# Machine Learning in Virtual Screening

James L. Melville[1,†], Edmund K. Burke[2] and Jonathan D. Hirst[*,1]

[1]*School of Chemistry, University of Nottingham, University Park, Nottingham, NG7 2RD, UK*

[2]*School of Computer Science & Information Technology, University of Nottingham, Jubilee Campus, Nottingham, NG8 2BB, UK*

**Abstract:** In this review, we highlight recent applications of machine learning to virtual screening, focusing on the use of supervised techniques to train statistical learning algorithms to prioritize databases of molecules as active against a particular protein target. Both ligand-based similarity searching and structure-based docking have benefited from machine learning algorithms, including naïve Bayesian classifiers, support vector machines, neural networks, and decision trees, as well as more traditional regression techniques. Effective application of these methodologies requires an appreciation of data preparation, validation, optimization, and search methodologies, and we also survey developments in these areas.

**Keywords:** Machine learning, virtual screening, data mining, drug discovery.

## INTRODUCTION

Machine learning is a branch of artificial intelligence. A common task is the assignment of objects (also called observations or instances) into classes. In chemoinformatics [1], the objects are usually molecules, and the classes are categorical and often dichotomous. For example, machine learning is widely used to classify molecules as inactive or active against a particular target. However, classification is not restricted to binary choices; the classification may also be numerical, e.g. predicting $IC_{50}$ values. The statistical term "regression" is used to describe those learning problems. For either classification or regression, the usual machine learning presents an algorithm with examples of molecules that are already known to be active or inactive. This process is called "training" the algorithm. Learning problems that require the data to be pre-assigned to a fixed number of classes are called "supervised". Alternatively, there may be situations when no pre-assignment of the classes is possible or desired. An example is selecting a small number of molecules from a larger database, while attempting to retain the overall properties of the entire set. Each chosen molecule can be thought of as a representative of a number of other molecules in the larger set. The ability to partition a dataset in this way implies the existence of different classes of molecule, although the number of classes is unlikely to be known *a priori*, and the physical meaning of the classes is unlikely to be as obvious as the activity. Such learning tasks are known as "unsupervised".

In this review we will examine the application of machine learning to virtual screening (VS). Virtual screening is the evaluation of a molecular property (such as activity) *in silico*, the computational analogue of high throughput screening (HTS). In virtual screening a binary active/inactive classification is a more useful starting point than regression methods. HTS assays are rarely accurate enough to provide real-valued affinity data (although this is beginning to change [2]), and collating large datasets therefore necessitates mixing data from different labs, perhaps obtained under different experimental conditions; this may introduce errors which can rarely be controlled or accounted for. However, for VS to be a useful tool in the service of prospective drug discovery, one normally wishes to prioritize molecules, and only suggest a small subset of the database for experimental screening. Therefore, it is also desirable to provide a numerical value for each molecule in a database that can be used for ranking. Such numerical values can be trivially converted to classifications by applying a threshold. This adds fuzziness to the basic classification problem and algorithms that cannot provide such a ranking may not be suitable for some VS purposes. However, when screening for pharmacokinetic properties or potential toxicity, a binary classification can be helpful.

A broad definition of machine learning and virtual screening would encompass nearly all aspects of chemoinformatics. Therefore, in this review we concentrate on supervised learning, which naturally lends itself to the fundamental task of drug discovery, the partitioning of molecules into active and inactive categories. The vast majority of these applications use the statistical machine learning paradigm, so we focus exclusively on these applications. However, alternative approaches exist, most notably inductive logic programming, applied by the groups of King [3-6] and Sternberg [7]. Additionally, we bias our survey towards recent and novel developments which have been applied to prioritizing large datasets, to mirror closely the practices of HTS, although we mention studies on smaller datasets where the extension to HTS-scale data is obvious. Our survey considers each major algorithm in turn and in each section, we will describe relevant applications in both ligand-based and structure-based contexts. Optimization is an important part of many virtual screening techniques, especially if they employ machine learning; we therefore also review new developments in this field. We shall briefly describe each algorithm,

*Address correspondence to this author at the School of Chemistry, University of Nottingham, University Park, Nottingham, NG7 2RD, UK; Tel: +44-115-951-3478; Fax: +44-115-951-3562;
E-mail: jonathan.hirst@nottingham.ac.uk

†Current address: Cresset BioMolecular Discovery Ltd, BioPark Hertfordshire, Broadwater Road, Welwyn Garden City, Hertfordshire, AL7 3AX, UK

but direct the reader to the relevant references in the text for full details. Good overviews of machine learning as a whole can be found in books by Mitchell [8], Hastie *et al.* [9] and Witten and Frank [10].

## LINEAR METHODS

Probably the most familiar machine learning method in chemoinformatics is multiple linear regression (MLR), which has been used widely in quantitative structure-activity relationships (QSAR) [11]. 3D QSAR techniques, such as CoMFA [12], make use of a large number of descriptors, and this has driven the adoption of partial least squares (PLS) [13], which can be thought of as the application of MLR to mutually orthogonal linear combinations of the original descriptor set. Despite the long history of QSAR, the suitability of linear methods such as PLS for large scale VS is uncertain. Usually, QSAR is used as a lead optimization tools, and normally use substantially less than 100 molecules. A dearth of publicly available data has also hampered development. However, recently, Gedeck and co-workers at Novartis carried out a large-scale evaluation on 944 corporate datasets with $IC_{50}$ data using PLS in conjunction with a variety of 2D and 3D descriptors [14]. The predictivity of the linear models was modest. Even the best performing descriptors (fragment counts) predicted more than 50% of the variance in the test set in no more than 30% of the datasets. Additionally, to get these results, at least 50% of the molecules in each dataset (on average 600 molecules, although there were some datasets with 1000s of molecules) were used to train the PLS model. Therefore, even modest predictivity of $IC_{50}$ values requires substantial amounts of quantitative biological data, which makes such approaches inappropriate in many virtual screening scenarios. Nonetheless, Evers and co-workers at Aventis demonstrated [15] that, when searching for biogenic amine-binding G-protein coupled receptors (GPCR) ligands from the MDL drug data report (MDDR), PLS regression and the classification version of PLS, PLS-discriminative analysis (PLS-DA), using 2D topological and physicochemical descriptors, were superior to docking into homology models with GOLD, although the PLS models did require training on several hundred actives, taken from the Aureus database (www.aureus-pharma.com).

An important prerequisite for the use of many 3D QSAR approaches has been the generation of a meaningful alignment of the molecules in the dataset. Where a suitable protein structure is available, docking has been used to generate this alignment [16]. However, beyond providing an alignment, the protein structure was not used for any other part of the QSAR derivation. An obvious extension is to use the protein structural information directly in the learning algorithm, at which point, the boundaries between 3D QSAR and docking begin to dissolve, where scoring functions can be thought of as structure-based QSAR making use of both protein and ligand-based descriptors. The combination of different scoring functions, called data fusion [17] or consensus scoring [18], is an extension of this principle. The majority of data fusion applications involves weighted voting and combining ranks. These techniques are sufficiently straightforward that we do not consider them to be machine learning. Linear models of regression and classification have been used to combine scoring functions, although normally they are included only for comparative purposes (usually unfavorably) with non-linear methods. One exception is the study of Catana and Stouten [19], who use a multi-way extension to PLS, N-PLS [20, 21], to create new scoring functions.

An excellent example of a hybrid of a QSAR and scoring-function is the adaptation of fields for molecular comparison (AFMoC) method [22], which replaces the receptor-agnostic fields of 3D QSAR techniques such as CoMFA and CoMSIA with fields derived from the structure of the protein, using the DrugScore scoring function. Both the original authors [22-24] and others [25] have used the AFMoC to produce tailored scoring functions.

Docking, despite being preferable to experimental evaluation of a large database, can still consume substantial computational resources, which may preclude its use in some screening situations. Assuming that one believes that docking produces valid results, docking scores are now valid targets for other machine learning techniques, as long as they are faster to train and to predict than the docking method. Cherkasov and co-workers described a method they call "progressive docking" [26], which hybridizes QSAR and docking in this fashion. A subset of the database is docked to a target in the usual fashion, and then a QSAR is built to predict the docking scores, using 2D descriptors. The resulting model can then be used to predict the scores of the remaining compounds. Low scoring compounds are removed from consideration, and potentially good candidates docked. The process can then be repeated with the new data. The authors demonstrated the technique's efficacy by docking 90,000 molecules to human sex hormone-binding globulin (SHBG) with the program Glide [27], in batches of 10,000, and the results used to build a QSAR model using the "inductive" descriptors developed by the Cherkasov group [28]. The 20% of compounds with the poorest predicted docking score were removed from consideration, before the process was iterated. Docking times could be reduced by several hundred hours, a time saving of 40%, while four structures from the top 16 non-steroidal ligands identified by the method were found to be SHBG binders with low micromolar affinities.

A more indirect use of machine learning in docking was provided by Jacobsson and Karlén [29], who used PLS to correct the well-known size bias of many scoring functions [30, 31]. Because scoring functions often involve summing pairwise contributions between atoms of the ligand and protein, larger molecules will tend to have better docking scores, simply by virtue of having more atoms. To account for this, Vigers and Rizzi [32] suggested docking molecules across several different protein targets, against the majority of which the ligand is assumed to be inactive. The mean of these scores, the multiple active site correction (MASC) represents a baseline value that should be subtracted from all future scores against other targets for that ligand and scoring function. Jacobsson and Karlén demonstrated that the MASC could be well approximated by a simple PLS model using only simple molecular descriptor data such as molecular weight (MW) and ClogP. Across eight targets, using Glide and most of the currently popular scoring functions, the PLS-MASC could improve enrichments of known actives from a decoy database of 10,000 molecules.

Vieth and Cummins presented a method that uses machine learning and docking to elucidate the docking mode that is consistent with a structure-activity relationship (DoMCoSAR) [33]. Repeated dockings of a dataset are carried out and the multiple results clustered to generate candidate binding modes. The molecules in each pose are then characterized by four simple descriptors: the van der Waals and electrostatic interaction energies, the internal strain of the molecule, and the exposed surface area. PLS is used to generate a QSAR for each mode, and the best model is assumed to be the bioactive binding mode. The authors reported success for two small datasets, although they point out that the technique may only be applicable to congeneric series of molecules.

Although not strictly a linear classification technique, simple nonlinear methods such as logistic regression have found some limited success in VS, although the more sophisticated nonlinear methods we discuss below are now preferred. Nonetheless, for four GPCR targets, Feher and co-workers [34] used consensus scoring to combine several very different ligand-based approaches using 2D descriptors, e.g. fingerprints and BCUT [35], and 3D pharmacophore models using HypoGen [36]. Results showed that the sum rank method was superior to any individual scoring function, but training a logistic regression method using 1000 molecules and 25% of the known actives could improve performance further.

## NEAREST NEIGHBORS

Conceptually one of the simplest approaches to predict the activity of a molecule is to search a database for the molecule that is most similar to it. The predicted value is the activity of this ‘nearest neighbor’. An obvious extension to this scheme is to use more than one neighbor. This method is known as ‘*k* nearest neighbor’ (*k*NN). In general, *k* is fixed (a value between 1 and 10 is common), or may be chosen *via* cross-validation. Sometimes, *k* for each molecule is chosen based on a distance cutoff, as used by Helma in the toxicity prediction system, Lazar [37]. *k*NN approaches are ‘lazy’: training is deferred until a prediction is required. The *k*NN method is an effective method of machine learning, and is still popular. In particular, Tropsha and co-workers have demonstrated its efficacy over several datasets, including its use to identify nine potential anti-convulsants out of a database of 250,000 molecules, seven of which were confirmed to be active [38]. The same group also uses *k*NN as the learning algorithm in their CoLiBRI [39] and ENTess [40] methods, which include descriptors representing the receptor as well as the ligand. Several attempts have been made to improve on the *k*NN method. The most common modification is to take into account the distance between the molecule to be predicted and its neighbors, where the activities of closer neighbors are given more weight. Several different functional forms for the weighting equation (normally referred to as a ‘kernel’) have been explored. Jensen *et al*. used a Gaussian weighting function of the Tanimoto distance between molecules to predict the activity of 1,000 inhibitors of cytochrome P450 2D6 and 3A4 [41]. Cedeño and Agrafiotis employed an inverse distance kernel (with an additive correction to avoid singularities) in a QSAR study [42]. A further refinement to such methods is to weight the contribution of each descriptor to the distance calculation. Miller described a variation on *k*NN classification, called modified flexible metric nearest neighbor (MFMNN) [43], where the descriptor weights are determined by how well each descriptor partitions the nearest neighbors of the test instance into homogeneous classes. This was more effective than a standard *k*NN when screening the NCI AIDS dataset. In their kScore algorithm [44], Oloff and Muegge used an inverse square Euclidean distance function to optimize the weighting of each descriptor by a conjugate gradient or steepest descent method. Using atom pair descriptors for an in-house database of 10,200 compounds assayed against a kinase, kScore ranking of 775,000 molecules from an HTS campaign yielded an enrichment factor of 35 for the top 1% of the database.

Weighted averaging is the simplest method of combining the data points to make new predictions. An obvious extension is to build classification and regression models using only the neighbors. Such techniques are referred to as ‘lazy local’ learning methods, as opposed to ‘global’ methods which use the entire dataset. Lazy local methods have been applied to chemoinformatics by Kumar *et al*. [45], Guha *et al*. [46] and Zhang *et al*. [47].

Additionally, our group has made use of several types of weighted nearest neighbor and local regression for QSAR [48-50]. However, the application of kernel methods to the largest datasets [51-57] has involved binary kernel discrimination (BKD). These use kernels specifically adapted for binary fingerprint data as descriptors. Recently, Willett and co-workers have compared BKD to the performance of a Gaussian kernel for continuous-valued descriptors using the Parzen window approach [58]. Using datasets taken from the MDDR, trained on 100 actives and 4,000 inactives, they found that application of continuous kernel discrimination (CKD) with hologram descriptors (i.e. counts of structural features) was superior to physicochemical properties (e.g. atom counts, polar surface area, molecular weight, log *P* etc.) and topological descriptors, but this superiority was greatly reduced if the dataset consisted of fairly heterogeneous actives. However, there was little difference in performance compared to using BKD. Both the CKD and BKD performed comparably to the use of support vector machines (SVM) using radial basis function (RBF) kernels (vide infra).

## NAÏVE BAYESIAN CLASSIFICATION

Naïve Bayesian classification is one of the simplest classifiers. Each descriptor value of a test set molecule is considered in turn. The probability of activity is considered to be proportional to the ratio of actives to inactives that share the descriptor value. The overall probability of activity is simply the product of these probabilities. This approach therefore assumes that each descriptor is statistically independent. This is clearly not the case for most datasets and descriptor sets used in virtual screening; however, theoretical results suggest that large deviations from independence can be tolerated [59], and the overall ranking of a dataset will be unchanged (although the absolute values of the probabilities will be incorrect). Therefore, it is expected that NBC should be competitive in a VS scenario.

The earliest use of a NBC methodology is that of Binary QSAR, introduced by Labute [60], although the similarity of this method to NBC does not appear to have initially been appreciated. Binary QSAR differs from other uses of NBC in the way the descriptors are prepared: Principal Component

Analysis (PCA) is used to decorrelate the descriptors (note however, that this is *not* sufficient to ensure statistical independence), and after binning these new scores, a Gaussian function is used to smooth the contribution of each descriptor over adjacent bins, ameliorating any edge effects that might arise from the discretization. Few large-scale VS studies using binary QSAR have been carried out; however, a recent study by Prathipati and Saxena on the AChE dataset used by Jacobsson and co-workers in their study of supervised consensus scoring functions [61], indicated that the performance of binary QSAR was comparable to the performance of PLS-Discriminative Analysis (PLS-DA) [62]. One of the most prominent proponents of the use of the NBC in VS has been a group at Novartis. Glick and co-workers initially used the NBC with extended connectivity fingerprint (ECFP) descriptors as a post-processing tool, to prioritize hits from noisy experimental HTS data, where the NBC ranking enriched the top 10% of the data with 26%-45% of the genuine actives across four HTS campaigns [63]. This work was extended to use docking in place of the HTS as a source of potential actives [64]. The docking scores were converted to a two-class classification using a threshold (three standard deviations below the mean energy), rather than relying on *a priori* knowledge of actives, which allows the technique to be applied to any database of molecules. Using 179,805 molecules from the Available Chemicals Directory (ACD) as inactives, the enrichment of protein tyrosine phosphatase 1B (PTP-1B) inhibitors showed substantial increases in enrichment whether using FlexX, DOCK or Glide. However, no increase in enrichment was observed when searching for inhibitors of protein kinase B/Akt (PKB). The explanation for this difference was that the PTP-1B docking results already showed some enrichment, which was enhanced by the application of NBC. Conversely, docking failed to find any genuine actives when screening against PKB. This result suggests that the NBC works to enhance an existing “signal” in noisy data, but cannot work under conditions where the signal is effectively zero. The authors later showed [65] that consensus scoring using CScore and ranking by the median rank was able to create an initial enrichment for the PKB set, and training an NBC on the top 1% of the database increased the number of actives in the top 1% of the database from 24% to 64%. Further studies using HIV-1 protease as a target [66], demonstrated that the docking-guided NBC approach was superior to conventional similarity searching. The authors also compared the use of NBC with bagged decision trees and a RBF SVM [67], using data from four recent campaigns targeting protease, chemokine receptor and GPCR targets. Classifiers trained on 4,000 molecules, with 100-300 actives, were used to screen 170,000 molecules. In general, the NBC was comparable to bagged decision trees. However, the SVM was the best performer of the three methods. On increasing the level of false positives in the training set, the performance of the NBC and bagged decision tree was unaffected, but the SVM’s performance fell until it was no more effective than the NBC and decision tree. Recently, the authors have extended the use of the NBC to predict the biological target of a molecule [68], and to model numerical descriptors using a Gaussian distribution for ADME classifications [69]. Further use of the NBC has been made by Bender and co-workers, in conjunction with MOLPRINT2D, a 2D topological fingerprint, and variable selection using the information gain criterion, normally used as a splitting rule in the C4.5 decision tree methodology [70]. Using several datasets from the MDDR database, they demonstrated that the NBC showed superior performance to data fusion of standard similarity searching with bitstrings or holograms, and was competitive with BKD [71, 72]. Consistent with the conclusion of Willett and co-workers in their evaluation of the NBC, Bender and co-workers applied the NBC to screen 100,000 molecules from the McMaster University High-Throughput Data-Mining and Docking Competition [73], and concluded that the NBC is not suited to similarity searching in datasets which are structurally heterogeneous compared to the training data [74]. The technique has also been extended to use 3D structures with MOLPRINT3D, an alignment independent 3D descriptor based on binned energy evaluations at a molecular surface [75] using probes from the GRID forcefield [76]. This was not as effective as MOLPRINT2D, but still outperformed several other similarity searching methods using the MDDR benchmarks. Willett and co-workers have also recently investigated the use of the NBC [58], comparing it to group fusion, a simple consensus score defined as the maximum similarity between a test set molecule and any training set molecule. In these experiments, the Extended Connectivity Fingerprint (ECFP4) implemented in Pipeline Pilot (SciTegic Inc., San Diego) were used as descriptors across 14 datasets taken from the MDDR, using 5 or 10 known actives as training set molecules and 200 inactive molecules. Results showed that the NBC is superior to group fusion only in the case where the actives are structurally homogeneous, and the inactives are heterogeneous. Filikov and co-workers also used the NBC in their “surrogate docking” approach [77], which aims to use QSAR models to predict docking scores, rather than bioactivity. It has strong similarities with, but predates the previously described “progressive docking” technique of Cherkasov and co-workers, without the iterative aspect. The authors demonstrated the approach in a retrospective screening simulation using ICM [78] to dock two datasets of size 85,000: a selection of molecules from the NCI screening library, and a CDK-focused combinatorial library. In both cases, a training set of 5,000 molecules was used, including only 50 known actives. The ECFP and functional connectivity fingerprints (FCFP) descriptors, combined with radial basis functions and NBC classifiers were used to build the QSAR models. In both cases, the authors found that the top 5,000 molecules in the 80,000 compound test set contained around 50 % of the docking hits. In general, the NBC has been found to be a capable algorithm. However, Svetnik and co-workers evaluated NBC against a large CDK2 dataset and it performed the least well of all the methods they studied [79].

## SUPPORT VECTOR MACHINES

Support vector machines (SVMs) were developed by Vapnik [80,81] and introduced into chemoinformatics by Burbidge and co-workers [82] and Czerminski and co-workers [83]. Apparently, classification by SVM is similar to simple methods such a linear discrimination. A separating hyperplane is defined and instances are classified depending on which side of the boundary they lie. When a ranking is required, the distance between an instance and the hyperplane is used. Important features of the SVM methodology are that the hyperplane is chosen to maximize the distance between the hyperplane and the nearest instance (normally referred to as the margin) and only a small subset of training instances (the support vectors) define the boundary. Additionally, noisy data or experimental errors can be accounted for by allowing some instances to be on the wrong side of the hyperplane. Each instance is then associated with a 'slack' variable, which is non-zero for misclassified instances. This penalty is included in the optimization, and allows a trade-off between the number of misclassifications and the size of the margin. An additional useful feature of SVMs is the application of the so-called 'kernel trick', which extends SVM classification from linear to non-linear hyperplanes, using Mercer's theorem [84,85] to choose suitable 'kernel' functions. These kernel functions allow the calculation of distances between instances in very high dimensional non-linear spaces, without requiring an explicit transformation of the instances themselves into the high dimensional non-linear space. Suitable functions that meet this requirement include polynomial kernels and radial basis function (RBF) kernels. For more details, see the tutorial of Burges [86]. A variety of decomposition algorithms have been developed [87-89], which have allowed the application of SVMs to large datasets. The SVM methodology has also been extended to regression (SVR) [90].

Initial work on the use of SVM in VS was carried out by the Willett group, in collaboration with Syngenta. Their results indicated that SVMs were markedly inferior to the use of BKD, when tested on the 35,991 molecules in the NCI AIDS data set, using UNITY fingerprints as descriptors [56]. However, later work on 125,657 compounds with measured pesticidal activity and a polynomial kernel of degree five, outperformed BKD [52]. Schneider and co-workers screened 2.7 million compounds from their COBRA database, using 94 known COX-2 inhibitors for training [91]. A number of compounds cherry picked from the top of the database, yielded three molecules with inhibitory activity, one of which achieved activity higher even than celecoxib and rofecoxib. Chuman and co-workers extracted 21 datasets related to depression (including serotonin, dopamine, adenosine and monoamino-oxidase ligands) from the MDDR [92], and classified (but did not rank) 30,000 molecules per dataset with an RBF SVM and an atom count descriptor. True negatives were well predicted, with at least 96% correctly classified; however, positive recall was variable, ranging from 44% to 89%. Jorissen and Gilson carried out a screening simulation for five receptor classes ($\alpha_{1A}$ AR, CDK2, COX2 and PDE5) [93], using 50 known actives for each target, and either 1,892 molecules from the NCI diversity set or 25,175 from the Maybridge database as inactives. Using an RBF SVM with 2D descriptors generated by the DRAGON program (http://www.disat.unimib.it/chm/), they retrieved more actives from the top 2% to 10% of the screened database than either BKD or data fusion methods using fingerprints. Saeh and co-workers at AstraZeneca used SVMs with a variety of kernels and 3D pharmacophore triplets on a mixture of proprietary and MDDR data for G-protein coupled receptors [94]. They were able to prioritize 1,573 compounds from a list of 129,994, yielding a hit rate 69 times higher than random selection. A genuinely prospective use of SVR in VS was provided by Schneider and co-workers [95], who used a model built from 331 dopamine receptors inhibitors to screen the SPECS and Interbioscreen databases (over 255,000 molecules combined) to suggest 11 compounds with a high selectivity for the D3 receptor. One weak hit was successfully optimized by similarity searching to a nanomolar affinity, with a tenfold preference in affinity for the D3 receptor over the D2. Interesting recent theoretical work on SVMs has been the development of kernels that can be applied to data with different structures. This is useful in virtual screening, where it is more natural to represent a molecule as a graph rather than a vector of descriptors. Recent work in this area has included the development of kernels to represent pharmacophores [96], trees [97] and graphs [98]. Kless and co-workers have also suggested new kernel functions, including those based on Slater-type orbitals and the Tanimoto coefficient [99], and demonstrated that the different costs associated with misclassified active and inactive molecules can be accounted for by using different slack variables for the different classes.

## NEURAL NETWORKS

Artificial neural networks (ANNs) have played a long-established role in chemoinformatics [100]. Two major types of neutral network are in use: feed-forward networks (which are supervised), and self-organizing maps (SOMs), which are mainly unsupervised. Both networks are made up of a series of connected neurons. A neuron takes multiple numerical inputs, and outputs a transformed, weighted sum of the inputs. Common transformation functions include the tanh and sigmoid functions. For feed forward networks, descriptors are used as inputs into the initial layer of neurons, and then further layers take their inputs from the output of the previous layers. The final layer contains the predictions, with one output neuron per predicted value. Therefore most feed forward networks used for virtual screening have only a single neuron in the output layer, although ANNs can be extended to predicting multiple properties simultaneously easily.

As the name suggests, the inspiration for the structure and working of ANNs is that of neurons firing in the brain, but statistically, it may be more constructive to note the similarities between them and projection pursuit regression [9], which mainly differ in the training procedure by which the weights are established. For feed-forward ANNs, the most usual technique is backpropagation [101], where the weights are initialized to random values, and then adjusted to reduce the error observed in the output, moving backwards from the layer closest to the output.

Several uses of ANNs to create consensus scoring functions for docking have been presented recently. GFScore is a neural network that combines GOLDscore, DOCKscore, PMFscore, ChemScore and FlexXScore, using 78 proteins from the FlexX dataset as a training set [102], and was shown to be superior to a linear combination of the same scoring functions. Other uses of neural networks to create new scoring functions have been reported by Barbault *et al.*, who trained an ANN using the AutoDock scoring function to create a scoring function tailored to RNA targets [103]. However, only eight molecules were used in the training set, and no large-scale validation was attempted. Sem and co-workers used ANNs to create a consensus score for predicting CYP2D6 [104] binding affinity with AutoDock and

XScore. Additionally, they used the classification accuracy of the ANN model to determine which of 20 conformations of the CYP2D6 binding site, generated by a molecular dynamics simulation, was likely to resemble the bioactive form. The stochastic nature of most neural network training methods means that the resulting models are not reproducible. Agrafiotis and co-workers suggested that this variance in output can be put to good use by combining the results of several neural ANNs together in a type of consensus scoring [105]. Although no details of any truly large scale screening of pharmaceutical data have been published, Seierstad and Agrafiotis have used ensembles of ANNs to derive QSARs for several hundred compounds with inhibitory activity against the human ether-á-go-go-related gene (hERG)-encoded cardiac potassium channel [106].

In contrast to feed forward ANNs, the architecture of a SOM [107] is that of a grid, rather than a series of layers. Each neuron is connected to its neighbors, and the output of which is an entire vector of values rather than a single value.

The dimensionality of the vector is the same as that of the instances used for training, and the values are normally initially randomly distributed. The process of training a SOM involves presenting each training instance to the network, and choosing a "winning" node, which has a vector most similar to the training instance's own descriptors. The vector of the winning node, and its neighbors, are updated to become more similar to that of the training instance, and the process is repeated until a stopping criterion is reached. In this way, the neurons take on the character of the training set. A trained SOM can then be used for clustering, by finding the neuron that most closely resembles a test set instance. In chemoinformatics, a SOM is often used as a classification tool – test set molecules that resemble neurons that are occupied by known actives are deemed also to be active. A closely related architecture is the counter-propagation neural network [100]. SOMs remain popular for classification and several examples of their successful use for VS on large datasets have been described. Selzer and Ertl used a 3D radial distribution function (RDF) descriptor to describe 27,000 molecules from the World Drug Index, including 1,700 GPCR ligands [108]. A counter-propagation ANN trained with these data recorded an enrichment factor of 13-fold for the top 1% of a test set of similar size to the training set. Ecker and co-workers trained a SOM with 131 propafenone-type inhibitors of P-glycoprotein, represented by autocorrelation descriptors and screened 134,767 compounds in the SPECS database. Seven out of 15 compounds chosen for experimental testing were active at micromolar and submicromolar concentrations [109, 110]. Schneider and Nettekoven built a SOM with topological atom pair descriptors to classify a small library of 153 molecules screened for purinergic human A2A receptor activity [111]. The ACD was screened to find 17 molecules similar to the actives in the training set, one of which displayed nanomolar activity towards A2A with a hundredfold selectivity over the A1 receptor.

Artificial neural networks have also been used for VS outside of the pharmaceutical area, for example in heterogeneous catalysis [112,113]. A novel application of neural networks for virtual screening was provided by Thaler [114], in the search for ultra-hard binary compounds. Thaler used an auto-associative neural network, in which the output is of the same dimensionality and type as the inputs, i.e. the network is trained to reproduce its inputs. Thaler trained one network with a simple representation of binary mixtures of elements and their relative stoichiometries. The network is then induced to produce novel mixtures by randomly perturbing the weights, and connecting the output of the network to the input, and allowing the network to iterate to a steady state. The author claims that this allows the creation of novel structures not found in the training set, but following the rules of stoichiometry encoded in the network. Further networks were trained to predict hardness. The study suffers from a lack of statistical or experimental validation, but remains a unique application of artificial neural network methodology.

## DECISION TREES

A decision tree represents the conjunction of a series of "rules", where each rule is a predicate concerning a set of descriptors. The order of application of the rules is fixed, and which rules are applied to an instance depends on whether the previous rule evaluated to true or false. When visualized, this branching path structure resembles an inverted tree. After all rule evaluation, a "leaf" is reached, where an assignment to a classification or a value is made. The process of training a decision tree determines which rules are included in the tree, in which order, and what classification to make at each leaf. The process is normally carried out by choosing a single rule that splits the training instances into two or more (not necessarily equally-sized) groups. The process is then repeated for each of the subsets, until a termination criterion is met, which gives rise to the technique's alternative name, recursive partitioning. In general, the training methods construct a rule using only one descriptor at a time, a binary split is made for each rule, and the choice of rule is based on producing a split that generates two subsets which are most homogeneous in terms of the response value (although other methods are possible, e.g. using evolutionary programming [115]). Decision trees can be used for regression purposes, but are most commonly used for classification. A method similar to decision trees is rule induction, or association rule mining, which also constructs a series of rules for classifying the data set. Unlike decision trees, the rule set need not display a hierarchical structure. Rule induction techniques have been applied almost exclusively for mining toxicology databases for detecting frequently occurring substructures; the reports of Kazius [116] and co-workers and Karwath and De Raedt [117] are recent examples. However, Clark and co-workers reported the use of rule induction for scaffold hopping during lead optimization of a library of 1,600 molecules screened for activity against respiratory tract infections [118].

The relatively few applications of rule mining in VS is perhaps surprising given how popular decision trees have been as classification tools in QSAR, in no small part due to the interpretable models they produce, despite representing a non-linear learning methodology. Applications to large screening datasets were published by workers at GlaxoSmithKline, who demonstrated its use as an integrated tool for iterative screening. Jones-Hertzog and co-workers [119] reported results for screening 23,000 compounds against 14 GPCR targets, outperforming random selection and similarity searching in the majority of cases. Rusinko and co-workers used a 2,500 member library displaying only a weak SAR as a training set [120], and returned 22 out of the 50 most active compounds after screening only 6% of a similarly sized test library. Other uses of decision trees for HTS data include Blower and co-workers, who combined decision trees with the Simulated Annealing search method for variable selection, applying it to the NCI cancer set [121], and Van Rhee and co-workers, who carried out a retrospective analysis of 20,000 compounds tested against an ion channel target, using a 5, 000 member training set, and increasing hit rates in primary screens by four fold [122]. Recently, Yamakazi and co-workers used a single decision tree, trained on 130 PDE-5 inhibitors and 10,000 inactives to screen 50,520 molecules in the SPECS database [123]. Seven of the 19 selected compounds showed inhibitory activity at 10 μm. Despite these successes, the predictions of decision trees are known to suffer from high variance, and the structure of the decision tree is sensitive to small changes to the training set. Therefore, the machine learning community has produced a series of techniques to ameliorate these problems, which we describe in the next section.

## ENSEMBLE METHODS

Ensemble methods describe a series of classifiers that combine the output of other classifiers (called "base classifiers") to come up with the final prediction. The ensemble methods that have proven most popular are those that combine classifiers of the same type, normally decision trees, although most of the following techniques are general and have also been widely applied to neural networks. Indeed, even heterogeneous collections of classifiers can be combined. Differences between the decision trees are introduced by training them with slightly different training sets, or making use of the stochastic nature of many training algorithms. Simple examples of the ensemble approach include the study of van Rhee and co-workers [124], who used pairs of decision trees to screen 3,000 molecules, which gave a 13 fold enrichment over the hit rate of a 14,000 member HTS when screening against an ion channel target. Miller presented a method where each tree is trained on a different subset of the training set, split on the basis of the ring systems [125]. When applied to the NCI cancer data set, it was as predictive as a single tree built on the entire data set, but was also more easily interpreted.

More advanced methods make use of the bootstrap [126], which creates subsamples of the training set with replacement, so that a molecule can appear more than once in the subset. This provides the perturbation to produce different models. This scheme is called bagging [127], and was introduced to chemoinformatics by Agrafiotis and co-workers [105]. While they used bagging in conjunction with neural networks, in recent years the decision tree has been most often used as a base classifier. Landrum and co-workers described an application of bagging using a feature-map vector (FMV) as a descriptor [128], which consists of the distances between pharmacophoric features in the molecule and the equivalent features in a 3D pharmacophore model created by the alignment of known actives. A bagged decision tree was used to search for 161 CDK2 actives from a database of 3,464 inactives, and a set of 50 $HT_3$ antagonists from 1,400 inactives. In both cases enrichments were much larger than similarity searching with the FMV descriptor or using 2D similarity. Other notable features of this study are that the effect of choosing a diverse versus a random set of actives for training, and that of using crystallographic structures versus computationally generated conformers for creating the aligned pharmacophore model was evaluated, and found to be fairly minor. Additionally, these impressive results were obtained using only three or four actives. One of the earliest attempts at a more sophisticated machine learning approach to consensus scoring in docking was carried out by Jacobsson and co-workers [61]. Here seven different scoring functions were combined: GOLDscore, DOCKscore, PMFscore, ChemScore and FlexXScore, which make up the popular CScore consensus scoring function [129], plus two scoring functions available with the docking program ICM. Four types of classifier were used: bagged decision trees, bagged rule induction, PLS-DA, and a Bayesian classifier assuming a multivariate normal distribution of the docking scores. All machine learning approaches improved on ICM-Score and CScore over four different targets, but the bagged classifiers were the most effective. No successful regression models were found, so the authors suggested consensus scoring should be used to remove as many inactive molecules as possible, rather than to identify particularly active compounds. The number of actives was only 12 to 43, and approximately two-thirds of the actives were used for training, so the results of this study may be subject to a large amount of statistical uncertainty.

A similar ensemble method is boosting. Like bagging, it involves the majority vote of perturbed base classifiers, but the training set presented to each classifier is determined by the performance of the previously constructed classifier. Instances that were poorly predicted by the previous classifier are given a higher weighting to the subsequent classifier, to allow classifiers to concentrate on those observations that are difficult to predict correctly. To determine the classification of test set molecules, the majority vote of bagging is replaced by a weighted majority vote, where the weight of the vote given to the decision of each base classifier is proportional to the number of correct classifications it made during training. Svetnik and co-workers at Merck evaluated the ability of boosting to screen 10,940 structures screened against CDK2, and 23,102 molecules assessed for their ability to inhibit an unspecified channel protein using atom pair descriptors [79]. In both cases, boosting was superior to a single decision tree, and comparable to the performance of RBF SVMs.

Random forest represents a third variation on ensemble methods, and is designed to be used with decision trees. The resampling method is a bootstrap, as in bagging, without the iterative component of boosting. Unlike bagging and boosting, however, instances and variables are sampled randomly, and only a small subset of the available descriptors are chosen at random to be used by the tree building algorithm when searching for an optimal rule. Svetnik and co-workers have also examined the use of random forest against several datasets, including the same HTS data they used to evaluate boosting [130,131], where it performed competitively. A related method is the Decision Forest [132] of Tong and co-workers, where a different set of descriptors is used to build each tree, and which was recently applied to mining estrogen receptor-binders from a dataset of 57,000 molecules.

Random forest has also been used in conjunction with docking. It is generally accepted that scoring functions are unable to estimate binding affinity consistently [133]. Conversely, the geometric accuracy with which an active molecule is placed in a binding site is considered to be much better. To this end, Teramoto and Fukunishi used a random forest classifier to predict the root mean square deviation (RMSD) of a docked conformation from the bioactive conformation [134]. They used 100 protein-ligand complexes with X-ray coordinates, and for each ligand produced 100 decoys using AutoDock. Descriptors for the random forest classification were generated using 11 scoring functions. The random forest classifiers predicted which poses were within 2.0 Å RMSD of the X-ray coordinates for 90% of cases, while the performance of the individual scoring function varied between 26% and 76%.

Finally, Springer and co-workers described PostDOCK [135], which uses a random forest classifier combined with 32 descriptors combining the ChemScore, DOCK grid score and solvent accessible surface area (SASA) of a docked pose (using DOCK 4.01) to discriminate between actives and inactives. However, it is interesting to note that while the random forest model could achieve an enrichment factor as high as 19.0 at a false positive rate of 5%, a model built using the simpler logistic regression method could achieve an enrichment factor as high as 16.5.

## SEARCH METHODOLOGIES – PARTICLE SWARMS AND ARTIFICIAL ANTS

Search methods [132] are used to explore a multidimensional landscape for an optimal value, where the dimensions are controllable parameters. Common uses of search methods for virtual screening are found in docking, where the goal is to optimize a ligand's atomic coordinates with respect to those of the binding site, and during the training of supervised methods, most often for variable selection. Search methods such as the simplex and steepest descent suffer from being easily caught in local minima. Increasingly, stochastic methods inspired by nature are being used as an alternative. In chemoinformatics, simulated annealing and genetic algorithms are widely used. However, other methods have received increasing attention in recent years.

Particle swarm optimization (PSO) [136] creates a population of solutions, similarly to a genetic algorithm. Each solution is a particle that 'flies' with a particular velocity through the search space. The particle is attracted to search the space near both the best solution that it has seen, and the best solution seen by any particle. The speed and direction of each particle is updated based on these two goals, and the particle then flies to a new location in the search space based on its new velocity. By controlling the weight of the particle's attraction to its personal best solution and that of the entire swarm, a balance between a global and a local search can be carried out. PSO was originally introduced in chemoinformatics as a variable selection technique [137], and this has been their main application (see for example the work of Yu and co-workers [138]). However, the discrete nature of this optimization problem (variables are either in the model or not) does not seem to lend itself well to the continuous nature of PSO, although an extension of PSO to discrete variables has been developed [139].

A more natural application of PSO is in docking, where the coordinates of each particle maps to the coordinates of a ligand. Several groups have now applied PSO to docking, and the Lamarckian genetic algorithm (LGA), a genetic algorithm hybridised with a local search method due to Solis and Wets [140], in AutoDock 3.05 has proven to be a popular benchmark. Janson and Merkle combined PSO with clustering techniques in their ClustMPSO algorithm [141], which uses subswarms and a multi-objective optimization (intermolecular and intramolecular energy in this case) to maintain diversity in the solution space. They demonstrated that ClustMPSO could find lower energies with substantially fewer energy evaluations than LGA, but only two molecules (with 9 and 10 rotatable bonds) were evaluated. Chen and co-workers [142] also combined PSO with the AutoDock scoring function and used the hierarchical fair competition method of Hu and Goodman [143] to prevent premature convergence, calling their technique Tribe-PSO. For 100 ligand-protein complexes taken from the Protein Data Bank (PDB: http://www.rcsb.org/pdb/), the PSO preserved diversity of the solutions and showed superior convergence for very flexible (more than 15 rotatable bonds) ligands, compared to the standard genetic algorithm. Ho and co-workers, with their SODOCK method, also found that PSO (also using the Lamarckian algorithm as a local search technique) could outperform LGA when reproducing the crystal structure of flexible ligands [144]. Additionally, they demonstrated that PSO optimization reproduced crystal structure coordinates with a lower RMSD than DOCK, GOLD and FlexX 19 times out of the 37 structures investigated.

Ant colony optimization (ACO) was introduced by Dorigo and co-workers [145]. In this approach, each descriptor is discretized into several bins. For each descriptor, a weight is associated with each bin. An artificial ant then traces out a path through the solution space, by visiting each descriptor, and choosing one of the values at random, with each bin being chosen with a probability proportional to the weight. The ant leaves a trail of pheromone behind it, which results in the weight of the bins that it visits being increased. This makes it more likely that those bins will be visited by other ants. As with PSO, ant systems were initially used for variable selection in QSAR [146, 147]. However, Korb *et al*. recently described the docking method Protein-Ligand ANT System (PLANTS) [148], which combines an ant system with simplex. Using 33 complexes from the CCDC/ASTEX "clean" list as a validation set [149], PLANTS, when combined with the CHEMPLP scoring function, found more complexes within 2.0 Å of the crystallographic coordinates in a sixth of the time, compared to using GOLD. In a comparison with GOLD for the VS of 43 known factor Xa actives (the same as those used by Jacobsson and co-workers [61]) and 817 inactives (taken from the ZINC database [150]), PLANTS showed a slightly higher enrichment for the top 5% of the ranked database, but was four times faster.

## DATA PREPARATION AND VALIDATION

As machine learning becomes more established as a virtual screening tool, it will become ever more important to ensure that the conclusions that are drawn are well founded. Advances in this area cannot be ensured by novel algorithms alone. There are two important issues: validation of the results, and preparing datasets so that the validation has the chance to succeed. Lessons learnt from the QSAR community remain valid, although good answers to the most pressing problems remain areas of active research. As a larger number of algorithms become available, and computational resources grow, the temptation to apply blindly all available methods to a dataset grows commensurately. Under these conditions, the statistics and validation measures that served the community previously must be modified to account for the much larger risk of a false positive result [151] and we have demonstrated their use in our evaluation of classifiers [152]. Useful pointers for future research may be found in bioinformatics, particularly microarrays, where datasets of similar dimensions to those used in chemoinformatics are encountered, and the false discovery rate is an area of intense interest [153].

For regression, measures related to the root mean square error of prediction are well-established and uncontroversial measures of success. However, for classification, the question of which evaluation measure to use is a subject of debate. Simple classification accuracy has long been known to be inadequate, due to the disparity in sizes between known active and inactive molecules for any given target. For classifiers that predict only the category for a molecule (e.g. active/inactive), then confusion matrices and reporting true/false positive/negative values suffice. However, many classifiers report a weight along with their prediction, e.g. a number between 0 and 1, where 0 would be considered inactive and 1 active. Typically, a threshold of 0.5 is used to convert these predictions into a binary active/inactive classification. However, it is possible to choose a different threshold, and therefore multiple partitions of the data into active and inactive can be generated. In these cases, the receiver operating characteristic curve has been advocated recently [154], which provides a graphical display of the trade off of false positives and false negatives as the threshold is altered. More specifically, the ROC can be plotted as the sensitivity against (1 – specificity).

A further attractive property of ROC curves is that the area under the curve (AUC) can be calculated to provide a convenient single number to summarize the overall behavior [155]. The ROC curve is not without its critics [156], because it summarizes the ability of the algorithm to rank a database over its entirety. It should be borne in mind that most uses of virtual screening are to prioritize a very small number of compounds, and therefore in these cases it is only necessary for the molecules that are predicted to be active with the highest confidence (i.e. the very top of the database after sorting by confidence of activity) to be true positives. Enrichment factors are well known measures that meet this requirement. Alternatives such as the robust initial enhancement have been advocated as improvements [157]. Other alternatives include cost curves [158], and ANOVA [159]. No matter which measure is used, while better than random performance is easily demonstrated in most reports in the literature, Bender and Glen noted that sophisticated descriptors do not always perform substantially better than using simple atom counts [160].

Beyond choosing an appropriate measure to validate the success of screening, it would be even more useful to assess when a prediction is likely to be wrong at the time it is made. Outlier detection remains a problem in QSAR. Recent work on Gaussian processes has demonstrated a more rigorous theoretical framework using a Bayesian approach to estimate uncertainties [161].

Other recent approaches are based on measuring the similarity between a molecule to be predicted and those in the training set, where molecules similar to those in the training set should be more reliably predicted. Jurs and co-workers have described some approaches along these lines based on clustering and ‘reciprocal nearest neighbor’ curves, which can express the sparsity of the space around a molecule as single number [162]. The previous approaches use the same descriptors used to build the learning algorithm to measure similarity; conversely, Sheridan and co-workers advocated using binary fingerprints and standard similarity searching techniques for measuring the similarity of molecules [163]. This raises the question of whether it can be assumed that molecules that appear similar using descriptors in the supervised learning will be similar using fingerprint, but the authors reported promising results. Keefer and Woody used the structure of the Decision Forest classifier to generate a novel measure of similarity [164]. For each decision tree in the forest, the leaf in which each molecule ends up is recorded. Molecules that share many leaves across the forest are considered to be similar to each other. When predicting the class of a new molecule, as well as using the standard majority vote or averaging prediction, a *k*-nearest neighbors prediction can be obtained, where the similarity is based on the number of shared leaves. If this neighbor-based prediction differs from the standard aggregate prediction, then this molecule is flagged as unclassifiable. This method was tested using a dataset of several thousand molecules with measured activity against cytochrome P450 2D6 and 3A4 and error rates in the unclassifiable set were around 10% to 20% higher than the predictions that were kept. Around 20% to 30% of the datasets were found to be unclassifiable by decision forests.

## CONCLUSIONS

The full panoply of modern machine learning has been brought to bear on the problem of virtual screening in recent years. Can we point to one algorithm that outperforms the others? The recent comparative study of Plewczynski and co-workers [165] provides some insight, having examined five targets, each consisting of approximately 10,000 compounds taken from the MDDR, and having compared nearly all the algorithms described in this review. The SVM with a linear kernel was the best performing algorithm, with random forest and *k*NN (in conjunction with a genetic algorithm for variable selection) a close second. ANN performed less well, while decision tree and NBC were less effective still. The fact that ensemble methods can perform close to that of SVM accords well with our own experience on smaller datasets [152], as well as the experiences of Svetnik and co-workers [79, 130, 131], who also found that single decision trees and the NBC were substantially less effective than more modern approaches. Additionally, Glick and co-workers, strong advocates of the use of NBC, have published analyses that show under favorable conditions, an SVM performs better than a decision tree or NBC [67]. However, their same analysis demonstrates that the SVM tolerates noisy data less well, which is likely to be more of an issue in real world screening data, rather than curated databases such as the MDDR.

A final issue to bear in mind is that SVMs have probably the largest number of adjustable parameters of the algorithms under discussion and it is unclear how sensitive the results are to these settings.

Both ligand-based methods, extensions of QSAR and similarity searching, and structure-based methods using docking have been enhanced by machine learning approaches. Indeed, the borders between these disparate approaches are beginning to dissolve, and techniques from the machine learning and data mining communities seem likely to be applied to chemical problems with increasing rapidity. Six years passed between the publication of the first edition of Vapnik's book on SVM [80], and it first being applied to chemical data, and likewise for bagging. However, perhaps the time taken for chemists to notice developments in the machine learning literature is shortening: only three years elapsed between the publication on random forest appearing and its evaluation of relevance for chemoinformatics. Ideally, given the increased interest in machine learning algorithms from those in the chemoinformatics community, the flow of ideas between disciplines will be two-way, and algorithms specifically tailored for the datasets facing virtual screeners (noisy, unbalanced and high dimensional) will be developed. With new algorithms and abundant data come new challenges: given the widespread availability of highly functional statistical and machine learning software, the greatest challenges facing the community are less about the application of the algorithms, and more about ensuring the data are relevant in the first place, and the results are meaningful. With these caveats in mind, we fully expect machine learning to occupy a central place in virtual screening in the coming years.

## ACKNOWLEDGEMENTS

We thank the EPSRC for support (GR/S75765/01) and we are grateful for the use of the University of Nottingham's High Performance Computer.

## REFERENCES

[1] Gasteiger, J., Eds. *Handbook of Chemoinformatics*; Wiley-VCH: Weinheim, **2003**.

[2] Inglese, J.; Auld, D. S.; Jadhav, A.; Johnson, R. L.; Simeonov, A.; Yasgar, A.; Zheng, W.; Austin, C. P. *Proc. Natl. Acad. Sci. USA*, **2006**, *103*, 11473-11478.

[3] Buttingsrud, B.; Ryeng, E.; King, R. D.; Alsberg, B. K. *J. Comput.-Aided Mol. Des*., **2006**, *20*, 361-373.

[4] Srinivasan, A.; Page, D.; Camacho, R.; King, R. *Mach. Learn*., **2006**, *64*, 65-90.

[5] Enot, D. P.; King, R. D. In *Knowledge Discovery in Databases: Pkdd 2003; Proceedings*, **2003**, pp. 156-167.

[6] Marchand-Geneste, N.; Watson, K. A.; Alsberg, B. K.; King, R. D. *J. Med. Chem*., **2002**, *45*, 399-409.

[7] Sternberg, M. J. E.; Muggleton, S. H. *QSAR Comb. Sci*., **2003**, *22*, 527-532.

[8] Mitchell, T. M. *Machine Learning*; McGraw-Hill: Singapore, **1997**.

[9] Hastie, T.; Tibshirani, R.; Friedman, J. *The Elements of Statistical Learning; Data Mining, Inference, and Prediction*; Springer-Verlag: New York, **2001**.

[10] Witten, I. H.; Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*; Morgan Kaufmann: San Francisco, **2005**.

[11] Hansch, C.; Fujita, T. *J. Am. Chem. Soc*., **1964**, *86*, 1616-1630.

[12] Cramer, R. D., III; Patterson, D. E.; Bunce, J. D. *J. Am. Chem. Soc*., **1988**, *110*, 5959-5967.

[13] Wold, S.; Sjostrom, M.; Eriksson, L. *Chemom. Intell. Lab Syst*., **2001**, *58*, 109-130.

[14] Gedeck, P.; Rohde, B.; Bartels, C. *J. Chem. Inf. Model*., **2006**, *46*, 1924-1936.

[15] Evers, A.; Hessler, G.; Matter, H.; Klabunde, T. *J. Med. Chem*., **2005**, *48*, 5448-5465.

[16] Sippl, W. *J. Comput.-Aided Mol. Des*., **2002**, *16*, 825-830.

[17] Ginn, C. M. R.; Turner, D. B.; Willett, P.; Ferguson, A. M.; Heritage, T. W. *J. Chem. Inf. Comput. Sci*., **1997**, *37*, 23-37.

[18] Charifson, P. S.; Corkery, J. J.; Murcko, M. A.; Walters, W. P. *J. Med. Chem*., **1999**, *42*, 5100-5109.

[19] Catana, C.; Stouten, P. F. W. *J. Chem. Inf. Model*., **2007**, *47*, 85-91.

[20] Bro, R. *J. Chemom*., **1996**, *10*, 47-61.

[21] Bro, R.; Smilde, A. K.; de Jong, S. *Chemom. Intell. Lab Syst*., **2001**, *58*, 3-13.

[22] Gohlke, H.; Klebe, G. *J. Med. Chem*., **2002**, *45*, 4153-4170.

[23] Silber, K.; Heidler, P.; Kurz, T.; Klebe, G. *J. Med. Chem*., **2005**, *48*, 3547-3563.

[24] Radestock, S.; Bohm, M.; Gohlke, H. *J. Med. Chem*., **2005**, *48*, 5466-5479.

[25] Matter, H.; Will, D. W.; Nazare, M.; Schreuder, H.; Laux, V.; Wehner, V. *J. Med. Chem*., **2005**, *48*, 3290-3312.

[26] Cherkasov, A.; Ban, F. Q.; Li, Y.; Fallahi, M.; Hammond, G. L. *J. Med. Chem*., **2006**, *49*, 7466-7478.

[27] Friesner, R. A.; Banks, J. L.; Murphy, R. B.; Halgren, T. A.; Klicic, J. J.; Mainz, D. T.; Repasky, M. P.; Knoll, E. H.; Shelley, M.; Perry, J. K.; Shaw, D. E.; Francis, P.; Shenkin, P. S. *J. Med. Chem*., **2004**, *47*, 1739-1749.

[28] Cherkasov, A. *Curr. Comput.-Aided Drug Des*., **2005**, *1*, 21-42.

[29] Jacobsson, M.; Karlen, A. *J. Chem. Inf. Model*., **2006**, *46*, 1334-1343.

[30] Pan, Y. P.; Huang, N.; Cho, S.; MacKerell, A. D. *J. Chem. Inf. Comput. Sci*., **2003**, *43*, 267-272.

[31] Verdonk, M. L.; Berdini, V.; Hartshorn, M. J.; Mooij, W. T. M.; Murray, C. W.; Taylor, R. D.; Watson, P. *J. Chem. Inf. Comput. Sci*., **2004**, *44*, 793-806.

[32] Vigers, G. P. A.; Rizzi, J. P. *J. Med. Chem*., **2004**, *47*, 80-89.

[33] Vieth, M.; Cummins, D. J. *J. Med. Chem*., **2000**, *43*, 3020-3032.

[34] Baber, J. C.; William, A. S.; Gao, Y. H.; Feher, M. *J. Chem. Inf. Model*., **2006**, *46*, 277-288.

[35] Pearlman, R. S.; Smith, K. M. *Persp. Drug Disc. Des*., **1998**, *9-11*, 339-353.

[36] Li, H.; Sutter, J.; Hoffmann, R. In *Pharmacophore Perception, Development and Use in Drug Design*; International University Line: La Jolla, CA, **2000**; pp. 171-190.

[37] Helma, C. *Mol. Div.*, **2006**, *10*, 147-158.

[38] Shen, M.; Beguin, C.; Golbraikh, A.; Stables, J. P.; Kohn, H.; Tropsha, A. *J. Med. Chem*., **2004**, *47*, 2356-2364.

[39] Oloff, S.; Zhang, S. X.; Sukumar, N.; Breneman, C.; Tropsha, A. *J. Chem. Inf. Model*., **2006**, *46*, 844-851.

[40] Zhang, S. X.; Golbraikh, A.; Tropsha, A. *J. Med. Chem*., **2006**, *49*, 2713-2724.

[41] Jensen, B. F.; Vind, C.; Padkjar, S. B.; Brockhoff, P. B.; Refsgaard, H. H. F. *J. Med. Chem*., **2007**, *50*, 501-511.

[42] Cedeño, W.; Agrafiotis, D. K. *J. Comput.-Aided Mol. Des*., **2003**, *17*, 255-263.

[43] Miller, D. W. *J. Chem. Inf. Comput. Sci*., **2001**, *41*, 168-175.

[44] Oloff, S.; Muegge, I. *J. Comput.-Aided Mol. Des*., **2007**, *21*, 87-95.

[45] Kumar, R.; Kulkarni, A.; Jayaraman, V. K.; Kulkarni, B. D. *Internet Electron. J. Mol. Des*., **2004**, 118-133.

[46] Guha, R.; Dutta, D.; Jurs, P. C.; Chen, T. *J. Chem. Inf. Model*., **2006**, *46*, 1836-1847.

[47] Zhang, S. X.; Golbraikh, A.; Oloff, S.; Kohn, H.; Tropsha, A. *J. Chem. Inf. Model*., **2006**, *46*, 1984-1995.

[48] McNeany, T. J.; Hirst, J. D. *J. Chem. Inf. Model*., **2005**, *45*, 768-776.

[49] Hirst, J. D.; McNeany, T. J.; Howe, T.; Whitehead, L. *Bioorg. Med. Chem*., **2002**, *10*, 1037-1041.

[50] Constans, P.; Hirst, J. D. *J. Chem. Inf. Comput. Sci*., **2000**, *40*, 452-459.

[51] Hert, J.; Willett, P.; Wilton, D. J.; Acklin, P.; Azzaoui, K.; Jacoby, E.; Schuffenhauer, A. *J. Chem. Inf. Model*., **2006**, *46*, 462-470.

[52] Wilton, D. J.; Harrison, R. F.; Willett, P.; Delaney, J.; Lawson, K.; Mullier, G. *J. Chem. Inf. Model*., **2006**, *46*, 471-477.

[53] Chen, B. N.; Harrison, R. F.; Pasupa, K.; Willett, P.; Wilton, D. J.; Wood, D. J.; Lewell, X. Q. *J. Chem. Inf. Model.*, **2006**, *46*, 478-486.
[54] Chen, B. N.; Harrison, R. F.; Hert, J.; Mpanhanga, C.; Willett, P.; Wilton, D. J. *Mol. Simul.*, **2005**, *31*, 597-604.
[55] Hert, J.; Willett, P.; Wilton, D. J. *J. Chem. Inf. Comput. Sci.*, **2004**, *44*, 1177-1185.
[56] Wilton, D.; Willett, P.; Lawson, K.; Mullier, G. *J. Chem. Inf. Comput. Sci.*, **2003**, *43*, 469-474.
[57] Harper, G.; Bradshaw, J.; Gittins, J. C.; Green, D. V. S.; Leach, A. R. *J. Chem. Inf. Comput. Sci.*, **2001**, *41*, 1295-1300.
[58] Chen, B.; Harrison, R. F.; Papdatos, G.; Willett, P.; Wood, D. J.; Lewell, X. Q.; Greenidge, P.; Stiefl, N. *J. Comput.-Aided Mol. Des.*, **2007**, *21*, 53-62.
[59] Domingos, P.; Pazzani, M. *Mach. Learn.*, **1997**, *29*, 103-130.
[60] Labute, P. *Pac. Symp. Biocomput.*, **1999**, *4*, 444-455.
[61] Jacobsson, M.; Liden, P.; Stjernschantz, E.; Bostrom, H.; Norinder, U. *J. Med. Chem.*, **2003**, *46*, 5781-5789.
[62] Prathipati, P.; Saxena, A. K. *J. Chem. Inf. Model.*, **2006**, *46*, 39-51.
[63] Glick, M.; Klon, A. E.; Acklin, P.; Davies, J. W. *J. Biomol. Screen.*, **2004**, *9*, 32-36.
[64] Klon, A. E.; Glick, M.; Thoma, M.; Acklin, P.; Davies, J. W. *J. Med. Chem.*, **2004**, *47*, 2743-2749.
[65] Klon, A. E.; Glick, M.; Davies, J. W. *J. Med. Chem.*, **2004**, *47*, 4356-4359.
[66] Klon, A. E.; Glick, M.; Davies, J. W. *J. Chem. Inf. Comput. Sci.*, **2004**, *44*, 2216-2224.
[67] Glick, M.; Jenkins, J. L.; Nettles, J. H.; Hitchings, H.; Davies, J. W. *J. Chem. Inf. Model.*, **2006**, *46*, 193-200.
[68] Nidhi; Glick, M.; Davies, J. W.; Jenkins, J. L. *J. Chem. Inf. Model.*, **2006**, *46*, 1124-1133.
[69] Klon, A. E.; Lowrie, J. F.; Diller, D. J. *J. Chem. Inf. Model.*, **2006**, *46*, 1945-1956.
[70] Quinlan, J. *Mach. Learn.*, **1986**, *1*, 81-106.
[71] Bender, A.; Mussa, H. Y.; Glen, R. C.; Reiling, S. *J. Chem. Inf. Comput. Sci.*, **2004**, *44*, 170-178.
[72] Bender, A.; Mussa, H. Y.; Glen, R. C.; Reiling, S. *J. Chem. Inf. Comput. Sci.*, **2004**, *44*, 1708-1718.
[73] Lang, P. T.; Kuntz, I. D.; Maggiora, G. M.; Bajorath, J. *J. Biomol. Screen.*, **2005**, *10*, 649-652.
[74] Bender, A.; Mussa, H. Y.; Glen, R. C. *J. Biomol. Screen.*, **2005**, *10*, 658-666.
[75] Sanner, M. F.; Olson, A. J.; Spehner, J. C. *Biopolymers*, **1996**, *38*, 305-320.
[76] Goodford, P. J. *J. Med. Chem.*, **1985**, *28*, 849-857.
[77] Yoon, S.; Smellie, A.; Hartsough, D.; Filikov, A. *J. Comput.-Aided Mol. Des.*, **2005**, *19*, 483-497.
[78] Schapira, M.; Abagyan, R.; Totrov, M. *J. Med. Chem.*, **2003**, *46*, 3045-3059.
[79] Svetnik, V.; Wang, T.; Tong, C.; Liaw, A.; Sheridan, R. P.; Song, Q. H. *J. Chem. Inf. Model.*, **2005**, *45*, 786-799.
[80] Vapnik, V. N. *The Nature of Statistical Learning Theory*, Springer, **1999**.
[81] Vapnik, V. N. *Statistical Learning Theory*; Wiley, **1998**.
[82] Burbidge, R.; Trotter, M.; Buxton, B.; Holden, S. *Comput. Chem.*, **2001**, *26*, 5-14.
[83] Czerminski, R.; Yasri, A.; Hartsough, D. *Quant. Struct.-Act. Relat.*, **2001**, *20*, 227-240.
[84] Mercer, J. *Philos. Trans. R Soc. London Series A*, **1909**, *209*, 415-446.
[85] Boser, B. E.; Guyon, I. M.; Vapnik, V. N. In *Proceeedings of the 5th Annual ACM Workshop on Computational Learning Theory*; Haussler D., Eds. ACM Press, Pittsburgh: Pennsylvania, **1992**, pp. 144-152.
[86] Burges, C. J. C. *Data Mining Knowl. Discov.*, **1998**, *2*, 121-167.
[87] Platt, J. *Sequential Minimal Optimization: A fast algorithm for training support vector machines;* Microsoft Research Technical Report **1998**.
[88] Chang, C. C.; Hsu, C. W.; Lin, C. J. *IEEE Trans. Neur. Networks*, **2000**, *11*, 1003-1008.
[89] Fan, R. E.; Chen, P. H.; Lin, C. J. *J. Mach. Learn. Res.*, **2005**, *6*, 1889-1918.
[90] Smola, A. J.; Scholkopf, B. *Stat. Comput.*, **2004**, *14*, 199-222.
[91] Franke, L.; Byvatov, E.; Werz, O.; Steinhilber, D.; Schneider, P.; Schneider, G. *J. Med. Chem.*, **2005**, *48*, 6997-7004.
[92] Lepp, Z.; Kinoshita, T.; Chuman, H. *J. Chem. Inf. Model.*, **2006**, *46*, 158-167.
[93] Jorissen, R. N.; Gilson, M. K. *J. Chem. Inf. Model.*, **2005**, *45*, 549-561.
[94] Saeh, J. C.; Lyne, P. D.; Takasaki, B. K.; Cosgrove, D. A. *J. Chem. Inf. Model.*, **2005**, *45*, 1122-1133.
[95] Byvatov, E.; Sasse, B. C.; Stark, H.; Schneider, G. *ChemBioChem*, **2005**, *6*, 997-999.
[96] Mahe, P.; Ralaivola, L.; Stoven, V.; Vert, J. P. *J. Chem. Inf. Model.*, **2006**, *46*, 2003-2014.
[97] Micheli, A.; Portera, F.; Sperduti, A. *Neurocomputing*, **2005**, *64*, 73-92.
[98] Ralaivola, L.; Swamidass, S. J.; Saigo, H.; Baldi, P. *Neur. Networks*, **2005**, *18*, 1093-1110.
[99] Eitrich, T.; Kless, A.; Druska, C.; Meyer, W.; Grotendorst, J. *J. Chem. Inf. Model.*, **2007**, *47*, 92-103.
[100] Zupan, J.; Gasteiger, J. *Neural Networks in Chemistry and Drug Design*; Wiley-VCH: Weinheim, **1999**.
[101] Rumelhart, D. E.; Durbin, R.; Golden, R.; Chauvin, Y. *Mathematical Perspectives on Neural Networks*; Lawrence Erlbaum Associates: Hillsdale, NJ, **1996**; pp. 533-566.
[102] Betzi, S.; Suhre, K.; Chetrit, B.; Guerlesquin, F.; Morelli, X. *J. Chem. Inf. Model.*, **2006**, *46*, 1704-1712.
[103] Barbault, F.; Zhang, L. R.; Zhang, L. H.; Fan, B. T. *Chemom. Intell. Lab Syst.*, **2006**, *82*, 269-275.
[104] Bazeley, P. S.; Prithivi, S.; Struble, C. A.; Povinelli, R. J.; Sem, D. S. *J. Chem. Inf. Model.*, **2006**, *46*, 2698-2708.
[105] Agrafiotis, D. K.; Cedeno, W.; Lobanov, V. S. *J. Chem. Inf. Comput. Sci.*, **2002**, *42*, 903-911.
[106] Seierstad, M.; Agrafiotis, D. K. *Chem. Biol. Drug Des.*, **2006**, *67*, 284-296.
[107] Kohonen, T. *Biol. Cybernet.*, **1982**, *43*, 56-69.
[108] Selzer, P.; Ertl, P. *QSAR Comb. Sci.*, **2005**, *24*, 270-276.
[109] Pleban, K.; Kaiser, D.; Kopp, S.; Peer, M.; Chiba, P.; Ecker, G. F. *Act. Biochim. Pol.*, **2005**, *52*, 737-740.
[110] Kaiser, D.; Terfloth, L.; Kopp, S.; Schulz, J.; deLaet, R.; Chiba, P.; Ecker, G. F.; Gasteiger, J. *J. Med. Chem.*, **2007**, *50*, 1698-1702.
[111] Schneider, G.; Nettekoven, M. *J. Comb. Chem.*, **2003**, *5*, 233-237.
[112] Klanner, C.; Farrusseng, D.; Baumes, L.; Lengliz, M.; Mirodatos, C.; Schuth, F. *Angew. Chemie Int. Ed.*, **2004**, *43*, 5347-5349.
[113] Omata, K.; Kobayashi, Y.; Yamada, M. *Cat. Commun.*, **2007**, *8*, 1-5.
[114] Thaler, S. L. *J. Alloys Comp.*, **1998**, *279*, 47-59.
[115] DeLisle, R. K.; Dixon, S. L. *J. Chem. Inf. Comput. Sci.*, **2004**, *44*, 862-870.
[116] Kazius, J.; Nijssen, S.; Kok, J.; Back, T.; Ijzerman, A. P. *J. Chem. Inf. Model.*, **2006**, *46*, 597-605.
[117] Karwath, A.; De Radet, L. *J. Chem. Inf. Model.*, **2006**, *46*, 2432-2444.
[118] Wolohan, P. R. N.; Akella, L. B.; Dorfman, R. J.; Nell, P. G.; Mundt, S. M.; Clark, R. D. *J. Chem. Inf. Model.*, **2006**, *46*, 1188-1193.
[119] Jones-Hertzog, D. K.; Mukhopadhyay, P.; Keefer, C. E.; Young, S. S. *J. Pharmacol. Toxicol. Methods*, **1999**, *42*, 207-215.
[120] Rusinko, A.; Young, S. S.; Drewry, D. H.; Gerritz, S. W. *Comb. Chem. High-Throughput Screen.*, **2002**, *5*, 125-133.
[121] Blower, P.; Fligner, M.; Verducci, J.; Bjoraker, J. *J. Chem. Inf. Comput. Sci.*, **2002**, *42*, 393-404.
[122] van Rhee, A. M.; Stocker, J.; Printzenhoff, D.; Creech, C.; Wagoner, P. K.; Spear, K. L. *J. Comb. Chem.*, **2001**, *3*, 267-277.
[123] Yamazaki, K.; Kusunose, N.; Fujita, K.; Sato, H.; Asano, S.; Dan, A.; Kanaoka, M. *Bioorg. Med. Chem. Lett.*, **2006**, *16*, 1371-1379.
[124] van Rhee, A. M. *J. Chem. Inf. Comput. Sci.*, **2003**, *43*, 941-948.
[125] Miller, D. W. *J. Chem. Inf. Comput. Sci.*, **2003**, *43*, 568-578.
[126] Wehrens, R.; Putter, H.; Buydens, L. M. C. *Chemom. Intell. Lab Syst.*, **2000**, *54*, 35-52.
[127] Breiman, L. *Mach. Learn.*, **1996**, *24*, 123-140.
[128] Landrum, G. A.; Penzoff, J. E.; Putta, S. *J. Comput.-Aided Mol. Des.*, **2006**, *20*, 751-762.
[129] Clark, R. D.; Strizhev, A.; Leonard, J. M.; Blake, J. F.; Matthew, J. B. *J. Mol. Graph. Modell.*, **2002**, *20*, 281-295.
[130] Svetnik, V.; Liaw, A.; Tong, C.; Wang, T. *Multiple Classifier Systems; Proceedings* **2004**; pp. 334-343.
[131] Svetnik, V.; Liaw, A.; Tong, C.; Culberson, J. C.; Sheridan, R. P.; Feuston, B. P. *J. Chem. Inf. Comput. Sci.*, **2003**, *43*, 1947-1958.

[132] Tong, W. D.; Hong, H. X.; Fang, H.; Xie, Q.; Perkins, R. *J. Chem. Inf. Comput. Sci.*, **2003**, *43*, 525-531.
[133] Leach, A. R.; Shoichet, B. K.; Peishoff, C. E. *J. Med. Chem.*, **2006**, *49*, 5851-5855.
[134] Teramoto, R.; Fukunishi, H. *J. Chem. Inf. Model.*, **2007**, *47*, 526-534.
[135] Springer, C.; Adalsteinsson, H.; Young, M. M.; Kegelmeyer, P. W.; Roe, D. C. *J. Med. Chem.*, **2005**, *48*, 6821-6831.
[136] Kennedy, J.; Eberhart, R. *IEEE International Conference on Neural Networks 1995*; IEEE Service Center, Perth: Australia, **1995**, pp. 1783-1793.
[137] Agrafiotis, D. K.; Cedeno, W. *J. Med. Chem.*, **2002**, *45*, 1098-1107.
[138] Shen, Q.; Jiang, J.-H.; Jiao, C.-X.; Shen, G.-L.; Yu, R.-Q. *Eur. J. Pharm. Sci.*, **2006**, *22*, 145-152.
[139] Kennedy, J.; Eberhart, R. C. *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*; Piscataway: New Jersey, **1997**, pp. 4104-4109.
[140] Solis, F. J.; Wets, R. J. B. *Math. Oper. Res.*, **1981**, *6*, 19-30.
[141] Janson, S.; Merkle, D. *Hybrid Metaheuristics*, Proceedings; **2005**; pp. 128-141.
[142] Chen, K.; Li, T. H.; Cao, T. C. *Chemom. Intell. Lab Syst.*, **2006**, *82*, 248-259.
[143] Hu, J.; Goodman, E. D. *The Genetic and Evolutionary Computation Conference, GECCO-2002*; Morgan Kauffman, New York: USA, **2002**, pp. 772-779.
[144] Chen, H. M.; Liu, B. F.; Huang, H. L.; Hwang, S. F.; Ho, S. Y. *J. Comput. Chem.*, **2007**, *28*, 612-623.
[145] Dorigo, M.; Maniezzo, V.; Colorni, A. *IEEE Trans. Syst. Man. Cybernet. B*, **1996**, *26*, 29-41.
[146] Izrailev, S.; Agrafiotis, D. K. *SAR QSAR Environ. Res.*, **2002**, *13*, 417-423.
[147] Izrailev, S.; Agrafiotis, D. *J. Chem. Inf. Comput. Sci.*, **2001**, *41*, 176-180.
[148] Korb, O.; Stutzle, T.; Exner, T. E. *Ant Colony Optimization and Swarm Intelligence*; *Proceedings* **2006**; pp. 247-258.
[149] Nissink, J. W. M.; Murray, C.; Hartshorn, M.; Verdonk, M. L.; Cole, J. C.; Taylor, R. *Proteins*, **2002**, *49*, 457-471.
[150] Irwin, J. J.; Shoichet, B. K. *J. Chem. Inf. Model.*, **2005**, *45*, 177-182.
[151] Demsar, J. *J. Mach. Learn. Res.*, **2006**, *7*, 1-30.
[152] Bruce, C. L.; Melville, J. L.; Pickett, S. D.; Hirst, J. D. *J. Chem. Inf. Model.*, **2007**, *47*, 219-227.
[153] Langaas, M.; Lindqvist, B. H.; Ferkingstad, E. *J. Roy. Statist. Soc. Ser. B*, **2005**, *67*, 555-572.
[154] Triballeau, N.; Acher, F.; Brabet, I.; Pin, J. P.; Bertrand, H. O. *J. Med. Chem.*, **2005**, *48*, 2534-2547.
[155] Huang, J.; Ling, C. X. *IEEE Trans. Knowl. Data Eng.*, **2005**, *17*, 299-310.
[156] Truchon, J. F.; Bayly, C. I. *J. Chem. Inf. Model.*, **2007**, *47*, 488-508.
[157] Sheridan, R. P.; Singh, S. B.; Fluder, E. M.; Kearsley, S. K. *J. Chem. Inf. Comput. Sci.*, **2001**, *41*, 1395-1406.
[158] Drummond, C.; Holte, R. C. *Mach. Learn.*, **2006**, *65*, 95-130.
[159] Seifert, M. H. J. *J. Chem. Inf. Model.*, **2006**, *46*, 1456-1465.
[160] Bender, A.; Glen, R. C. *J. Chem. Inf. Model.*, **2005**, *45*, 1369-1375.
[161] Schwaighofer, A.; Schroeter, T.; Mika, S.; Laub, J.; terLaak, A.; Sulzle, D.; Ganzer, U.; Heinrich, N.; Muller, K. R. *J. Chem. Inf. Model.*, **2007**, *47*, 407-424.
[162] Guha, R.; Dutta, D.; Jurs, P. C.; Chen, T. *J. Chem. Inf. Model.*, **2006**, *46*, 1713-1722.
[163] Sheridan, R. P.; Feuston, B. P.; Maiorov, V. N.; Kearsley, S. K. *J. Chem. Inf. Comput. Sci.*, **2004**, *44*, 1912-1928.
[164] Keefer, C. E.; Woody, N. A. *Chemom. Intell. Lab Syst.*, **2006**, *84*, 40-45.
[165] Plewczynski, D.; Spieser, S. A. H.; Koch, U. *J. Chem. Inf. Model.*, **2006**, *46*, 1098-1106.

**Received: May 3, 2007** **Revised: August 31, 2007** **Accepted: September 4, 2007**