Comparisons of Classification Methods for Screening Potential Compounds

Aijun An
Department of Computer Science
York University
Toronto, Ontario M3J 1P3 Canada
aan@cs.yorku.ca

Yuanyuan Wang
Department of Statistics and Actuarial Science
University of Waterloo
Waterloo, Ontario N2L 3G1 Canada
y32wang@uwaterloo.ca

Abstract

We compare a number of data mining and statistical methods on the drug design problem of modeling molecular structure-activity relationships. The relationships can be used to identify active compounds based on their chemical structures from a large inventory of chemical compounds. The data set of this application has a highly skewed class distribution, in which only 2% of the compounds are considered active. We apply a number of classification methods to this extremely imbalanced data set and propose to use different performance measures to evaluate these methods. We report our findings on the characteristics of the performance measures, the effect of using pruning techniques in this application and a comparison of local learning methods with global techniques. We also investigate whether reducing the imbalance in the training data by up-sampling or down-sampling would improve the predictive performance.

1 Introduction

High throughput screening (HTS) is a technique in which predicting extreme values is more important than predicting low or mid-ranged values [15]. HTS has been used in drug discovery to screen large numbers of potential compounds against a biological target. Biotechnology advances, such as newly developed synthetic methods and better assay techniques, make it possible to screen tens of thousands to hundreds of thousands of compounds at the early stage of drug design. For example, in the application this paper presents, nearly 30,000 compounds have been assayed to discover their biological activities for protecting human cells from HIV infection. However, it is impractical to test every available compound against every biological target. Pharmaceutical companies now have of order one million compounds in their databases, and combinatorial chemistry can generate similar numbers of new compounds. Therefore, there is a great need to optimize this high throughput screening process by developing methods that can identify promising compounds from a large chemical inventory on the basis of a relatively smaller set of tested compounds. One approach is to use the data from tested compounds to relate biological activity to molecular descriptors of chemical structures. Discovering this structure-activity relationship helps biologists and chemists make decisions on which compounds are most likely to be highly active, so that they can speed up the searching process [17].

Many techniques can be used to discover the structure-activity relationship based on a set of tested compounds. For example, Jones-Hertzog *et al* [8] applied the recursive partitioning technique for building decision trees to model the structure-activity relationship. King *et al* [9] applied an inductive logic programming program to discover the structure-activity relationship. Neural networks have also been applied [3]. A major challenge in modeling this structure-activity relationship is that, although the data set may contain a large number of tested compounds, active compounds are often rare. For example, in the data set that we are working on for discovering a compound's activity in protecting human cells from HIV infection, only 2% of the compounds are active. Therefore, we are facing with a problem of learning from an extremely imbalanced data set. Learning with this kind of imbalanced data set presents problems to learning systems, problems which are not revealed when the systems work on relatively balanced data sets. Since most inductive learning algorithms assume that maximizing accuracy on a full range of cases is the goal [13], these systems exhibit accurate prediction for the majority class cases, but very poor performance for cases associated with the low frequency class. A solution to this problem is to reduce the imbalance in the data set by using different sampling techniques, such as data reduction or "down-sampling" techniques that remove only majority class examples [10] and "up-sampling" techniques that duplicate the training examples of the minority class or create new examples by corrupting existing ones with artificial noise [6]. An alternative to balancing the classes is to develop a learning algorithm that is intrinsically insensitive to

0-7695-1119-8/01 $17.00 © 2001 IEEE

class distribution in the training set [12]. An example of this kind of algorithm is the SHRINK algorithm [11] that finds only rules that best summarizes the positive examples (of the small class), but makes use of the information from the negative examples.

In this paper, we investigate several existing data mining and statistical methods for modeling the structure-activity relationship based on an extremely imbalanced data set. The methods include a decision tree learning method, a rule induction method, a neural network method, a k-nearest neighbor method and a few regression methods. Our objective is as follows. First, we would like to determine how each of these methods reacts to the extremely imbalanced class distribution and which of these methods is most appropriate for this kind of learning problem. Second, we would like to evaluate these methods using different performance measures and determine whether there is correlation between the performance measures. Third, we would like to investigate whether pruning techniques that are often used in the decision tree and decision rule learning to avoid overfitting would help improve the predictive performance on imbalanced data sets. Finally, we would like to investigate whether reducing the imbalance in the training data by up-sampling or down-sampling would improve performance.

2 The Data Set

The study was performed with a data set of nearly 30,000 compounds obtained from GlaxoSmithKline (GSK). The data were collected by the National Cancer Institute (NCI) in an effort to discover new compounds capable of inhibiting the HIV virus. The response variable indicates the activity status of the compounds as confirmed by the Developmental Therapeutics Program (DTP) AIDS anti-viral screen which measures how a compound protects human CEM cells from HIV-1 infection. The activity measure for each compound has three levels: 0 (inactive), 1 (moderately active), and 2 (active). The six descriptor variables were generated by GlaxoSmithKline chemists and are continuous variables called BCUT numbers which describe the structure of the compounds such as their surface area, bounding patterns, charges, and hydrogen bond donor and acceptor ability. They were calculated based on the work by Burden [4], who discovered that the compounds with similar structures have similar BCUT values.

The data set is very unbalanced. There are 215 active compounds, 393 moderately active compounds and the rest (29,204) are inactive. Our initial analysis of the data indicated that the inactives have a very complex distribution in the space of each pair of the descriptors and the active compounds are located in regions where there are many inactive ones. Because some of the methods we evaluated were designed to handle a binary response and there are relatively few compounds in the two active categories, we combined the active and moderately active compounds into one group. As a result, the data set contains 608 active compounds, which are about 2% of the compounds in the data set, and the rest are inactive compounds. To evaluate the modeling methods, the data set is further randomly split into a training and a test set. The training and test sets are of equal size and are both comprised of 304 active compounds and 14602 inactive compounds.

3 Performance Measures

Our objective is to evaluate some data mining and statistical classification methods on the data set. For each method, we build a classification model based on the training set and then test the model on the test set. A simple way to test the model is to classify the compounds in the test set and collect the classification accuracy as the performance measure. However, since it is the active compounds that are of interest and the active compounds occupy a very small percentage of the data, accuracy on the entire test set is not an appropriate performance measure for this application. Furthermore, simply classifying compounds is not sufficient. The domain experts would like the compounds in the test set to be presented to them in decreasing order of a prediction score with the highest prediction indicating the most probably active compound so that the compounds that are most likely to be active can be assayed in their labs first. Due to this reason, for each method being evaluated, we have to find a way to assign a prediction score to a test case and rank the cases according their scores. To be cost effective, it is preferred that a high proportion of the compounds ranked highest are actually active.

To evaluate the predictive performance of this kind, a popular measure is the *hit rate*, which is the proportion of active compounds or "hit" amongst those selected [15]. In this paper, we use three performance measures to evaluate the methods. First, we propose to use *average hit rate*, which is defined as the average of the hit rates at the points where active compounds are correctly recognized in a ranked list. The average hit rate has the similar definition to the performance measure of *average precision* used in information retrieval[1]. The second measure we use is called *hit curves*, which depicts the number of active compounds versus the number of compounds selected from a ranked list. With a hit curve we would like to show the performance on the highest region of a ranked list. Therefore, we restrict the number of selected compounds to be no more than 500. The third performance measure we use is *ROC*

[1] Finding active compounds from a large collection of compounds is similar to finding relevant documents from a large collection of documents in information retrieval. Both tasks have a highly skewed class distribution and output a ranked list of objects.

curves [14], which depict how the percentage of correctly recognized active compounds depends on the percentage of the incorrectly classified inactive compounds. ROC curves illustrate tradeoffs between true positive rates and false positive rates with respect to the active compounds. Given a ranked list, a ROC curve shows the predictive performance on the whole region of the list.

4 Description of Classification Models

We evaluate the following data mining and statistical methods for discovering classification models from data. Since it is necessary to output a ranked list of test examples in the prediction phase, we also describe the ranking criterion that we used for each method.

4.1 ELEM2

ELEM2 [1] is a rule induction algorithm that generates a set of classification rules by selecting attribute-value pairs from data. The learning strategy used in ELEM2 is a sequential covering method, which sequentially learns a single conjunctive rule, removes the examples covered by the rule, then iterates the process until all positive examples of a class are covered or until no rule can be generated. The result of the learning process for one class is a disjunctive set of conjunctive rules. ELEM2 also has an option to use a post-pruning technique that removes some attribute-value pairs from the condition part of a rule to avoid over-fitting. To post-prune a rule, ELEM2 computes a rule quality value for the rule according to one of the rule quality formulas described in [2]. In post-pruning, ELEM2 checks each attribute-value pair in the rule in the reverse order in which they were selected to determine if removal of the attribute-value pair will decrease the rule quality value. If not, the attribute-value pair is removed and the procedure checks all the other pairs in the same order again using the new rule quality value resulting from the removal of that attribute-value pair to discover whether another attribute-value pair can be removed. This procedure continues until no pair can be removed.

The rules generated by ELEM2 can be used to classify new examples in a test set. To apply ELEM2 to identify active compounds in our particular application, we design a ranking procedure that calculates a numerical score for each test example and ranks the test examples according to both their predicted classes and their scores. The score is called *ranking score* and measures an example's likelihood of belonging to a class, e.g., the category of active compounds in our particular application. To define the ranking score of an example e with respect to a class C, we first compute a matching score between e and a rule r of C using $MS_C(e, r) = \frac{m}{n} \times Q(r)$, where n is the number of attribute-value pairs that r contains, m is the number of attribute-value pairs in r that are matched with e, and $Q(r)$ is the rule quality value of r. The ranking score of e with respect to C is defined as

$$RS(e, C) = \sum_{i=1}^{k_1} MS_C(e, r_i) - \sum_{j=1}^{k_2} MS_{\neg C}(e, r_j),$$

where r_i is a rule of C, k_1 is the number of rules of C, r_j is a rule of classes other than C, and k_2 is the number of this kind of rules.

The ranking algorithm of ELEM2 ranks the test examples according to both the predicted class label (produced by ELEM2's classification program) for the example and the ranking score of that example with respect to a specified class C, e.g., the active compounds. It places test examples that are classified into the specified class C in front of other test examples and ranks the examples in each group in decreasing order of their ranking scores with respect to C.

4.2 Classification Trees

We use a Classification and Regression Tree (CART) program implemented in S [5] to test the performance of decision trees on our data set. Using binary recursive partition, a decision tree method successively splits the data along coordinate axes of predictors. At each division, the resulting two subsets of data are as homogeneous as possible with respect to the response of interest. In S, the splitting criterion is *deviance*, which measures the homogeneity of the two subsets. Let p_{ij} denote the probability of one observation in subset i to be in class j and n_{ij} denote the number of observations that are in subset i and belong to class j. The deviance of subset i is defined as $D_i = -2 \sum_j n_{ij} \log p_{ij}$. The deviance of the subtree generated by the split is defined as $D = \sum_i D_i$. The split that minimizes D is chosen. The default setting in S uses the following two constraints to stop further splitting the data:

(1) there must be at least 10 observations in a node; and

(2) the node deviance must be at least 1% of the root node deviance.

We call the trees generated with this default setting *default trees*. A *pure tree* can be constructed by removing these two constraints to allow the tree to perfectly adapt to the training data. In our experiments, we evaluate both kinds of trees on our data set.

To rank the examples in the test set, we calculate a score for each terminal node of a tree. A test example is assigned the score of the terminal node it falls in. Examples in the test set are ranked according to their assigned scores. To calculate the score for a terminal node in a default tree, we first

calculate the estimated hit rate for the node as $\hat{p} = \frac{n_{active}}{n}$, where n_{active} is the number of active compounds in the training set that fall in the node and n is the number of compounds falling in the node. To account for uncertainty in $\hat{p}$, we assume a Binomial model for responses in each terminal node and calculate a 95% one-sided confidence interval or lower bound, $\hat{p_{lb}}$, for the true hit rate p. The $\hat{p_{lb}}$ score is large if $\hat{p}$ is large and there are many compounds in a node. We use the $\hat{p_{lb}}$ score to rank terminal nodes and test examples for a default tree. For a pure tree, we rank terminal nodes and test examples according to the node size, which is the number of active compounds in the training set that fall in the node.

4.3 k-Nearest Neighbor

K-nearest neighbor (kNN) classification is a very simple but powerful algorithm. For each case in the test set, a kNN method finds k nearest points in the training set according to a distance measure and assigns a predicted class to the test case by using a (weighted) vote among the k selected neighbors. In our experiment, k is determined by using leave-one cross-validation on the training data and Euclidean distance is used as the distance measure to find k nearest neighbors. Votes for the active compounds from the k nearest neighbors relative to all the votes can be regarded as the probability of the test case to be in the class of active compounds. This probability is used to rank test cases.

4.4 Logistic Regression Model

Logistic Regression Model (LRM) is a special case of generalized linear models and it is a popular model to handle data with binary response. LRM assumes

$$\log\left(\frac{p}{1-p}\right) = b_0 + b_1 x_1 + b_2 x_2 + ... + b_n x_n, \quad (1)$$

where p denotes the probability that the observed case (represented by $x_1, x_2, ..., x_n$) is in the category of interest and b's are the parameters to be estimated from the training data. In the prediction phase, the probability, p, that a test case is in the category of interest can be calculated with the estimated b's. Test cases are then ranked according to their probabilities of being in this category, which is the category of active compounds in our application.

4.5 Generalized Additive Model

The Generalized Additive Model (GAM) used in our study is an extension to the logistic regression model. In GAM, each predictor x_i in (1) is replaced by a smooth function, $f_i(x_i)$, where $i = 1, ..., n$, as follows:

$$\log\left(\frac{p}{1-p}\right) = b_0 + f_1(x_1) + f_2(x_2) + ... + f_n(x_n), \quad (2)$$

where $f_1, f_2, ...$ and f_n are smoothing splines which can be estimated from the training set.

4.6 Neural Networks

A feed-forward neural network [16] with one hidden layer of 9 nodes is used in our study. It is the simplest but the most common form of neural networks. The neural network can output an estimated probability that a test case is in a category of interest. We use this probability to rank test cases.

4.7 MARS

In Multivariate Adaptive Regression Splines (MARS), a multiple regression function is approximated using linear splines and their tensor products. Detailed description of MARS can be found in [7].

5 Empirical Comparison of Modeling Methods

To evaluate the above data modeling methods, we randomly split our data set four times, resulting in four training-test data splits. The training and test sets in each split are of equal size and both consist of 304 active and 14602 inactive compounds.

5.1 Results

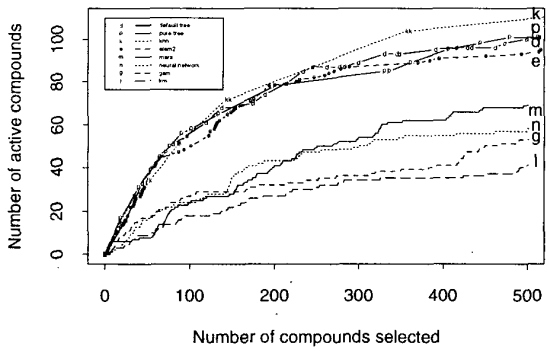

Figure 1. Hit Curves from Different Methods

Figure 1 compares the methods using hit curves for one random training-test data split[2]. From these hit curves, we

[2] The curves for three other splits are similar

can observe that *default tree, pure tree*, ELEM2[3] and the k-nearest neighbor method have the lead, followed by MARS and the neural network method. The worst performance is given by GAM and LRM.

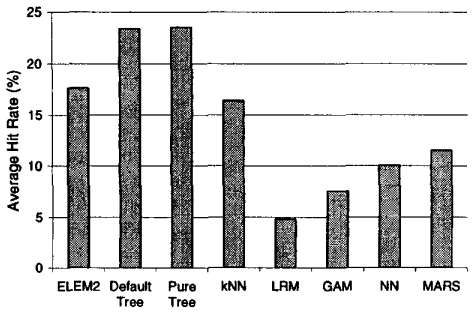

Figure 2. Average Hit Rates for the 8 Methods

Figure 2 compares the methods in terms of average hit rates. The average hit rate for each method is obtained by averaging the average hit rates on 4 random training-test data splits. It can be easily observed from the figure that the two tree methods take the lead, followed by ELEM2 and kNN, which in turn followed by MARS and the neural network method. LRM has the worst performance.

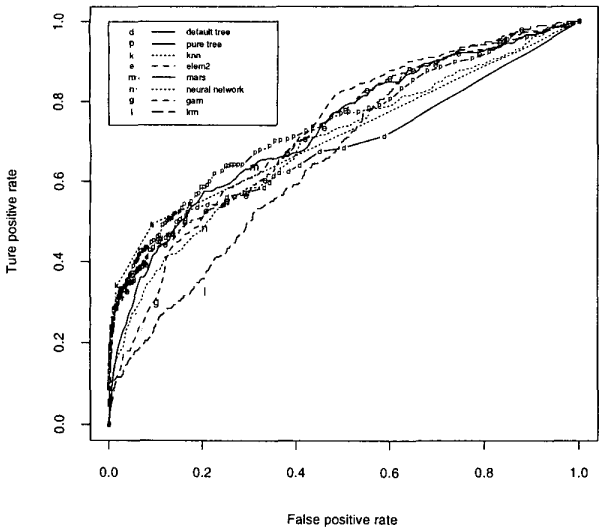

Figure 3. ROC Curves from Different Methods

Figure 3 presents the ROC curves produced by the 8 methods on one random training-test data split[4]. ROC curves illustrate the performance on the entire region of a ranked list. The 8 curves cross each other and there is no single method dominating all the time. However, we can generally say that at the very beginning, pure tree, default tree, ELEM2 and kNN are among the best and their curves overlap, then the kNN method outperforms others and later the pure tree and GAM pick up the lead. MARS is also a good method in this comparison even if its top ranking performance displayed in hit curves is not as good. The LRM method remains the lowest in ROC curves, which is consistent with its ranking in terms of top ranking performance. Another observation we obtain from these ROC curves is that default trees are not as good as pure trees, which conflicts with observations from many other applications that smaller trees usually give better performance.

5.2 Discussion

It is obvious that hit curves depict the top ranking performance, while ROC curve illustrate the performance on the entire region of a ranked list. However, it was not obvious at what region of a ranked list the average hit rate is good at measuring the performance. By comparing our results on average hit rates with the results on hit curves, we observe that the average hit rate actually measures the top ranking performance. For our data set, we can roughly say that it measures the performance for the top 200 compounds in a ranked list even if the definition of average hit rate is based on the entire region. This observation can be explained by analyzing the definition of average hit rate. An active compound that ranks higher makes larger contribution to the value of the average hit rate. A low ranking active compound has a very small weight in the computation of the average hit rate.

In terms of the modeling methods we compare, we observe that local methods, including the classification tree methods, ELEM2 and kNN, which are good at capturing the local behavior and interactions in the data, are more successful techniques, outperforming all the other methods. These local methods are more general, flexible and make minimal assumptions about the underlying relationships. They are able to focus on very local regions with concentrations of active compounds. In kNN, strong local behavior in this data set is also indicated by the fact that an optimal k is chosen by using cross-validation on the training data.

Among the other methods we evaluate, LRM is a popular model for problems with binary response variables. However, it assumes that the log odd of the probability of active response, instead of inactive response, can be well approximated by a linear combination of the BCUT numbers. This assumption is unrealistic for the data that have only six predictors and may have a lot of large noises. GAM incorporates adaptive smoothing (smoothing splines) on each pre-

[3] ELEM2 has an option for choosing a rule quality formula. The result presented here is from using the $C2$ rule quality formula [2].

[4] The results on three other random splits are similar.

dictor into the data model, which results in a better performance than LRM. MARS allows not only non-linear effects within predictors but also some interactions among them. Feed-forward neural networks can also be seen as a method to parameterize a fairly general non-linear function [16]. We have observed from our experiments that both MARS and feed-forward neural networks give better performance than LRM and GAM on our data set.

6 Pruning vs No-pruning

We noticed from the above results that the pure tree method is comparable to the default tree method in terms of average hit rates and hit curves. In terms of ROC curves, it is obvious that the pure tree outperforms the default tree. Therefore, we can say that the pure tree performs as well as or better than the default tree in terms of top ranking performance and it is better than the default tree in terms of the performance in the entire region of the ranked list. This finding was a surprise to us because a pure tree perfectly adapts to the training data and it is usually considered to be overfitting the data. In many other applications, smaller trees, rather than overfitting trees, are preferable. A possible reason for our finding is that conventional criteria used in learning algorithms, such as misclassification rates, accuracy measurements or deviance, often assume that target classes have a balanced distribution.

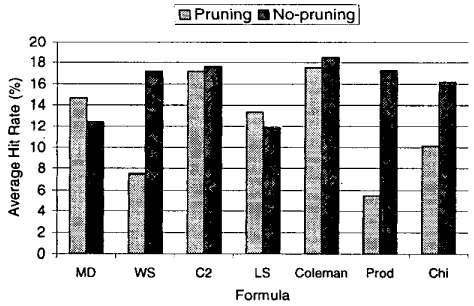

Figure 4. Comparison of Pruning with No-pruning

To further test the results, we conducted experiments with ELEM2 using different rule quality formulas. A rule quality formula is used in the post-pruning process of ELEM2 to determine whether an attribute-value pair in a rule should be removed to avoid overfitting. Description of the rule quality formulas used in ELEM2 can be found in [2]. Figure 4 shows the average hit rates of ELEM2 using different rule quality formulas with or without the pruning technique[5]. The results indicate that for most of the rule quality formulas, such as *WS, Prod, Chi, C2* and *Coleman*, no-pruning produces better average hit rates than pruning. Only for formulas *MD* and *LS*, pruning is beneficial. However, these two formulas, even with the pruning option, are not as competitive as some other formulas on this imbalanced data set. The best performances are given by formulas *Coleman* and *C2*, indicating that *Coleman* and *C2* better handle the imbalance in the data set and that the pruning technique is not beneficial when the data is extremely imbalanced.

7 Balancing the Data

Another objective of this research is to discover whether reducing the imbalance in the training data would improve the predictive performance for the 8 modeling methods we have evaluated. To reduce the imbalance in the training data, we conducted both up-sampling and down-sampling. For up-sampling, we created 6 additional training sets by duplicating the examples of active compounds to increase the prevalence of active compounds in the training data. Percentages of active compounds in these 6 training sets are 4%, 8%, 14%, 25%, 40% and 50%, respectively. The original distribution of active compounds is 2%. Figure 5 illustrates the average hit rates versus the percentage of active compounds in the training data for the 8 modeling methods. From the curves we can observe that only for the pure tree method, balancing the data by up-sampling increases the performance. However, the increases are small and are not considered as significant. For other methods, balancing the data is not beneficial. The curves for the LRM and GAM methods are flat, indicating that they are not sensitive to the changes in the distribution of the data. MARS and the feed-forward neural network method are sensitive to the changes. The performance for these two methods can be increased by up-sampling. But their performance can also be decreased if the "right" distribution is not chosen. For the kNN method, the performance is decreasing at the beginning and is later flattened as the percentage of the active compounds increases. This is because in kNN k is chosen by using cross-validation and as the active compounds are being duplicated many times, k is consistently determined to be 1 because too many duplicated active compounds are in the training data.

We also conducted experiments with down-sampling for reducing the imbalance in the data. For down-sampling, we keep all the active compounds in the training data and randomly select a subset of inactive compounds. We created 13 down-sampled training sets. Each of the training sets contains $n \times 304$ inactive compounds, where n changes

[5] The results in the figure are the average over 4 random training-test data splits.

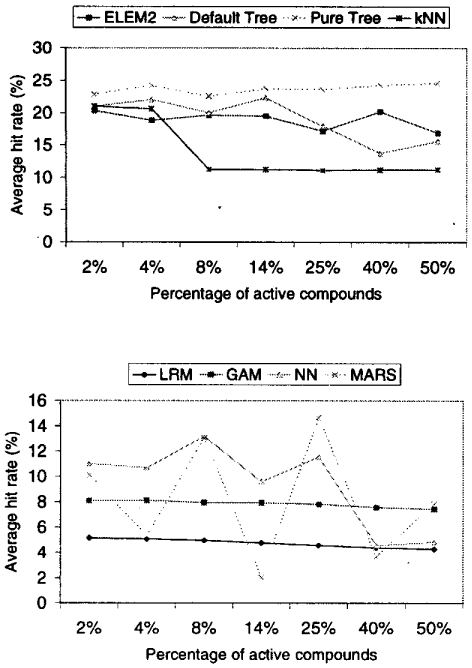

Figure 5. Effects of Up-sampling on Average Hit Rates

from $1, 3, 5, 7, ...$, until 40. Consequently, the percentage of active compounds in the training set changes from $50\%, 25\%, 16.7\%, 12.5\%, ...$, until 2.4%. In the original training data, the percentage of active compounds is 2.04%. Our experimental results for down-sampling is shown in Figure 6. The horizontal axis represents the percentage of active compounds in the down-sampled training data. The left-most point represents the true distribution of the data. As the percentage of active compounds increases, the size of training data decreases because of more inactive compounds are removed due to down-sampling. The curves describe the changes of average hit rates with respect to the changes in the percentage of the active compounds for the 8 modeling methods. We can observe that all the methods reach their lowest average hit rates when the data become perfectly balanced (50%). This is because too much information is lost by reducing the number of inactive compounds to be the same as active compounds. It can also be observed that LRM and GAM are not sensitive to the changes in training data since their curves are pretty much flat. The performance for the kNN method consistently decreases as the more and more inactive compounds are removed from the training data. For ELEM2, none of the down-sampled training sets improves the performance. For each of the tree methods, small improvement can be seen

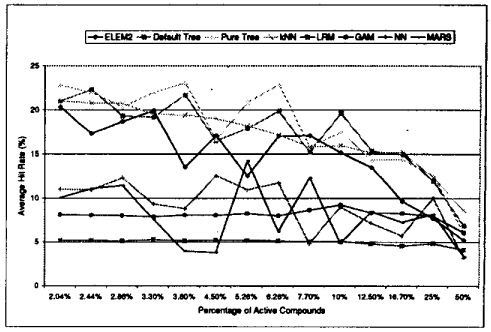

Figure 6. Effects of Down-sampling on Average Hit Rates

at two points. But, generally, their performance decreases. For MARS and the neural network method, down-sampling can help, but it can also decrease the performance compared to using the original training set.

8 Conclusions

We have compared a number of classification methods on an extremely imbalanced data set for modeling the structure-activity relationship to identify active compounds for drug discovery. Among the methods we evaluated, local methods, such as the tree methods, the ELEM2 rule induction method and the kNN method, which are able to identify local behaviors and interactions, outperform other methods in terms of measures for the top ranking performance. Among the local methods, the pure tree method is robust in that it performs well on all the four data splits and that it does not only outperform others in the top region, but is also among the best methods evaluated on the entire region of the ranked list.

The pure tree's outstanding performance over the default tree indicates that the pre-pruning technique used in the default tree learning to avoid overfitting the data is not beneficial. This result is consistent with the results obtained from ELEM2 that uses post-pruning techniques to prevent the rules from overfitting the data. We found that for most of the rule quality formulas tested, especially those that give the best performance in the imbalanced data set, post-pruning does not improve the predictive performance. This conclusion needs to be further tested on other imbalanced data sets to determine whether the result is unique for our application.

We also found in this research that reducing the imbalance in the training data by either up-sampling or down-sampling does not increase the predictive performance for most of the evaluated methods. Only for MARS and the

neural network method, the performance can be improved at certain points of up-sampling or down-sampling. At other points, the performance can be decreased. Therefore, careful selection of up-sampling or down-sampling points in terms of percentage of active compounds in the training data is necessary for using the up-sampling or down-sampling technique with MARS and neural networks. Cross-validation may be used in this selection.

In terms of performance measures used in our evaluation, ROC curves are good at depicting the performance in the entire region of a ranked list. Both hit curves and average hit rates are good at measuring the top ranking performance, which is considered to be important for our application. A benefit of using average hit rates is that it is easy to rank the methods being compared, while hit curves may cross each other multiple times, which makes it hard to determine which method is actually better overall. The measure of average hit rate can also be easily incorporated into a learning algorithm as a criterion for doing some sort of selection. For example, we may use the average hit rate as a criterion in cross-validation for determining an optimal tree size in the tree post-pruning process. This is one of the items we will work on in the future. We expect that a "right" criterion in pruning can lead to better performance.

Acknowledgment

This research is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). We would also like to thank Nick Cercone, Hugh Chipman and William Welch for their suggestions on this work.

References

[1] An, A. and Cercone, N. 1998. "ELEM2: A Learning System for More Accurate Classifications." *Proceeding of the 12th Canadian Conference on Artificial Intelligence*, Vancouver, Canada.

[2] An, A. and Cercone, N. 2000. "Rule Quality Measures Improve the Accuracy of Rule Induction: An Experimental Approach", *Proceedings of the 12th International Symposium on Methodologies for Intelligent Systems*, Charlotte, NC. 119-129.

[3] Andrea, T.A. and Kalayeh, H. 1991. Application of Neural Networks in Quantitative Structure-Activity Relationships of Dihydrofolate Reductase Inhibitors. *J. Med. Chem.*, Vol.34, 2824-2836.

[4] Burden, F.R. 1989. "Molecular Identification Number for Substructure Searches". *Journal of Chemical Information and Computer Sciences*, Vol.29, 225-227.

[5] Clark, L.A., Pregibon, D. 1991 "Tree-Based Models" in *Statistical Models in S*, edited by J. M. Chambers and T. J. Hastie. Wadsworth & Brooks/cole Advanced Books & Software Pacific Grove, California.

[6] DeRouin, E., Brown, J., Beck, H., Fausett, L. and Schneider, M. 1991. "Neural Network Training on Unequally Represented Classes", In Dagli, C.H., Kumara, S.R.T. and Shin, Y.C. (eds.), *Intelligent Engineering Systems Through Artificial Neural Networks*, ASME Press. 135-145.

[7] Friedman, J.H. 1991. "Multivariate Adaptive Regression Splines", *The Annals of Statistics*, 19, 1-141.

[8] Jones-Hertzog, D.K., Mukhopadyay, P., Keefer, C.E., Young, S.S. 2000. "Use of Recursive Partitioning in the Sequential Screening of G-protein-coupled Receptors". *Journal of Pharmacological and Toxicological Methods*, Vol.42. No.1999, 207-215.

[9] King, R.D., Muggleton, S., Lewis, R.A. and Sternberg, M.J.E. 1992. "Drug Design by machine learning: The use of inductive logic programming to model the structure-activity relationships of trimethoprim analogus binding to dihydrofolate reductase". *Proc. Natl. Acad. Sci.*, Vol.89, No.23.

[10] Kubat, M. and Matwin, S. 1997. "Addressing the Curse of Imbalanced Training Sets: One-Sided Sampling". *Proceedings of the Fourteenth International Conference on Machine Learning*, Morgan Kaufmann. 179-186.

[11] Kubat, M., Holte, R. and Matwin, S. 1997. "Learning when Negative Examples Abound," *Proceedings of ECML-97*, Springer. 146-153.

[12] Kubat, M., Holte, R. and Matwin, S. 1998. "Machine Learning for the Detection of Oil Spills in Satellite Radar Images", *Machine Learning*, 30, pp.195-215.

[13] Provost, F. 2000 "Machine Learning from Imbalanced Data Sets", *Invited paper for the AAAI'2000 Workshop on Imbalanced Data Sets*,

[14] Provost, F. and Fawcett, T. 2001. "Robust Classification for Imprecise Environments", *Machine Learning*, 42, 203-231.

[15] Tatsuoka, K., Gu, C., Sacks, J. and Young, S.S. 1998. Predicting Extreme Values in Large Datasets. *Journal of Computational and Graphical Statistics.*

[16] Venables, W.N., and Ripley, B.D. (1999).*Modern Applied Statistics with S-plus*, 3rd edition. New York : Springer, c1999.

[17] Wang, Y., Chipman, H.A., Welch W.J. (2001) "Mining Nuggets from High Throughput Screening Data" a paper presented at the annual meeting of the Statistical Society of Canada (SSC) in 2001.