BIOINFORMATICS

Vol. 19 no. 6 2003, pages 764–771
DOI: 10.1093/bioinformatics/btg054

Feature selection and transduction for prediction of molecular bioactivity for drug design

Jason Weston[1, 2,*], Fernando Pérez-Cruz[2], Olivier Bousquet[1, 2], Olivier Chapelle[1, 2], André Elisseeff[1, 2] and Bernhard Schölkopf[1, 2]

[1]Max Planck Institute, Spemannstr. 38, Tuebingen, 72076, Germany and [2]BIOwulf Technologies, 305 Broadway, New York, NY 10007, USA

Received on April 16, 2002; revised on October 31, 2002; accepted on November 10, 2002

ABSTRACT

Motivation: In drug discovery a key task is to identify characteristics that separate active (binding) compounds from inactive (non-binding) ones. An automated prediction system can help reduce resources necessary to carry out this task.

Results: Two methods for prediction of molecular bioactivity for drug design are introduced and shown to perform well in a data set previously studied as part of the KDD (Knowledge Discovery and Data Mining) Cup 2001. The data is characterized by very few positive examples, a very large number of features (describing three-dimensional properties of the molecules) and rather different distributions between training and test data. Two techniques are introduced specifically to tackle these problems: a feature selection method for unbalanced data and a classifier which adapts to the distribution of the the unlabeled test data (a so-called transductive method). We show both techniques improve identification performance and in conjunction provide an improvement over using only one of the techniques. Our results suggest the importance of taking into account the characteristics in this data which may also be relevant in other problems of a similar type.

Availability: Matlab source code is available at http://www.kyb.tuebingen.mpg.de/bs/people/weston/kdd/kdd.html

Contact: jason.weston@tuebingen.mpg.de

Supplementary information: Supplementary material is available at http://www.kyb.tuebingen.mpg.de/bs/people/weston/kdd/kdd.html.

INTRODUCTION

The first step in the discovery of a new drug is usually to identify and isolate the receptor to which it should bind, followed by testing many small molecules for their ability to bind to the target site. This leaves researchers with the task of determining what separates the active (binding) compounds from the inactive (non-binding) ones. Such a determination can then be used in the design of new compounds that not only bind, but also possess certain other properties required for a drug. See KDD (2001) for an overview. The task of determination can be seen in a machine learning context as one of feature selection (choosing relevant features to describe the unknown dependency between characteristics and labeling of the data). The present task is a particularly challenging one for at least three reasons:

1. *Few positive examples*: Only a few of the tested compounds may bind to the target site. Thus little information is given indicating positive correlations between features and the labels.
2. *Large number of features*: The relevant features have to be selected from a huge collection of *potentially* useful features. This makes it likely that at least some of the features that are in reality uncorrelated with the labels appear to be correlated due to noise.
3. *Different distributions*: Finally, to make matters even worse, one cannot expect the data to come from a fixed distribution. In the real-world of distributions drug design environment the test molecules are compounds engineered based on previous (training set) results and therefore will have a distribution of their own. The training set is thus not fully representative of the test examples.

Many conventional machine learning algorithms are ill-equiped to deal with these problems. Often, algorithms employ a cost model and feature selection method which rely on the data being balanced. Many algorithms generalize poorly due to the high dimensionality of the problem, moreover also because of the problem size many methods are no longer computationally feasible. Finally, most machine learning algorithms cannot deal with training and testing data coming from different distributions.

---

[*] To whom correspondence should be addressed.

The task then is to design methods that try to overcome these three problems. To do this we employed a feature selection criterion, called the unbalanced correlation score, which attempts to take into account the unbalanced nature of the data and is simple enough to avoid overfitting (and computational expense). Secondly, we designed a classifier which takes into account the different distributions in the test data compared to the training data when performing feature selection and classification. This was done using a type of inference procedure called transduction (see e.g. Vapnik, 1998), as opposed to the usual technique of induction. Induction builds a model based only on the distribution of the training data, transduction also takes into account the test inputs, i.e. properties of the molecules whose activity we wish to predict. Combining these two techniques we obtained improved prediction accuracy compared to using one of the techniques alone, although even using the techniques alone we obtained improved performance compared to existing techniques we tried.

We focussed on a well studied data set that was part of the KDD Cup 2001 competition. KDD (Knowledge Discovery and Data Mining) is one of the premier meetings of the data mining community. It holds an annual competition where several large scale real-world benchmarks are made available. In 2001, there were three benchmarks. There were 116 competitors from academic institutions and industrial laboratories who analyzed the Thrombin benchmark, see KDD (2001). We had knowledge of their results and methods, making it difficult to compare our work directly. Nevertheless, we also make a short discussion concerning the competition's results.

MATERIALS AND METHODS

Data set

The data set used to examine the effectiveness of the prediction methods was provided by DuPont Pharmaceuticals for the KDD Cup 2001 competition. The problem is to predict whether a given drug binds to a target site on thrombin, a key receptor in blood clotting. See KDD (2001) for more information.

The data for the competition was split into a training and a test set. Each example (observation) has a fixed length vector of 139,351 binary features (variables) in $\{0, 1\}$ which describe three-dimensional properties of the molecule[†]. We will refer to examples that bind as having label $+1$ (and hence being called positive examples). Conversely, negative examples (that do not bind) are labeled -1.

In the training set there are 1909 examples, 42 of which bind. Hence the data is rather unbalanced in this respect (42 positive examples is 2.2% of the data).[‡] The test set contains 634 additional compounds that were in fact generated based on the assay results recorded for the training set.

An important characteristic of the data is that very few of the feature entries are non-zero (0.68% of the $1909 \times 139\,351$ training matrix). See the Supplementary information for further statistical analysis of the data set.

System assessment

The task is to determine which of the features are critical for binding affinity and to accurately predict the class values using these features.

Performance is evaluated according to a weighted accuracy criterion due to the unbalanced nature of the number of positive and negative examples. That is, the score of an estimate $\hat{y}$ of the labels y is:

$$\begin{aligned}\ell_{\mathrm{bal}}(y, \hat{y}) = \frac{1}{2}\left(\frac{\#\{\hat{y} : y = 1 \wedge \hat{y} = 1\}}{\#\{y : y = 1\}}\right) \\ + \frac{1}{2}\left(\frac{\#\{\hat{y} : y = -1 \wedge \hat{y} = -1\}}{\#\{y : y = -1\}}\right)\end{aligned}$$

where complete success is a score of 1. In this report we also multiply this score by 100 and refer to it as the percentage weighted success rate.

Methodology

Our analysis is comprised of learning to predict the labels on the test set by using a machine learning algorithm. Most machine learning algorithms have a number of tunable parameters (hyperparameters). The tunable parameters of these algorithms that we did not fix before-hand are chosen via cross validation. The positively and negatively labeled training examples are split randomly into n groups for n-fold cross validation such that as close to $1/n$ of the positively labeled examples are present in each group as possible (we call this balanced cross validation). This balanced version of cross validation is necessary as there are so few positive examples, otherwise some folds would not give meaningful results. The method is then trained on $n - 1$ of the groups and is tested on the remaining group. This procedure is repeated n times each time using a different group for testing, taking the final score for the method as the mean of the n scores. We use $n = 8$.

[†] Unfortunately, this is the only information given by the data set provider concerning the nature of the features. The reason for this was so that the competition would evaluate the effectiveness of the algorithms without advantage being gained from the designer's prior biological knowledge. However, one guess could be that the features are calculated by superimposing the compounds and measuring field values of pharmacophore properties at grid points. These are probably also supplemented with other features, like total hydrophobicity.

[‡] As we will see later in the final analysis the test set is not as unbalanced. There are in fact 150 positive and 484 negative examples (23.66% and 76.34% respectively).

Hyperparameters are thus chosen by selecting the values with the best score.

Note that our entire procedure has probably been biased by knowledge of the method of the KDD Cup Winner. This bias is difficult to avoid when analyzing a benchmark data set after conclusion of the competition.

In the next sections we will describe the methods that we used: feature selection algorithms, inductive classifiers and transductive classifiers.

Feature selection

We propose the following feature selection criterion, which we call the unbalanced correlation score: rank the features according to the criterion:

$$f_j = \sum_{y_i=1} X_{ij} - \lambda \sum_{y_i=-1} X_{ij} \qquad (1)$$

where the score for feature j is f_j, the training data is denoted as a matrix X where columns are features and examples are rows, and a larger score is assigned a higher rank.[§] We suggest to take λ very large in order to select features which have non-zeros entries only for positive examples. In this data set, $\lambda \geqslant 3$ achieves this goal. One then chooses d features by selecting those with the highest rank.

This score is an attempt to encode prior information that the data is unbalanced, has a large number of features and only positive correlations are likely to be useful.

Justification We can justify the unbalanced correlation score using methods of information theory (see e.g. Cover and Thomas, 1991). Entropy, as defined in information theory, measures the unlikeliness that an event will occur, being $-p_i \ln(p_i)$, where p_i is the probability of appearance of event 'i'. To compute the entropy of a given feature we will need to assign it a probability. A feature with a low probability of random appearance is unlikely to be randomly generated and is more likely to describe some underlying input–output correspondence. The probability of random appearance of a feature with an unbalanced score of $N = N_p - N_n$ (i.e. for $\lambda = 1$), where N_p and N_n are, respectively, the number of one entries associated to class $+1$ and class -1 samples in the feature, is:

$$P_1(T_p, T_n, N_p, N_n) = \binom{N_p + N_n}{N_p} \prod_{i=0}^{N_p+N_n-1} \frac{1}{T_p + T_n - i} \prod_{i=0}^{N_p-1} (T_p - i) \prod_{i=0}^{N_n-1} (T_n - i) \qquad (2)$$

where T_p and T_n are, respectively, the total number of positive and negative labels in the training set. This probability can be used to compare features that add up to the same value in the same N, but, to compare the probability of appearance of features that add up to a different value, we will need to compute the probability that a certain N might occur randomly. This probability is given by:

$$P_2(T_p, T_n, N) = \frac{1}{Tp + Tn} \sum_{i=0}^{\min(T_p - N, T_n - N)} P_1(T_p, T_n, \max(0, N) + i, \max(0, -N) + i) \qquad (3)$$

Finally, one can compute the entropy for each feature as $-P_1 P_2 \log(P_1 P_2)$. The entropy and unbalanced score will not reach the same features in any setting, because the unbalanced correlation score will not select samples with low negative N. However in this particular problem, they do reach a similar ranking of the features due to the unbalanced nature of the data. In Table 1, we show the first 6 features for both scores in which it can be seen than 5 out of 6 are the same ones (for 16 features, 12 coincide).

Table 1. Features selected by the unbalanced correlation score and the entropy criteria

	1	2	3	4	5	6
Feature UCS	**79 650**	**90 405**	**91 838**	3391	**27 150**	**29 152**
Feature Entr.	**79 650**	**90 405**	**29 152**	**91 838**	16 793	**27 150**

Informally, we can justify this score by noticing the data has very few non-zero values. This means to make a classifier using negative correlations would require a large amount of features, but it will be very easy for a feature to be noisy and appear (partially) negatively correlated. We therefore pay more attention to the positive correlations.

Multivariate unbalanced correlation Note that the feature selection algorithm described so far is univariate, the simplicity of such an algorithm reduces the chance of overfitting (see, e.g. Kohavi, 1995). On the other hand, if dependencies between the inputs and targets are too complex this assumption may be too restrictive.

We can extend our criterion to assign a rank to a subset of features rather than just a single feature to make the algorithm multivariate. This can be done by computing the logical OR of the subset of features S (as they are binary), i.e. $X_i(S) = 1 - \prod_{j \in S}(1 - X_{ij})$ and then evaluating the score on the vector $X(S)$. A feature subset which has a high score could thus be chosen using, for example, a greedy forward selection scheme (see, e.g. Kohavi, 1995).

Comparison techniques In our experiments we compare the unbalanced correlation score to that of a standard

---

[§] As suggested by one of the referees of this article, one could also consider more general versions of this criterion which converge to the Fisher score when the number of positive and negative examples become more equal.

(univariate) correlation score: the so-called Fisher score

$$f_j = \frac{(\mu_{j(+)} - \mu_{j(-)})^2}{(\sigma_{j(+)})^2 + (\sigma_{j(-)})^2}, \tag{4}$$

where $\mu_{(+)}$ and $\mu_{(-)}$ are the mean of the feature values for the positive and negative examples respectively, and $\sigma_{(+)}$ and $\sigma_{(-)}$ are their respective standard deviations. Note that in this score negative correlations are just as important as positive ones.

We also compared to some rather more sophisticated multivariate methods called RFE and AL0M developed as wrapper techniques for support vector machines ((Guyon *et al.*, 2002)). See the Supplementary results.

In each case, the algorithms are evaluated for different numbers of features d, in the range $d = 1, \ldots, 40$. This range was chosen as the task is to choose a small number of features in order to render interpretability of the decision function. Secondly, it is anticipated that a large number of the features are noisy and should not be selected. We note again however how we have been biased here by the results of the KDD competition itself.

(Inductive) Classification algorithms

The task may not simply be just to identify relevant characteristics via feature selection, but also (as in the KDD Cup problem) to provide a prediction system. This system can also be used to validate the quality of the selected features.

As our feature selection technique selects features with non-zero entries only for positive examples it leaves little information to train standard machine learning algorithms. We therefore selected one of the simplest of classifiers:

$$f(x) = 1, \quad \text{if } \frac{\sum_{i=1}^{d} x(i)}{d} > 0 \tag{5}$$

$$= -1, \quad \text{otherwise} \tag{6}$$

which assigns the prediction that a new molecule binds if any of the selected features (represented as a vector x, where $x(i)$ indexes the ith feature) were non-zero. We call this a (logical) OR classifier. (Note that we wrote it in this form because later we will also use it for real-valued inputs.)

The advantage this classifier has over more complex ones for this task is there is no training involved so it is less likely to overfit, moreover there are no hyperparameters to adjust. Finally, as it classifies examples positive given only the smallest amount of evidence (a single non-zero feature) it is quite suited to the weighted success criterion.

Comparison techniques In our experiments we compared a number of rather more sophisticated classification algorithms: support vector machines (SVMs), k-nearest neighbors (k-NN) and C4.5, a decision tree learner (see e.g Duda *et al.* (2001) for descriptions).

We compare with a version of (linear) support vector machines (SVMs, see e.g. Vapnik, 1998) that attempts to optimize the weighted success criterion following, e.g Brown *et al.* (2000). See the Supplementary results for more details.

To adapt k-NN to the prediction criterion (1) we considered the following modification: as usual the decision rule is to assign the class of the majority of the k-nearest neighbors $x_{i=1,\ldots,\ell}$ of a point x we wish to classify. However, we altered the distance measure so that if x_i is a positive example we scale the measure of distance to x by a parameter γ. By controlling γ one controls the importance of the positive class. In our experiments we looked at the maximum performance on the test set over the possible choices of γ and k to know the highest attainable success rate of k-NN. We refer to these results as k-NN*.

We performed a similar search procedure with SVMs, where we make a search over all possible values of the threshold parameter in the linear model after training, which we refer to as SVM*. However, no attempt was made to adapt C4.5.

Transductive inference

Most learning algorithms, such as the ones described in the previous section, use what is called inductive inference. In this procedure one is given labeled data from which one builds a general model, and then applies this model to classify previously unseen (test) data. Most well known machine learning algorithms are of this type, for example support vector machines, k-NN and C4.5 to name a few.

In transductive inference, in contrast to inductive inference, one takes into account not only the given (labeled) training set but also (unlabeled) data that one wishes to classify. Although there are many definitions of what is a transductive algorithm (see, e.g. Vapnik, 1998) in this article we will mean algorithms which use unlabeled data when building a model in order to improve predictions. Hence, in contrast to inductive inference, different models can be built when trying to classify different test sets even if the training set is the same in all cases. Note that a transductive method can but does not need to improve the prediction for a second independent test set of data: the result is not independent from the test set of data. It is this characteristic which could help to solve problem 3 (described in the Introduction): that the data we are given has different distributions in the training and test sets. This point is illustrated in Figure 1.

Transduction is not useful in all tasks: it depends on the expense of obtaining the extra unlabeled points relative to labeling them. In drug discovery in particular we believe it is useful. Developers often have access to huge databases of compounds, and compounds are

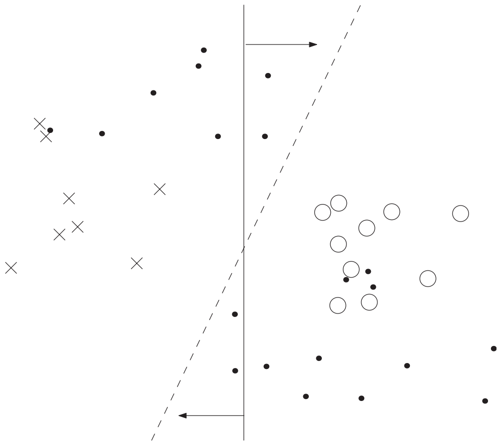

Fig. 1. Toy example demonstrating transductive inference. The training set is denoted as circle and cross symbols for the two classes. The test set, which has a different distribution than the training set, is denoted as dots, the labels of which are unknown. Inductive algorithms only use the training set, leading to decision rules like the solid line. Transductive algorithms take into account the positions of points in the unlabeled test set which can lead to improved decision rules such as the dashed line.

often generated using virtual Combinatorial Chemistry, so compound descriptors can be computed, even though the compounds have not been synthesized yet. In fact, unlabeled data has been used before for query learning (choosing which of the unlabeled points to label next), see e.g Warmuth *et al.* (2002). Drug discovery is an iterative process where the principal role of a machine learning method is to help choose the next test set. If the test set is the best that highly specialized experts can produce, it is certainly worth something to further improve it. Essentially using transduction can be seen as the second step in a two-step candidate selection procedure after a candidate test set has been produced, and its result is the final test set.

We propose to use a transductive scheme inspired by the ones used in Vapnik (1998); Jaakkola *et al.* (2000); Bennet and Demiriz (1998), and Joachims (1999). The algorithm is as follows.

Algorithm Firstly, we require an inductive classification algorithm C and an update function h. The classifier C should learn a real valued function $g(x)$ given data $x_i \in \mathbf{R}^N$, $y_i \in [-1, 1]$, $i = 1, \dots, \ell$. A label $y_i = \pm 1$ indicates high confidence that the ith example is labeled positive or negative; a value of $y_i = 0$ indicates no knowledge of the label; values in between describe differing degress of confidence. The magnitude of $g(x)$ should be proportional to the confidence in classification of assigning the label $y = 1$, if $g(x) > 0$, or $y = -1$ otherwise. Finally, the update function $h(x)$, which should be given as input a confidence level of the type output by $g(x)$, outputs a predicted label of the form $y \in [-1, 1]$ for the next iteration of the algorithm.

The iterative procedure is the following.

(1) One is given a training set of ℓ examples and a test set of m examples. Denote $n = \ell + m$ and $D = \{x_i, y_i\}_{i=1,\dots,n}$ where the first ℓ vectors are from the training set. Assign labels $y_i := 0, i = \ell + 1, \dots, n$ to the (unlabeled) test examples.

(2) Train the classifier C using the data D to obtain the function g.

(3) Update the test set labels, $y_i := h(g(x_i)), i = \ell + 1, \dots, n$.

(4) Repeat from step 2 until convergence or a maximum number of iterations have occurred.

(5) The final predictions for the test examples are given by $y_i = 1$, if $g(x_i) > 0$, or $y_i = -1$ otherwise.

The function h can be based on an estimate of the conditional probability of y given $g(x_i)$. We chose the function $h(x) = \tanh(s(x/n + b))$ where s and b were estimated using cross validation. This was done by choosing a grid of values of the free parameters and then choosing the instantion of the variables which resulted in the best weighted success rate. More details are given in the Supplementary information.

For the classifier C we choose the simple OR classifer given in the previous section in Equations (5) and (6) trained with the d features with the largest value of the unbalanced correlation score $f_j = \sum_{y_i>0} y_i X_{ij} + \lambda \sum_{y_i<0} y_i X_{ij}$ which is a real-valued generalization of Equation (1). Hence, $g(x) = (\sum_{i=1}^{d} x(i))/d$ is used as a measure of confidence in classification, whilst $f(x)$ defined in Equations (5) and (6) is used as a predictor. The weighted labels y_i of the test set are thus incorporated into the classifier C via the feature selection procedure. The transductive version of the OR classifier should improve results by incorporating extra information about the distribution of the test data. By combining the transductive classifier with our feature selection approach we hope to provide improved results over using only one of the techniques.

Comparison techniques We also tried using an SVM in conjunction with the unbalanced correlation score. This comparison was included to see if any improvements given by the transductive approach were universal, and as a sanity check to see if the OR classifier is a powerful enough classifier to predict accurately.

Table 2. Percentage weighted test success rate of algorithms after selecting $d = 2, \ldots, 40$ features with the unbalanced correlation score (cub)

	2	4	6	8	10	15	20	25	30	35	40
k-NN$_{cub}$	63.1	66.3	64.0	61.6	59.5	58.9	58.1	58.2	57.8	57.0	56.8
k-NN*$_{cub}$	72.8	75.0	76.6	74.5	73.9	72.3	64.1	59.5	58.5	58.0	56.8
SVM*$_{cub}$	72.8	75.0	76.6	74.5	73.9	73.2	64.1	60.8	57.8	56.5	55.2
OR$_{cub}$	72.8	75.0	76.6	74.5	73.9	72.3	64.1	59.5	58.5	55.9	56.7
TRANS-OR$_{cub}$	72.8	75.0	79.7	81.0	81.6	82.5	82.5	82.5	82.5	82.5	82.5

Table 3. Percentage weighted success rate of algorithms after selecting $d = 2, \ldots, 40$ features with the Fisher score (fish)

	2	4	6	8	10	15	20	25	30	35	40
C4.5$_{fish}$	50.0	52.4	52.4	51.7	50.6	48.2	51.9	48.7	48.6	49.0	49.0
k-NN*$_{fish}$	50.0	55.2	56.6	58.3	58.2	54.3	56.4	54.4	53.0	54.8	54.4
SVM*$_{fish}$	57.7	57.7	57.9	55.7	56.0	52.7	51.5	53.2	50.0	52.6	52.0
OR$_{fish}$	53.0	52.5	52.0	52.7	52.2	52.1	52.2	51.7	51.7	51.7	51.5

Summary of methods

In summary we have proposed two techniques: a feature selection score and a transductive classifier. They can be used together, in conjunction with a base classifier. Base classifiers we consider are: OR, SVM, C4.5 and k-NN. We also consider alternative feature selection schemes, in particular we study using the Fisher score. Finally, in the results section we will also briefly discuss the results of the KDD competition itself.

RESULTS

Accuracy of tested methods

Test performance of the algorithms is given in Tables 2 and 3. Table 2 shows the results of using the unbalanced correlation score to select $d = 1, \ldots, 40$ features and then training k-NN, SVMs, OR and TRANS-OR. Training C4.5 was also attempted but gave only 50% success rate for all feature set sizes, it appears this algorithm cannot deal well with our particular success criterion (which it was not designed for). The transductive method TRANS-OR performs best out of the methods, giving 82.5% balanced test success using 15–40 features. The model picked by cross validation (that gave the best cross validation success rate) had 10 features (the other free parameters were chosen to be $s = 4$ and $b = -0.15$), giving a test success of 81.6%. In comparison, the inductive version of the same classifier was a few percentage points worse for each value of d (the number of features selected) for $d > 4$. Note that the difference between the algorithms also becomes greater as d increases. This indicates that the transductive algorithm is consistently selecting more relevant features than the inductive one. The (usual) k-NN algorithm performed worse than both of these classifiers, reported here is the maximum success rate over choices $k = 1, \ldots, 8$. The modified k-NN algorithm (k-NN*), altered to maximize the weighted success rate as described in the Methods, perfomed slightly better. Reported are the maximum success rates over λ and k. The best results are still only comparable with the inductive OR classifier. SVMs and OR were also comparable. These comparisons justify that the OR classifier is not too poor a classifier for this problem.

Table 3 shows the results of using the same classification algorithms but using the Fisher score as a feature selector instead. Again C4.5 performs poorly, but this time SVMs and k-NN are slightly better than the OR classifier.

Further results We also conducted several other experiments, all these results are described in more detail in the supplementary material. Here, we will just briefly describe some of these points. Firstly, we also tested some more sophisticated multivariate feature selection methods (e.g from Guyon *et al.*, 2002) but the results were not as good as using the unbalanced criterion score. We tried using non-linear SVMs, but again, this did not improve results. Training an SVM using all the features results in only 50% success (chance level). We attempted to use SVMs as a base classifier for our transduction algorithm which was an improvement over using SVMs without transduction, and in fact the results were about as good as the best ones we achieved (using TRANS-OR$_{cub}$). In this experiment we also used the unbalanced correlation criterion with the parameter $\lambda = 1$, showing the score is somewhat robust with respect to this parameter. We also tried training the classifiers with larger numbers of features, and while in-

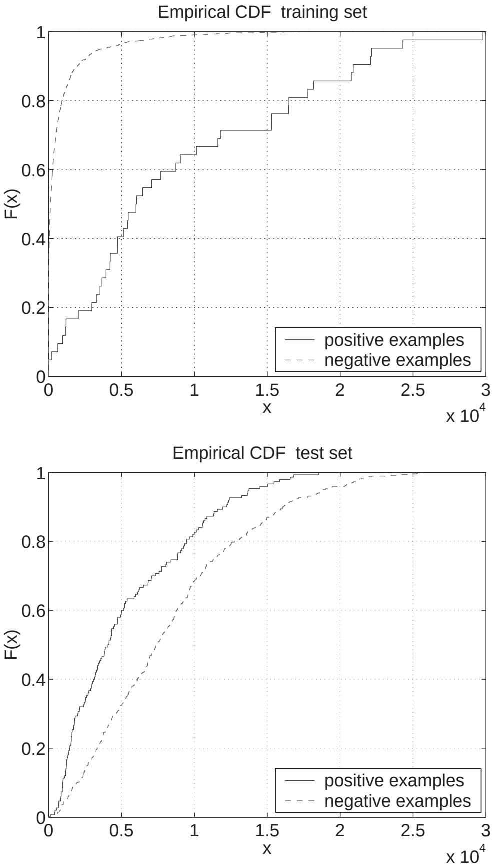

Fig. 2. Cumulative distribution functions for the random variable ‘number of nonzero features in the given examples’. The results show rather different distributions in the training and test sets.

ductive methods failed to learn anything after 200 features, the transductive methods continue to exhibit generalization behaviour up to 1000 features (TRANS-OR$_{cub}$ obtains 58% success with $d = 1000$ and 77% with $d = 200$), indicating that many of the features are in fact relevant to the problem, but can be difficult to find. We also compared the univariate and multivariate versions of the score (described in the Methods) and found an improvement when using the multivariate version.

Competition results

In the KDD competition itself only 7% of all entrants achieved a weighted success rate higher than 60%. The winner of the competition used a Bayes network; to be more precise, tree augmented Naive–Bayes (TAN). In Cheng *et al.* (1997a,b) the authors describe some of the Bayes network techniques which they employ. A summary of their procedure is given at http://www.cs.wisc.edu/∼dpage/kddcup2001/Cheng.ppt. Their result on the test set is a weighted success rate of 68.4%. Note that it is hard to compare with our results because we had knowledge of the competition results and methods.

Table 4. Distribution of one and zero attributes for the positive and negative classes with different feature sets: the features of TRANS-OR$_{cub}$ and 1000 randomly chosen features. Note the distribution changes between training and test set for the random features, which would hinder generalization

TRANS-OR$_{cub}$ solution					
Train set	+ve	−ve	Test set	+ve	−ve
1s	24.05%	0.11%	1s	56.93%	7.17%
0s	75.95%	99.89%	0s	43.07%	92.83%
Random features					
1s	6.47%	0.54%	1s	3.95%	5.90%
0s	93.53%	99.46%	0s	96.05%	94.10%

DISCUSSION

Data distribution

The data look rather different on the training and test sets, and do not appear to be i.i.d. To illustrate this, we consider constructing a single feature (random variable) from the original 139 351 binary features by summing them, for each example, i.e. we are left with the data $\hat{x}_j = \sum_{i=1}^{139\,351} X_{ji}$. One would expect two large samples from the same distribution to converge rather rapidly to the same empirical cumulative distribution functions for this random variable. We measured this on the training and test set, the results can be seen in Figure 2.

The training and test data are given in two separate plots, and for each data set the distributions for the negative and positive examples are also shown separately. Note that positive examples have a larger value of this variable than negative examples on the training set, but in the test set the opposite is the case. In the training set 10 of the 11 largest values of this variable belong to positive examples (and as there are only 42 positive examples this looks encouraging as a discriminative feature) but in the test set the largest 29 are all negative examples (and the test set is smaller in size but with more positive examples). Indeed, the two distributions for negative and positive examples almost switch characteristics from training to test set. See the Supplementary information for more details.

What can we conclude from this? With respect to the discriminative ability of features exhibited in both training and test set we can conclude there are many noisy features. These features appear to be discriminative on the training set (are +1 for positive examples and 0 for negative examples) but do not have the same behaviour on the test set. In this situation where many features may not be i.i.d. it is still possible that there exist a few features that are (or are close to) i.i.d. We would conclude that to perform well on this data set one may be required to perform feature selection to find these type of features.

Distribution of selected features

Let us now consider the features found by the model we selected by cross validation (TRANS-OR$_{cub}$ with $d = 10$), The question is: do the features selected by this method appear to be more i.i.d. ? Is this why it generalizes well? To try to answer these questions we measured the mean percentage of zeros and non-zeros across the chosen features separately for each class on both the training and test sets. The results are given for the features from the selected model, and from 1000 randomly selected features in Table 4.

The results in part suggest that the selected features are indeed more consistent than randomly chosen features. Although the number of non-zeros in positive examples increases from around 24% in the training set to 57% in the test set, the same trends are apparent in both the training and test sets (interestingly in the inductive algorithm a decrease is observed, see the Supplementary information). In the random features, on the other hand, one finds less correlation. For the training set the percentage of nonzeros for the positive examples is larger than the percentage of non-zeros for the negatives (6.47 versus 0.54%) whereas in the test set the converse is true (3.95 versus 5.90%). This makes discrimination very difficult. In the chosen features no such adverse affect is inherent and consequently improved generalization ability is observed.

Summary

Prediction of molecular bioactivity for drug design is a difficult problem which requires non-general techniques to obtain reasonable performance. The data is highly unbalanced, has a very large number of features and, although unlabeled data can be available to augment the labeled training data, it can have a rather different distribution. Of the large number of features, only a few seem to be relevant to the problem. Complex feature selection methods overfit and simple standard methods as the Fisher score also do not perform well. Constructing a criterion which does not overfit but also takes into account the unbalanced nature of the data when selecting features improves performance. Secondly, using transduction to take into account the distribution of the (unlabeled) test data to both build predictions and select features also improves results. Using both techniques together helps to select features which have similar distributions between training and test set, which can help to explain the positive results observed. However, we believe these results are relevant more from a general point of view (in terms of building algorithms which are suited to the problem) than in terms of their specific performance on this problem.

Transduction algorithms in the i.i.d. setting have produced only modest improvements over inductive algorithms. It is possible that their real strength lies in the type of problem studied in this article. We note that in this setting standard methods for measuring generalization performance such as cross validation on the training set are no longer reliable methods for model selection. Finally, transduction in the context of the data set analyzed can be seen as the second step in a two-step candidate selection procedure after a candidate test set has been produced using another selection / prediction method. It is possible that more direct ways of performing (combining) this procedure can be derived, and they remain the subject of further research.

ACKNOWLEDGEMENTS

The authors would like to thank DuPont Pharmaceuticals for providing the KDD Cup data. We also thank Alex Zien and Vladimir Vapnik for their help with this work.

REFERENCES

Bennet,K. and Demiriz,A. (1998) Semi-supervised support vector machines. In Kearns,M.S., Solla,S.A. and Cohn,D.A. (eds), *Advances in Neural Information Processing Systems, 11*. MIT Press, Cambridge, MA, pp. 368–374.

Brown,M.P.S., Grundy,W.N., D.,Lin, N.,Cristianini, C.,Sugnet, T.,S.Furey, Ares Jr,M. and Haussler,D. (2000) Knowledge-based analysis of microarray gene expression data using support vector machines. *Proc. Natl Acad. Sci. USA*, **97**, 262–267.

Cheng,J., Bell,D.A. and Liu,W. (1997a) An algorithm for Bayesian belief network construction from data. In *Proceedings of AI & STAT'97*.

Cheng,J., Bell,D.A. and Liu,W. (1997b) Learning belief networks from data: An information theory based approach. In *Proceedings of ACM CIKM'97*.

Cover,T.M. and Thomas,J.A. (1991) *Elements of Information Theory*. Wilson, New York.

Duda,R.O., Hart,P.E. and Stork,D.G. (2001) *Pattern Classification*. Wiley, New York.

Guyon,I., J.,Weston, S.,Barnhill and Vapnik,V. (2002) Gene selection for cancer classification using support vector machines. *Machine Learning*.

Jaakkola,T.S., Meila,M. and Jebara,T. (2000) Maximum entropy discrimination. In Solla,S.A., Leen,T.K. and Müller,K.R. (eds), *Advances in NIPS 12*. MIT Press, Cambridge, MA.

Joachims,T. (1999) Transductive inference for text classification using support vector machines. In Bratko,I. and Dzeroski,S. (eds), *Proceedings of the 16th International Conference on Machine Learning*. Morgan Kaufmann, San Francisco, pp. 200–209.

KDD (2001) Annual KDD cup, http://www.cs.wisc.edu/∼dpage/kddcup2001/.

Kohavi,J. (1995) Wrappers for feature subset selection. *AIJ issue on relevance*.

Vapnik,V. (1998) *Statistical Learning Theory*. Wiley, New York.

Warmuth,M.K., G.,Rätsch, Mathieson,M., Liao,J. and Lemmen,C. (2002) Active learning in the drug discovery process. In Becker,S., Dietterich,T.G. and Ghahramani,Z. (eds), *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA.