networks*. Cambridge Univ Pr.

Sedykh, A., H. Zhu, H. Tang, L. Zhang, A. Richard, I. Rusyn, and A. Tropsha (2011). Use of in vitro hts-derived concentration–response data as biological descriptors improves the accuracy of qsar models of in vivo toxicity. *Environmental health perspectives 119*(3), 364.

Shin, H. and M. Markey (2006). A machine learning perspective on the development of clinical decision support systems utilizing mass spectra of blood samples. *Journal of Biomedical Informatics 39*(2), 227–248.

Shipp, C. and L. Kuncheva (2002). Relationships between combination methods and measures of diversity in combining classifiers. *Information Fusion 3*(2), 135–148.

Sigletos, G., G. Paliouras, C. Spyropoulos, and M. Hatzopoulos (2005). Combining information extraction systems using voting and stacked generalization. *The Journal of Machine Learning Research 6*, 1751–1782.

Stefanowski, J. (2005). An experimental study of methods combining multiple classifiers-diversified both by feature selection and bootstrap sampling. *Issues in the Representation and Processing of Uncertain and Imprecise Information, Akademicka Oficyna Wydawnicza EXIT, Warszawa*, 337–354.

Stouch, T., J. Kenyon, S. Johnson, X. Chen, A. Doweyko, and Y. Li (2003). In silico ADME/Tox: why models fail. *Journal of computer-aided molecular design 17*(2), 83–92.

Tuv, E., A. Borisov, G. Runger, and K. Torkkola (2009). Feature selection with ensembles, artificial variables, and redundancy elimination. *The Journal of Machine Learning Research 10*, 1341–1366.

Vale, K., F. Dias, A. Canuto, et al. (2008). A class-based feature selection method for ensemble systems. In *Eighth International Conference on Hybrid Intelligent Systems*, pp. 596–601. IEEE.

van der Heijden, F., R. Duin, D. De Ridder, and D. Tax (2004). *Classification, parameter estimation and state estimation*. Wiley Online Library.

Vapnik, V. N. (1998). *Statistical learning theory*, Volume 2. Wiley New York.

Walum, E. (1998). Acute oral toxicity. *Environmental Health Perspectives 106*(Suppl 2), 497.

Wang, S., A. Mathew, Y. Chen, L. Xi, L. Ma, and J. Lee (2009). Empirical analysis of support vector machine ensemble classifiers. *Expert Systems with Applications 36*(3), 6466–6476.

Wolpert, D. (1992). Stacked generalization*. *Neural networks 5*(2), 241–259.

Zhu, H., T. Martin, L. Ye, A. Sedykh, D. Young, and A. Tropsha (2009). Quantitative Structure-Activity Relationship Modeling of Rat Acute Toxicity by Oral Exposure. *Chemical research in toxicology 22*(12), 1913–1921.