Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Biomedical Informatics 37 (2004) 224–239
Journal of Biomedical Informatics
www.elsevier.com/locate/yjbin

Classification and knowledge discovery in protein databases

Predrag Radivojac[a,1], Nitesh V. Chawla[b,2], A. Keith Dunker[c], Zoran Obradovic[a,*]

[a] Center for Information Science and Technology, Temple University, USA
[b] Customer Behavior Analytics, Canadian Imperial Bank of Commerce, Canada
[c] Center for Computational Biology and Bioinformatics, Indiana University School of Medicine, USA

Received 22 April 2004

Abstract

We consider the problem of classification in noisy, high-dimensional, and class-imbalanced protein datasets. In order to design a complete classification system, we use a three-stage machine learning framework consisting of a feature selection stage, a method addressing noise and class-imbalance, and a method for combining biologically related tasks through a prior-knowledge based clustering. In the first stage, we employ Fisher's permutation test as a feature selection filter. Comparisons with the alternative criteria show that it may be favorable for typical protein datasets. In the second stage, noise and class imbalance are addressed by using minority class over-sampling, majority class under-sampling, and ensemble learning. The performance of logistic regression models, decision trees, and neural networks is systematically evaluated. The experimental results show that in many cases ensembles of logistic regression classifiers may outperform more expressive models due to their robustness to noise and low sample density in a high-dimensional feature space. However, ensembles of neural networks may be the best solution for large datasets. In the third stage, we use prior knowledge to partition unlabeled data such that the class distributions among non-overlapping clusters significantly differ. In our experiments, training classifiers specialized to the class distributions of each cluster resulted in a further decrease in classification error.

Keywords: Classification; Feature selection; Noise; Class imbalance; Clustering; Class-distribution estimation

1. Introduction

One of the major objectives of bioinformatics is the automated characterization of a large number of proteins available from numerous databases. The ultimate goal of such a characterization is a detailed understanding of protein function and its complex network of interactions with other molecules in biochemical pathways. As a consequence of advanced technology and genome research, protein targets now include sequences associated with particular disease conditions or even putative proteins mapped from open reading frames that encode genes of various organisms. Hence, despite steady experimental efforts such as the protein structure initiative [1], the accelerated growth of data has given rise to the wide application of statistics, machine learning, and data mining to molecular biology data.

Machine learning and data mining approaches have been successfully applied to various tasks involving protein structure and function. Predictions of secondary structure [2], tertiary structure [3], protein disorder [4,5], relative solvent accessibility [6], switch sequences [7], number of contacts between amino acids [8], cleavage [9] or phosphorylation sites [10] are computational methods aimed at explaining biological phenomena, but also at reducing the cost of experimental research. Most of these approaches rely on well-known statistical techniques or various machine-learning approaches, e.g., linear regression, single or ensembles of neural networks, or support vector machines. Other techniques

* Corresponding author. Fax: +1 215 204 5082.
E-mail address: zoran@ist.temple.edu (Z. Obradovic).
[1] Present address: Indianna University School of Informatics.
[2] Present address: Department of Computer Science and Engineering, University of Notre Dame.

1532-0464/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jbi.2004.07.008

such as sequence profiles [11,12] or hidden Markov models [3,13] have been used for modeling protein homology or recognition of structurally/functionally important sequence motifs, while clustering algorithms [14,15] have been employed towards identifying common characteristics of biological sequences.

This study aims to address several issues that frequently occur in the prediction of protein's structural and functional properties. Restricting ourselves to a two-class classification framework, we put an emphasis to noisy and high-dimensional protein datasets where one class (positive or minority class) is underrepresented and small, while the other class (negative or majority class) is arbitrarily large. We present a complete machine learning framework for such datasets, thus countering issues from data preparation to prediction and exploiting unlabeled data. In the feature selection stage we employed Fisher's permutation test and compared its performance to other popular techniques. To address noise and class asymmetry, we combined minority class over-sampling with majority class under-sampling and showed that the increase in dataset size, together with the addition of synthetic examples, is often beneficial to model learning. The performance of logistic regression models, decision trees, and neural networks was systematically evaluated with the result that ensembles of logistic regression models favorably compared to other classifiers in cases of small- to medium-sized datasets. On the other hand, neural networks achieved the best results on the large datasets. Finally, we provided a mechanism for combining biologically related learning tasks in which the improved performance on the data with a relatively small number of labeled minority examples was achieved through a prior-knowledge based clustering of unlabeled data combined with estimation of class priors and predictor construction for each cluster.

This paper is structured as follows. Section 2 provides basic information about characteristics of protein data, followed by a brief overview of feature selection methods and a discussion on learning from class-imbalanced datasets. In Section 3 our methods for feature selection, strategies for learning from noisy and imbalanced data, and classification algorithms are described. Techniques for combining clustering, estimation of class priors, and predictor construction towards improved classification results are also presented. Section 4 describes the experimental setup and presents the most important results of the study. Finally, concluding remarks are contained in Section 5.

2. Background

In this section, we discuss characteristics of protein datasets and data representation most commonly used in this area. In Section 2.2, the basics of feature selection methods are introduced, while an overview of approaches used to address learning from skewed class distributions is contained in Section 2.3.

2.1. Characteristics of protein datasets

One of the common characteristics of protein datasets is that they are often noisy, high-dimensional, sparse, and class imbalanced. In general, three sources contribute to the noise in protein data: (i) biological complexity and variability (protein modification upon transcription, conclusions based on organism, gender or tissue specific cells, etc.); (ii) limitations of experimental procedures (sample preparation protocols, techniques such as X-ray crystallography or NMR[3] spectroscopy, etc.); and (iii) human error (lab conditions, misinterpretation of results, database labeling and curation, etc.). High dimensionality and sparsity of protein datasets are often consequences of so-called orthogonal (binary) data representation [16] which is predominantly used in this area. Each locus in a protein is represented by a 20-bit vector in which the observed amino acid is represented by a one and the remaining amino acids are represented by zeros (e.g., for alanine the representation is 10000000000000000000). Predictions for each residue in a protein are then based on all amino acids within a window of length w centered at that residue (windows of odd lengths are typically considered). Consequently, orthogonal data representation produces a high-dimensional sample with $20 \cdot w$ features, $19 \cdot w$ of which are zeros. It also introduces noise since in such a representation long-range sequence interactions are ignored. Additional attributes can be added to orthogonal representation to account for terminal (asymmetric) windows or prior knowledge. Finally, as a result of experimental constraints or due to uncommonness of certain events, protein datasets are often imbalanced, i.e., the numbers of available examples with different class designations are not approximately equal. For example, in secondary structure prediction, the percentages of available residues in α-helices, β-sheets, and coils are roughly 35, 20, and 45%; while in predicting intrinsically disordered regions, the ratio of ordered versus disordered residues in Protein Data Bank [17] is approximately 20:1. (Disordered residues are residues with unstable 3-dimensional conformation, either on a secondary or tertiary structure level, and can generally be extracted from the crystal-structure data as the residues with missing atomic coordinates [18]).

Another important characteristic of protein datasets is high redundancy. Proteins whose sequence identity is above ~30% are homologous with high probability.

[3] *Abbreviations used:* NMR – nuclear magnetic resonance, PAC – probably approximately correct.

Homologous proteins originate from gene duplication and/or speciation; they share a common ancestry and often carry out similar or identical functions. Additionally, many proteins correspond to different disease states or are engineered to facilitate lab experiments. Such proteins may easily form a large body of redundant data, which can lead to unrealistically high estimates of performance results. To account for data redundancy, it is a common practice to impose a threshold on sequence identity or to introduce some other measure of sequence similarity such that the analysis is performed on non-redundant sequences. The same was done in this study.

2.2. Feature selection for high-dimensional data

High-dimensional problem representation and the collection of high-dimensional data often require feature selection aimed at reducing the curse-of-dimensionality problems [19–21]. In addition, a large number of features can present a scalability issue to the learning algorithm. Feature selection has recently received considerable attention in the machine learning community.

Generally, methods for selecting subsets of features can be divided into wrappers, embedded methods, and filters [20]. Wrappers utilize the learning machine as a fitness function and search for the best features in the space of all feature subsets. This formulation of the problem allows the use of standard optimization techniques. Despite their simplicity and often having the best performance results [22], wrappers may suffer from overfitting and excessive computational complexity since the problem itself is NP-hard. In addition, wrappers highly depend on the inductive principle of the learning model. To reduce complexity, greedy approaches such as forward-selection or backward-elimination are often employed [19].

Embedded methods are incorporated into the learning procedure, and therefore are also dependent on the model. Many machine learning classifiers internally perform embedded feature selection, which may simply result in their weighting or a construction of composite features. Examples of embedded methods are CART [23] or the support vector machine approach of Guyon et al. [24]. In the recursive feature elimination approach of Guyon et al. [24], an initial model is trained using all features. Then, features are iteratively removed in a greedy fashion until the largest margin of separation is reached.

Filters are based on selecting the best features in one pass and are typically performed as a preprocessing step to model selection and learning. In domains such as text categorization or gene selection, filters are still dominant although combinations with both embedded methods and wrappers are appealing. Filters evaluate one feature at a time and estimate its usefulness for the prediction process according to various metrics. Recent papers on text categorization empirically evaluated several such metrics and suggested that information gain, χ^2 test, and bi-normal separation provide the best performance results [25,26]. We extend that work in this paper by evaluating the common techniques in the bioinformatics domain and by adding Fisher's permutation test as an alternative filter.

2.3. Classification methods for imbalanced data

The problem of class imbalance has to be carefully approached due to a possibility of considerable differences between class distributions in the labeled and unlabeled data, different costs of labeling and costs of classification for examples of each class [27–29], and a significantly degraded performance of some learners when the class distribution in the training data is heavily skewed [29,30].

There are two major groups of learning techniques designed to address class imbalance: supervised and unsupervised techniques. The supervised techniques have the knowledge of the class labels whereas the unsupervised techniques infer the labels for the minority class. The supervised techniques can be broadly categorized into three classes: (i) methods in which fractions of the minority and majority examples are controlled via under-sampling and/or over-sampling so that the desired class distribution is obtained in the training set, (ii) methods that use a recognition-based, instead of discrimination-based, inductive scheme, and (iii) methods that employ a cost-matrix to account for different costs of errors or examples [31,32].

An example of a supervised technique is the work of Kubat and Matwin [33] where majority examples were divided into four groups: noisy examples – examples with incorrect class designations, borderline examples – examples that are close to the class boundary (considered unreliable due to susceptibility to attribute noise), redundant examples – examples that repeat or do not introduce sample variability, and safe examples – examples that are worth keeping for the classification task. Noisy and borderline examples were detected via Tomek links [34] and together with redundant examples removed from the training data before the learning process began. In another study by Kubat et al. [35] a performance drop was detected with an increasing number of negative examples and the SHRINK system was proposed for detecting rare events. Their system concentrated on building a single "best positive region" and improved performance results with the increase of negative examples.

Another mechanism to overcome the curse of imbalance in the datasets is a combination of over-sampling and under-sampling [36–38]. Chawla et al. [36]

combined under-sampling and over-sampling techniques, but instead of simply replicating minority examples they generated artificial examples by interpolating between nearby minority examples. This approach proved to be particularly useful for improving the generalizability of decision trees and rule-learners. Chan and Stolfo [39] trained classification models such that their performance was optimized on a desired class distribution of unlabeled examples. They split the majority data into several disjunctive sets and trained a model on each set. Finally, models were combined via a second-stage classifier whose inputs were the predictions of individual models. A similar approach was used by Radivojac et al. [40] where an ensemble of classifiers was created. Each model was trained using all examples from the minority class and a random selection of examples from the majority class. In order to improve performance results on the minority class, Nugroho et al. [41] and DeRouin et al. [42] modified the architecture and the learning rate of a neural network. In a recognition-based approach, Japkowicz et al. [43] trained an autoassociator for each of the classes and then classified each new example based on the distance between the input example and the autoassociators' outputs. Thresholds for accepting or rejecting examples were optimized separately for every domain.

The second large class of techniques for detecting rare events involves an unsupervised framework. Although in most cases examples of only one class are not sufficient for successful learning in the PAC framework [44], it is possible to detect rare examples using unsupervised approaches. Initially, minority class examples are completely ignored (if available) and a model is trained using all the data from the majority class. Then, algorithms for outlier detection are employed where available minority examples may be used for threshold tuning. Techniques for outlier detection have been extensively studied in the field of statistics [45]. Statistical approaches first postulate an underlying probability distribution, and the outliers are detected as data points with small probabilities of occurrence. Depth-based [46] and distance-based [47] methods were also proposed, but just like the density estimation methods, they all may become inaccurate when the number of features increases, especially if a significant fraction of features are noisy. Breunig et al. [48] proposed an approach for detecting outliers based on the density of a data point's local neighborhood, while Aggarwal and Yu [49] searched for outliers in high-dimensional feature space by identifying them in lower dimensional projections of the data as examples with unusually low density. Finally, clustering methods such as CLARANS [50] or BIRCH [51] developed in the data mining community also detect outliers and then typically ignore them in the cluster construction. However, such algorithms are not optimized for the outlier detection.

3. Methods

We constrain our discussion to a standard two-class classification problem. Let D_L and D_U be the sets of labeled and unlabeled data. We define the dataset of labeled examples as $D_L = \{(\mathbf{x}_i, y_i) \mid i = 1,\ldots,n_L\}$, where $\mathbf{x} = (x_1, x_2, \ldots, x_k)^T \in X$ is a vector of features, $y \in Y = \{0, 1\}$ is the class designation, while n_L is the number of labeled data points. Using similar notation, $D_U = \{\mathbf{x}_i \mid i = 1,\ldots,n_U\}$, where $\mathbf{x} \in X$ and $n_U \gg n_L$ is the number of unlabeled data points. Also, let $\mathbf{p_L} = [p_L(0)\ \ p_L(1)]^T$ and $\mathbf{p_U} = [p_U(0)\ \ p_U(1)]^T$ be the imbalanced class distributions in D_L and D_U. We assume that, although the same type of sampling was used to generate both datasets, class distributions $\mathbf{p_L}$ and $\mathbf{p_U}$ may be significantly different. In protein datasets, this situation occurs frequently due to the considerably different costs of labeling examples of each class. The task of a classifier is to find a model $f(\mathbf{x})$ that best describes the data according to some performance measure. Generally, the model $f(\mathbf{x})$ maps X onto Y, but the internal task representation of each classifier can vary significantly.

3.1. Feature selection

Before discussing our approach, we briefly present three most successful filters for the feature selection process: information gain, χ^2 test, and bi-normal separation. For this purpose, we restrict X to a set of discrete k-dimensional vectors. (i) Information gain (average mutual information) between the feature $i \in \{1,\ldots, k\}$ and target variable y is defined as

$$I(x_i, y) = \sum_{x_i} \sum_{y} p(x_i, y) \cdot \log_2 \frac{p(x_i, y)}{p(x_i) \cdot p(y)}$$

and represents the expected amount of information (in bits) about the class designation if the only available knowledge about the query data point is its feature i. (ii) The χ^2 goodness-of-fit test is used in the feature selection process to measure statistical independence between the ith feature and the target. The test is based on calculating the χ^2 statistic which is defined as

$$\chi^2 = n_L \cdot \sum_{x_i} \sum_{y} \frac{(p(x_i, y) - p(x_i) \cdot p(y))^2}{p(x_i) \cdot p(y)}.$$

It can be shown that the χ^2 statistic follows the χ^2 distribution with $(|X_i| - 1)\cdot(|Y| - 1)$ degrees of freedom, where X_i represents the domain of the ith feature. Statistical significance of the test can be obtained using a lookup table. (iii) The bi-normal separation (BNS) is a recently proposed metric [26] for the cases when $X = \{0, 1\}^k$. The occurrence of each feature is modeled as a realization of a Gaussian variable exceeding a threshold. It is defined as

$$\mathrm{BNS}(x_i, y) = \left| F^{-1}\left(\frac{p(x_i = 1, y = 1)}{p(y = 1)}\right) - F^{-1}\left(\frac{p(x_i = 1, y = 0)}{p(y = 0)}\right) \right|,$$

where $F(\cdot)$ is a cumulative density function of the Gaussian distribution with zero mean and unit variance. As suggested by the author, $F^{-1}(0)$ is defined as 0.0005. Marginal and joint probabilities $p(x_i)$, $p(y)$, and $p(x_i,y)$ are in all cases calculated as relative frequencies.

3.1.1. Permutation test

Consider a data matrix D_L in which features are allowed to be real-valued or binary. In order to estimate the usefulness of a feature, we adopt a statistical approach and estimate the probability that its elements with different class designations are generated according to the same probability distribution. The lower the probability that the two samples are generated from the same distribution, the more important the feature is. More specifically, each feature column i of D_L is first divided into two vectors $\mathbf{u}$ and $\mathbf{v}$ according to the class designation. For example, let $\mathbf{u}$ be the elements with class 0 and $\mathbf{v}$ the elements with class 1. Vectors $\mathbf{u} = (u_1,u_2,\ldots,u_l)$ and $\mathbf{v} = (v_1,v_2,\ldots,v_m)$, where $l + m = n_L$, are assumed to be independent and identically distributed samples drawn from two probability distributions p_U and p_V. We would like to test the null hypothesis (H_0) that there is no difference between p_U and p_V. If, based on the available data and selected test statistic, H_0 cannot be conclusively rejected as highly unlikely, we conclude that sufficient evidence that p_U and p_V are different cannot be provided. The estimated significance level of the null hypothesis is used to rank the features.

The algorithm begins by choosing and calculating the test statistic θ, which in our case is the sample mean difference, i.e., $\theta = \bar{\mathbf{u}} - \bar{\mathbf{v}}$. Then, assuming that samples $\mathbf{u}$ and $\mathbf{v}$ were generated according to the same underlying distribution, they are concatenated into a single sample $\mathbf{w} = (\mathbf{u}, \mathbf{v})$ of size $l + m$. There are $(l + m)!/(l!{\cdot}m!)$ possible divisions of $\mathbf{w}$ into two parts of sizes l and m, each of which is equally likely under H_0. The achieved significance level (or p-value, p) of the statistical test is defined to be the probability of observing at least as large a mean difference by chance as θ. However, due to the sizes of samples $\mathbf{u}$ and $\mathbf{v}$, the exact significance level cannot be computed in most practical situations. In such cases, it is estimated using a fixed number of permutations (B) of the combined sample $\mathbf{w}$. In each step b ($b = 1,\ldots,B$), $\mathbf{w}$ is randomly shuffled and split into two parts $\mathbf{u}^*(b)$ and $\mathbf{v}^*(b)$ of lengths l and m. The test statistic $\theta^*(b)$ is calculated for each pair $\mathbf{u}^*(b)$ and $\mathbf{v}^*(b)$, and the p-value of the null hypothesis is finally estimated as the fraction of times $\theta^*(b) \geqslant \theta$ if $\theta > 0$ or $\theta^*(b) \leqslant \theta$ otherwise. The actually observed permutation $\mathbf{w}$ is included as the iteration $B + 1$.

This approach, called the permutation test, is a well-known statistical tool used to estimate whether two 1-dimensional samples were generated from the same distribution [52]. It was introduced by Fisher as an alternative statistical test in cases when the distribution of the data is not Gaussian. To determine how many permutations are enough for successful estimation of the p-value, Efron and Tibshirani [52] calculate the coefficient of variation as $\sqrt{(1-p)/(p \cdot B)}$. A coefficient of variation of 0.1, for example, indicates that the Monte Carlo estimation of the achieved significance level is within 10% of its true value. In such a case, in order to estimate a significance level of 0.05, approximately $B = 2000$ random permutations are required.

After the p-values of all k features are calculated, the features are ranked according to the ascending level of their p-values. Therefore, the most important features are the ones whose probability distributions of the components having different class labels are least likely to be identical.

3.2. Strategies for learning from noisy and class-imbalanced data

To address class-imbalance and noise we search for the best model in the supervised framework, using a combination of minority class over-sampling and majority class under-sampling. Classification models are trained on particular training class distributions, i.e., the fractions of majority and minority examples in training sets, and dataset sizes, and then the best performing model on a desired class distribution is selected depending on the achieved performance results. This approach is consistent with a simple and effective approach by Angluin and Laird [53] and Magdon-Ismail et al. [54] who characterized situations in which the increase of the size of noisy data caused an improvement in prediction results. Additionally, Anguin and Laird showed that the size of the training dataset in the PAC model grows with $1/(\varepsilon^2 - \eta^2)$, where ε and η represent the classification error and noise, respectively.

We denote class distribution in the training set as $\mathbf{p_T} = [p_T(0)\ p_T(1)]^T$, where $p_T(0)$ and $p_T(1)$ are relative frequencies of the majority and minority class examples. The combination of majority under-sampling and minority over-sampling, however, enables us not only to control $\mathbf{p_T}$, but also the size of the training set n_T. In order to evaluate different predictors, we introduce two parameters. The parameter us controls the amount of majority class under-sampling. For example, $us = n\%$ indicates that the majority class is reduced so that the size of the minority class represents $n\%$ of the size of the downsized majority class. After the majority population is reduced, a parameter os is used to define over-sampling of the minority class. For example,

$os = n\%$ means that $n\%$ of the minority class examples are added to the training set. No addition of minority examples amounts to $os = 0\%$. Based on these two parameters, the class distribution in the training set can be expressed as

$$\mathbf{p_T} = \left[\frac{100/us}{1 + 100/us + os/100} \; \frac{1 + os/100}{1 + 100/us + os/100}\right]^{\mathrm{T}},$$

while the size of the dataset represents $(1 + 100/us + os/100)\cdot 100\%$ of the original minority class size. For example, $us = 50\%$ combined with $os = 100\%$ corresponds to the training set where minority and majority classes are balanced, i.e., $\mathbf{p_T} = [1/2 \; 1/2]^{\mathrm{T}}$. On the other hand, the selection of $us = 25\%$ and $os = 300\%$ corresponds to the same class distribution, but with twice as large size of the training set.

While we use simple random under-sampling due to the non-redundant nature of the data, we consider two strategies for over-sampling the minority population: simple replication and the synthetic minority over-sampling technique (SMOTE) introduced by Chawla et al. [36]. In the case of simple replication, we randomly select the desired number of minority examples and add them to the training set. SMOTE, on the other hand, synthetically generates new minority class examples within the neighborhood of each minority class example, and is described in Section 3.2.1.

There are two opposing effects on model learning caused by the addition of synthetic minority examples. (i) The positive effect is that it enables using larger pools of majority data for a fixed training class distribution. At the same time it prevents overfitting that would occur due to the simply replicated minority examples (recall that the majority class is noisy and that as a result of its sheer size many majority examples may substantially penetrate the class region of the minority class). (ii) The negative effect is that synthetic examples inevitably introduce additional noise to the data and thus may hamper the learning process. The resulting effect of these two events, however, is hard to predict since it depends on the nature of the learning task, dataset size, class distribution, as well as on the type and amount of noise. Therefore, we believe that selecting the optimal parameters us and os should be done through a separate validation data.

3.2.1. The synthetic minority over-sampling technique

Sampling strategies such as replication and under-sampling are commonly used to counter the problem of imbalanced class distributions in a given dataset. The class imbalance, however, is closely related to the feature space and not merely to the data space. Consequently, replicating the minority class or under-sampling the majority class might not help in overcoming the inherent bias of a classifier towards the majority class. The sparsity of the minority class in the feature space would dictate that the new instances be created in order to further populate the feature space.

SMOTE works in the feature space and creates synthetic positive examples in the K-neighborhood of the desired number of true positive examples. In the case of continuous features, for each selected true positive example $\mathbf{x}$, one of the $K = 5$ nearest neighbors, $\mathbf{x}'$, is randomly chosen. Then, each feature i of the new synthetic example is constructed as $x_i + (x_i - x_i')\cdot\mathrm{rand}(0, 1)$. For the categorical features, the new examples are constructed by taking the majority vote of all the corresponding feature values among the five nearest neighbors. Thus, SMOTE has an effect of both populating and expanding the decision region of the minority class.

3.3. Learning algorithms

For the self-containment of this paper, here we briefly present the classifiers used in our study: logistic regression models, decision trees, and neural networks. We also use bagging [55] with each of the algorithms to see if the performance can be improved by considering simple ensembles.

3.3.1. Logistic regression

Logistic regression is a widely used statistical approach for classification. In two-class problems we use logit, the simplest version of logistic regression, in which the probability of class membership is defined as

$$p(y = 1|\mathbf{x}) = \frac{1}{1 + \mathrm{e}^{-\boldsymbol{\beta}^{\mathrm{T}}\cdot\mathbf{x}}},$$

where β is a $k \times 1$ vector of real-numbered coefficients. Assuming all n_{T} data points in the training set are equally likely and independent of one another, the optimal coefficients β^* are found by maximizing the following likelihood expression

$$l(\beta) = \prod_{i=1}^{n_{\mathrm{T}}} p(y_i \mid \mathbf{x}_i, \beta) = \prod_{i=1}^{n_{\mathrm{T}}} \left(\frac{1}{1 + \mathrm{e}^{-\boldsymbol{\beta}^{\mathrm{T}}\cdot\mathbf{x}_i}}\right)^{y_i} \cdot \left(1 - \frac{1}{1 + \mathrm{e}^{-\boldsymbol{\beta}^{\mathrm{T}}\cdot\mathbf{x}_i}}\right)^{1-y_i}.$$

After taking the logarithm, the above function is maximized using standard iterative optimization techniques. Here we use optimization based on the QR least-squares method which is accurate even in the cases of ill-conditioned data matrices [56]. Once the optimal coefficients β^* are found, classification of a query example $\mathbf{x}$ is based solely on the dot product $\beta^{*\mathrm{T}}\cdot\mathbf{x}$.

3.3.2. Decision trees

Decision trees are one of the most popular models used in the machine learning community. Model learning starts with all the training examples at the root node of a tree. The root node is then partitioned into several

groups based on the best single feature according to some measure such as information gain or gini-metric. After the data points are split, the procedure recursively continues for each of the newly generated nodes with only a subset of the examples from its parent node. The tree can be grown until some stopping condition is satisfied (e.g., minimum description length) or it can be fully grown and then pruned based on various criteria (e.g., error-based or rule-based pruning). Although single decision trees might not generalize well, it has been shown that ensembles of trees can significantly improve classification performance over single models. In this study, we employ the C4.5 classifier [57], which uses information gain for splitting and error-based pruning for overfitting prevention.

3.3.3. Neural networks

Two-layer feed-forward neural networks are universal approximators of bounded functions if provided enough data [58]. The expected number of data points necessary for successful learning is linear with the number of weights, but the worst-case scenario for difficult concepts requires an exponential number of examples. In addition, it is not always clear how to select the size of the hidden layer and parameters of the learning algorithm. Here we use the Levenberg–Marquardt algorithm [59] to train small- and medium-sized networks and the resilient back-propagation approach [60] for cases of large networks.

3.3.4. Bagging

Bagging is a well-known method of combining multiple predictors constructed over bootstrap samples drawn from the original dataset D, in which the aggregate prediction is obtained using majority voting by all trained models. Bagging can be successfully used with unstable learning algorithms and is generally a variance reduction mechanism [55]. Our choice of bagging over other ensemble methods (e.g., boosting) was motivated by its simplicity, small sensitivity to noise, and easy parallelizability of training in practical situations.

3.3.5. Influence of binary attributes on learning

Although feature selection may substantially reduce the number of attributes, the samples may still remain relatively high-dimensional and sparse. This situation is especially undesirable for logistic regression models since it may cause collinearity problems during training [61]. Therefore, we used an additional dimensionality reduction based on principal component analysis (PCA). Decision trees and neural networks were trained on the selected feature sets both with and without PCA.

3.4. Performance evaluation

A typical goal of a Bayesian optimal classifier [62] is to minimize the average cost or risk of applying a classifier to an unseen data point $\mathbf{x}$. The average cost C for a classifier $f(\mathbf{x})$ is given by

$$C = \sum_{i \in Y} \sum_{j \in Y} p(i|j) \cdot p_{\mathrm{U}}(j) \cdot c(i,j), \tag{1}$$

where indices i and j denote the predicted and actual class of the data point, $p_{\mathrm{U}}(j)$ is a prior probability of class j in the unlabeled dataset, and $p(i|j)$ is the conditional probability that predicted class is i given that the true class is j. The penalties of classifying a data point into the class i when the actual class is j are represented by a 2×2 matrix with elements $c(i, j)$. Minimization of the average cost requires precise knowledge of the a priori class distribution $\mathbf{p}_{\mathrm{U}}$ as well as of the penalties $c(i, j)$. However, in many practical situations the estimates of classification penalties may be hard to obtain so that it is reasonable to use $c(i, j) = 0$ if $i = j$, and $c(i, j) = 1$ otherwise. We refer to such a penalty matrix as a zero-one matrix.

We estimate the predictor's conditional probabilities $p(i|j)$ using cross-validation on out-of-sample test sets from the labeled dataset. The a priori class distribution $\mathbf{p}_{\mathrm{U}}$ can, on the other hand, be estimated experimentally in the lab [63] or computationally using unlabeled data [64,65]. Estimates $p(1|1)$ and $p(0|0)$ are commonly called sensitivity (sn) and specificity (sp) and are calculated in each cross-validation iteration from the confusion matrix. The final estimates are obtained by averaging over all iterations.

In cases of a zero-one penalty matrix, it is of interest to estimate predictor's performance for the general case of unknown priors. This is accomplished by plotting the receiver operating characteristics (ROC curves). The ROC curve is plotted as sn as a function of $1 - sp$ and thus shows the tradeoff between sensitivity and specificity that can be obtained using the same predictor, usually by shifting the decision threshold or by changing the class distribution in the training set. Given the imbalanced nature of protein data, an ROC plot may be more informative about predictor's performance than the expected cost as it allows one to visualize tradeoffs between sensitivity and specificity. The best predictor in the ROC sense closely follows the left-hand border and then top border of the diagram. To numerically evaluate ROC performance, we calculate the area under the curve (AUC) using the trapezoid rule.

3.5. Prediction of protein characteristics on large unlabeled datasets

In this section, we combine our strategy for learning from noisy and imbalanced labeled data with clustering and estimation of class priors from unlabeled data. We will show later that effective schemes for combining biologically related tasks can be presented through this

framework. Let D_L and D_U be the sets of labeled and unlabeled data as defined at the beginning of Section 3. Suppose now that D_U can be divided into n_C non-overlapping clusters, with $n_C \ll n_L \ll n_U$, such that the class distributions for any two clusters i and j differ. That is, $\mathbf{p}_{U,i} = [p_{U,i}(0)\ p_{U,i}(1)]^T \neq \mathbf{p}_{U,j} = [p_{U,j}(0)\ p_{U,j}(1)]^T$, for all $i, j \in \{1,2,\ldots,n_C\}$ and $i \neq j$. Let $c(\mathbf{x})$ be the algorithm that partitions D_U, i.e., maps X onto $\{1,2,\ldots,n_C\}$ according to some criterion. The predictor construction now proceeds separately for each cluster $D_{U,i}$ as follows.

Since the true class distribution $\mathbf{p}_{U,i}$ is unknown, we initially assume that it is balanced, i.e., $\mathbf{p}_{U,i} = [1/2\ 1/2]^T$, although prior knowledge can be used for a better guess. As a result of this assumption, we use a balanced training set, i.e., $\mathbf{p}_T = \mathbf{p}_{U,i} = [1/2\ 1/2]^T$, constructed from D_L to train a classifier $f_i(\mathbf{x})$. Important by-products of model training are its prediction accuracies on majority and minority classes, estimated using cross-validation. Following the notation of Section 3.4, we denote these two accuracies as $p(0|0)$ and $p(1|1)$. Now, we apply the predictor $f_i(\mathbf{x})$ to dataset $D_{U,i}$ and calculate the relative frequencies of predicted examples of both classes $\mathbf{q}_{U,i} = [q_{U,i}(0)\ q_{U,i}(1)]^T$. In the matrix form $\mathbf{q}_{U,i}$ can be expressed as

$$\mathbf{q}_{U,i} = \mathbf{P} \cdot \mathbf{p}_{U,i},$$

where

$$P = \begin{bmatrix} p(0|0) & 1-p(1|1) \\ 1-p(0|0) & p(1|1) \end{bmatrix}.$$

With known $\mathbf{P}$ and $\mathbf{q}_{U,i}$ an improved estimate of the class distribution $\mathbf{p}_{U,i}$ can be obtained as

$$\mathbf{p}_{U,i} = \mathbf{P}^{-1} \cdot \mathbf{q}_{U,i}.$$

This completes the first step of the algorithm. Starting from the newly estimated $\mathbf{p}_{U,i}$, it is iteratively applied until both the best predictor $f_i(\mathbf{x})$ and the estimate of the class distribution $\mathbf{p}_{U,i}$ converge. In each iteration, however, model $f_i(\mathbf{x})$ is not necessarily learned using $\mathbf{p}_T = \mathbf{p}_{U,i}$. Instead, the optimal class distribution in the training set is selected according to the ROC plots. In the same way, the optimal size of the training set n_T is selected based on an already characterized performance of the classifier. Since, in general $E[\mathbf{P}^{-1}] \neq E[\mathbf{P}]^{-1}$, improved estimates of $\mathbf{p}_{U,i}$ can be obtained using bootstrapping during both model training and estimating $\mathbf{q}_{U,i}$, as proposed by Vucetic and Obradovic [64]. Here, we modified their algorithm by using ROC plots to select $\mathbf{p}_T$ and data sampling to speed up the estimation. Non-convergence of $f_i(\mathbf{x})$ and $\mathbf{p}_{U,i}$ may indicate error in estimating $\mathbf{P}$ or a significant difference in data generators of D_L and D_U.

Finally, this iterative procedure is repeated for each cluster $D_{U,i}$, resulting in n_C classifiers $f_i(\mathbf{x})$, $i \in \{1,2,\ldots,n_C\}$, as illustrated in Fig. 1. Thus, the construction of a composite predictor $f(\mathbf{x})$ integrates partitioning of the unlabeled data with iterative estimation of the class distribution (from unlabeled data) and predictor construction (from the labeled data) for each of the n_C clusters. In Section 4.5 and 4.6, we consider two different types of partitioning, both based on prior knowledge, in the task of predicting phosphorylation sites from the amino acid sequence.

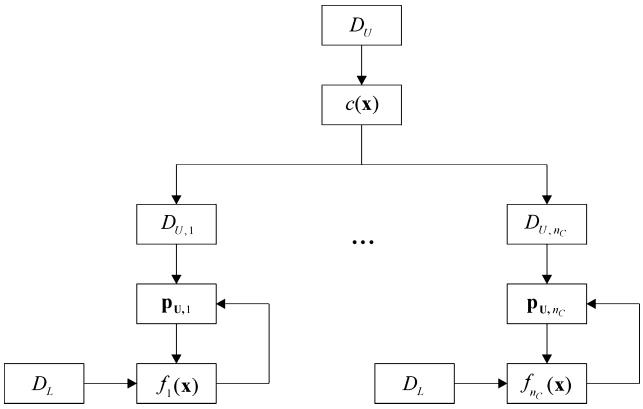

Fig. 1. Block-diagram of the composite classifier $f(\mathbf{x})$ whose construction incorporates partitioning unlabeled dataset D_U, and then iteratively combines estimation of class distribution $\mathbf{p}_{U,i}$ and model construction $f_i(\mathbf{x})$ for each cluster i. $f_i(\mathbf{x})$ is constructed using labeled dataset D_L and knowledge about $\mathbf{p}_{U,i}$. $c(\mathbf{x})$ is the clustering algorithm.

4. Experimental results and discussion

We start this section by discussing the biological meaning of the protein data that we used and details of the dataset construction process. Then, we evaluate all approaches from Sections 3.1, 3.2 and 3.3 using six protein datasets. Section 4.2 presents the comparison of various feature selection methods, while Section 4.3 evaluates our strategy for noisy and class imbalanced datasets and discusses the behavior of different classification algorithms. Finally, Sections 4.4 and 4.5 evaluate methods discussed in Section 3.5.

4.1. Datasets

We selected 6 datasets constructed by our group either previously or as a part of this study. All datasets are publicly available upon request.

1. PHOSS is a dataset of phosphorylation sites for the amino acid serine (S). A phosphorylation site represents a single amino acid (S, T or Y) to/from which the phosphate group can be attached/detached during cell regulation. A set of examples was constructed by combining PhosphoBase [66] with 832 phosphorylat-

able proteins extracted from Swiss-Prot [67]. Positive examples were created from 25-residue long fragments centered at all serine residues annotated as phosphorylation sites (the central residue was later excluded from the feature construction process). The remaining serine sites from the same set of proteins were included in the negative set, as phosphorylation was not observed despite assaying the protein. All fragments with sequence identity 30% or more with any other fragments were removed thus making the dataset non-redundant. In cases where one negative and one positive fragment were similar, the negative fragment was eliminated as less reliably labeled.
2. PHOST is a dataset of phosphorylation sites for amino acid threonine (T), extracted using the same procedure as for PHOSS.
3. PHOSY is a dataset of phosphorylation sites for amino acid tyrosine (Y), extracted using the same procedure as for PHOSS and PHOST. As compared to serine and threonine, phosphorylation of tyrosine may involve a different mechanism of attaching/detaching the phosphate group due to the presence of an aromatic ring in its structure. In addition, tyrosine residues are, as well as other aromatic amino acids, strongly associated with stable secondary structure and buried positions in a protein. On the other hand, threonine is moderately associated with the presence of fixed secondary structure (ordered regions), while serine is strongly associated with both surface position and lack of stable secondary structure, i.e., disordered regions [18].
4. Positive examples of the BOUNDARY dataset were constructed using a set of 24-residue long sequence fragments around order/disorder boundaries extracted from a set of 151 proteins containing 161 disordered regions at least 30 residues in length [40]. One half of the negative set was built from a set of 290 non-redundant completely ordered proteins [40] while the other half was built using non-boundary fragments from the existing set of 161 disordered regions. Balance of completely ordered and disordered segments in the negative set was maintained in order to prevent the predictor from adapting to protein disorder anywhere in the sequence.
5. The CAM dataset was built from 40 non-redundant proteins containing 42 calmodulin binding regions selected from the Calmodulin Target Database [68]. The set of positive instances was built using 42 regions that represent three classes of calmodulin targets whose binding activity depends on the concentration of calcium. The negative set consists of all residues not involved in calmodulin binding from the same set of 40 proteins. A sliding window of length 15 was used to create data examples. Seven terminal residues were excluded from all proteins.
6. The DISORDER dataset was constructed from 980 non-redundant proteins from the Protein Data Bank (PDB) characterized by X-ray diffraction. All residues in stretches from 3 to 30 whose atom coordinates were missing from the corresponding PDB files were assigned to the disorder class, while all other residues were assigned to order. This way of class labeling is highly susceptible to large amounts of noise due to the inability to account for crystal contacts, disorder-to-order transition upon binding with partners (that were crystallized together with target proteins), and ordered segments whose structure is hard to refine, e.g., wobbly domains [18]. The subset of proteins containing disordered regions does not overlap with the set of 151 proteins used for the BOUNDARY set. Data points were constructed using a sliding window of length 15, while seven residues at each terminus were excluded from all sequences. The number of ordered residues from any protein was limited to 100.

For all protein datasets we used orthogonal data representation. An overview of the datasets is presented in Table 1.

4.2. Evaluation of feature selection methods

We used Fisher's permutation test to select relevant features from the protein datasets. To substantiate our choice for using this approach, we estimated classification costs of the models in which the best features were selected according to the four criteria from Section 3.1: information gain, χ^2 test, bi-normal separation, and permutation test. For each of the feature selection criteria, we estimated classification costs in the cases in which the best 40, 60, 80, and 100 features were retained. In all cases the classifiers were constructed using ensembles of 30 neural networks, the a priori class probabilities in the unlabeled datasets were assumed to be equal, while the penalty matrix was assumed to be zero-one. Based on these parameters the class distributions of all training sets were set to $\mathbf{p_T} = [1/2\ 1/2]^T$. In order to make the comparisons feasible, we evaluated all methods using the four smaller datasets.

Table 1
Datasets: basic characteristics

Dataset	Number of features	Number of positive examples	Number of negative examples
PHOSS	480	613	10,798
PHOST	480	140	9051
PHOSY	480	136	5103
BOUNDARY	480	123	3386
CAM	300	942	17,974
DISORDER	300	4706	94,336

Table 2
Comparative performance evaluation for the four feature selection filters

Feature selection criterion	Wins − losses	Wins	Losses
Information gain	8	13	5
χ^2 test	−7	7	14
Bi-normal separation	−23	1	24
Permutation test	22	22	0

Classifiers were trained for PHOSS, PHOST, PHOSY, and BOUNDARY datasets using the best 40, 60, 80, and 100 features and their classification error was stored. For each pair (number of retained features, dataset) six pairwise comparisons among the feature selection methods were made. A win or a loss was assigned when there was statistically significant difference between the performances of the classifiers. Wins and losses were summed over all pairwise comparisons. The winning (losing) method was declared in 43 out of 96 comparisons.

The relative performance of the four feature selection schemes was compared using a standard pairwise win/loss methodology (e.g., see [21]). For each number of retained features, i.e., 40, 60, 80, and 100, we compared classification costs corresponding to all pairs of feature selection methods. For example, in the case when the best 40 features are retained, there are six pairwise comparisons among the four feature selection criteria for each of the four datasets. If one method in the pairwise comparison was significantly better than the other, it scored a "win" (+1 point), while if it was significantly worse it scored a "loss" (−1 point). Therefore, a win or a loss was assigned only in situations in which the two classification costs differed and there was no overlap in confidence intervals. The overall quality of each feature selection method was then expressed as the difference between the number of wins and the number of losses over all datasets and numbers of retained features (Table 2). This comparison strategy is sensitive to small differences between the algorithms.

4.3. Evaluation of different strategies for class imbalance

As a preprocessing step to model training and evaluation, a permutation test based feature selection procedure was performed on all datasets. We used p-value thresholds of 0.1 for the four smaller datasets and 0.001 for the two larger datasets. This reduced the dimensionality of the different datasets to the following: PHOSS–221, PHOST–91, PHOSY–112, BOUNDARY–175, CAM–132, and DISORDER–125 features. All models were trained using the reduced set of features.

In order to systematically evaluate performance of the approach from Section 3.2, we trained all classifiers using different sizes and class distributions of the training set. For each dataset, we evaluated all combinations of $us \in \{5, 10, 15, 25, 50, 75, 100, 125, 150, 175, 200, 300, 400\}$ and $os \in \{0, 100, 200, 300, 400, 500\}$ for a variety of learning parameters. However, due to space issues, the results summarized in Table 3 show the AUC scores only for the best performing models for each type of a classifier. Note that due to the separately performed feature selection and model training, the results in Table 3 may not represent unbiased estimates of the predictors' performance. However, this fact had a minor effect on the relative comparisons between the methods.

Logistic regression predictors benefited most from a combination of replication-based minority over-sampling in all cases, while the best performance of neural networks and decision trees was obtained using SMOTE. However, the best performance was not consistently observed for any specific minority over-sampling quantity. As discussed in Section 3, several factors contribute to this: dataset size, class distribution, concept, and type and amount of noise. Thus, in practical situations the optimal minority over-sampling (if necessary) has to be determined experimentally using a hold-out validation set.

The influence of ensemble averaging was considerably positive for all types of classifiers (Table 3). As expected, the smallest difference between single models and ensembles of models was observed for logistic regression, as it was the most stable classifier used in this study.

Interestingly, for five out of the six available datasets the best overall performance results were obtained for ensembles of logistic regression models (Table 3). Logistic regression is a robust procedure which is not sensitive to moderate amounts of noise in the training sets. High-dimensionality of the data and relatively small sample density for all five datasets likely had an effect that classifiers with moderate expressive power performed well enough. Neural networks require higher sample density in order to take advantage of their potential and this effect has been confirmed on DISORDER where they significantly outperformed other models. Finally, decision trees did not perform as well as the other two classifiers. Under-sampling and SMOTE shifted the inductive bias of the decision trees towards the minority class, which had a significant increase in sensitivity, but at a higher expense of specificity. Thus, neural networks and logistic regression models achieved a higher true positive rate at a lower false positive rate. Despite our efforts to prevent overfitting, decision trees still did not generalize well on the unseen, non-homologous protein data. Two representative examples of the ROC curves, corresponding to the experiments summarized in Table 3, are shown in Figs. 2 and 3.

4.4. Prediction of phosphorylation sites: clustering using functional keywords

Here we evaluate the approach from Section 3.5 in the prediction of the serine, threonine, and tyrosine

Table 3
AUC scores [%] for different classifiers on the six protein datasets

	Minority over-sampling (%)	Logistic regression		Decision trees		Neural networks	
		1 Model	30 Models	1 Model	30 Models	1 Model	30 Models
PhosS	0	78.9	80.1	**67.6**	**74.7**	75.4	79.6
	100	80.0	81.0	64.3	69.6	77.6	**80.6**
	200	80.5	81.0	63.8	70.3	77.6	80.0
	300	80.4	81.1	63.5	69.8	**78.7**	79.9
	400	81.0	81.0	63.8	70.0	78.2	79.6
	500	**81.1**	**81.2**	63.4	69.8	78.3	79.9
PhosT	0	82.7	85.7	72.0	78.0	75.7	84.6
	100	82.9	**85.9**	71.4	79.1	77.6	**85.4**
	200	**83.0**	85.3	71.2	79.5	79.8	83.4
	300	82.2	85.0	72.3	**79.6**	80.2	83.8
	400	82.4	84.2	**72.7**	78.4	**80.5**	82.7
	500	82.0	84.0	72.3	79.1	79.4	82.3
PhosY	0	83.9	88.1	70.4	76.5	74.6	84.6
	100	84.6	**88.3**	73.1	**78.5**	79.5	**86.5**
	200	85.8	88.1	72.3	77.8	79.5	85.9
	300	**85.9**	88.1	72.6	77.1	80.9	86.0
	400	85.4	87.9	72.9	76.8	**81.5**	85.5
	500	85.2	87.3	**73.6**	76.8	81.4	85.3
Boundary	0	73.4	77.4	61.1	66.5	62.6	77.1
	100	75.3	**78.8**	63.0	68.6	68.0	**77.8**
	200	74.5	77.8	63.3	69.4	72.7	77.3
	300	75.0	78.3	**64.5**	68.3	70.7	77.3
	400	**76.1**	78.0	64.1	68.5	71.4	77.2
	500	75.7	77.4	64.2	**69.9**	**73.3**	76.1
CaM	0	82.7	82.7	68.7	**76.1**	80.9	**82.4**
	100	82.8	83.2	69.3	74.7	81.7	81.9
	200	82.9	83.2	69.6	74.0	**81.9**	82.0
	300	**82.9**	83.1	**69.8**	73.3	81.5	81.8
	400	82.7	**83.3**	69.7	73.1	81.3	81.4
	500	82.5	82.8	69.8	73.3	81.2	81.4
Disorder	0	70.4	71.5	61.8	**66.1**	68.3	71.4
	100	70.4	71.4	61.8	65.0	70.3	75.9
	200	70.6	71.3	62.0	64.9	71.9	77.9
	300	70.7	**71.7**	**62.2**	64.6	72.6	79.1
	400	**70.9**	71.7	62.0	64.7	73.5	80.5
	500	70.7	71.3	61.7	63.9	**74.3**	**81.2**

Values in bold indicate best performing classifiers for each model type over six particular minority over-sampling amounts. For the logistic regression models, minority over-sampling was performed using replication, while for the decision trees and neural networks it was performed using SMOTE [36].

phosphorylation sites on a set of human proteins from Swiss-Prot. The objective of this experiment was two-fold. Its first part was designed to quantify the cost reduction (if any) in cases where the a priori class distribution is iteratively estimated and then used to train a classifier as compared to the situation where a predictor was simply trained using a balanced class distribution or the class distribution observed in the labeled dataset. The second part of the experiment was designed to test and quantify whether clustering unlabeled data combined with estimating class priors in each cluster can provide an additional decrease in classification cost.

Partitioning of the unlabeled data (human proteins) was performed according to functional categories associated with each protein in its Swiss-Prot entry, thus imposing a restriction that all data points from a single protein have to belong to the same cluster. We considered the following functional categories, indicated by the Swiss-Prot keywords: transcription (857 proteins), transport (593), structural (54), regulation (851), inhibitor (113), degradation (59), cytoskeleton (134), cancer (231), biosynthesis (245), membrane (179), and other. Group "other" contained proteins not associated with any of the functional keywords used for other groups and was ignored further. In total, this set of $n_C = 10$ functional subsets contained 3316 human proteins with $n_U = 3{,}116{,}794$ residues. Separated per residue, the set contained

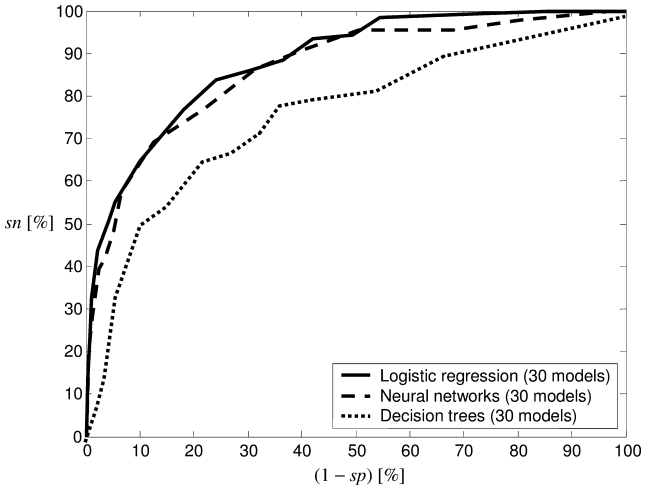

Fig. 2. ROC curves for the ensembles of logistic regression models, neural networks, and decision trees for the PHOSY dataset. Sensitivity (*sn*) and specificity (*sp*) were calculated for various fractions of minority and majority examples in the training set. Minority oversampling of 100% was used for all three classifiers (see Table 3).

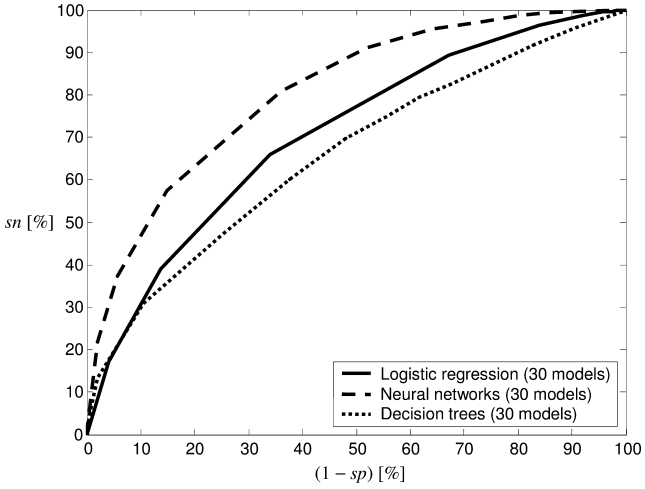

Fig. 3. ROC curves for the ensembles of logistic regression models, neural networks, and decision trees for the DISORDER dataset. Sensitivity (*sn*) and specificity (*sp*) were calculated for various fractions of minority and majority examples in the training set. Minority oversampling was 300% for logistic regression models, 0% for decision trees, and 500% for neural networks (see Table 3).

1,777,976 serine, 947,268 threonine, and 391,550 tyrosine residues.

In the first experiment for each phosphorylation predictor we compared three scenarios of model building: (i) the predictor is constructed using the balanced class distribution, i.e., $\mathbf{p_T} = [1/2\ 1/2]^T$, (ii) the predictor is constructed using the original class distribution in D_L, i.e., $\mathbf{p_T} = \mathbf{p_L}$, and (iii) predictor building and class distribution estimators are iteratively applied as described in Section 3.5. These scenarios were compared using the estimated average cost from Eq. (1). In the study of Vucetic and Obradovic [64] it was shown on artificial data that a similar class distribution estimator converged to within 1% of the true class distribution. Based on their results, we assumed that the estimated and true a priori class distributions were identical. Confidence intervals were estimated using bootstrapping. A comparison of the three scenarios for the regulatory proteins is shown in Fig. 4, while the complete results for all 10 clusters are given in the Appendix.

In the second experiment (Table 4) we compared two scenarios of model building: (i) the class distribution is estimated using the original unlabeled dataset, and (ii) the unlabeled data is initially clustered followed by class distribution estimation on each partition. In the first situation, the classification cost was estimated by directly applying Eq. (1), while in the second case it was calculated as a weighted average of the costs for each cluster. The weights were calculated as the fractions of residues belonging to each functional category.

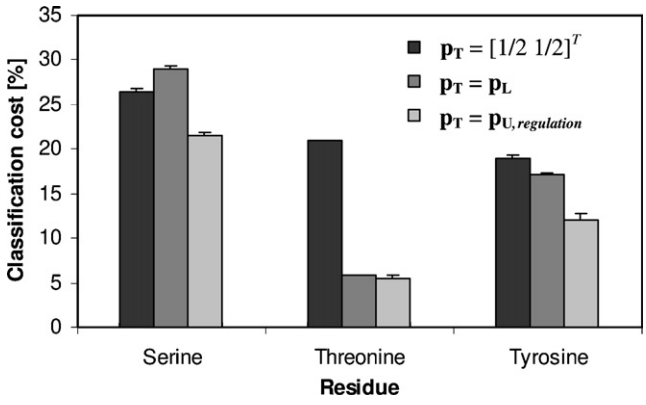

Fig. 4. Expected classification costs with standard errors for the three scenarios of model construction for the proteins involved in regulation. Scenario 1: $\mathbf{p_T} = [1/2\ 1/2]^T$ – training class distribution was balanced; Scenario 2: $\mathbf{p_T} = \mathbf{p_L}$ – training class distribution was identical to that in the labeled data; Scenario 3: $\mathbf{p_T} = \mathbf{p_{U,regulation}}$ – training class distribution is iteratively estimated for the regulatory proteins. The results signify estimating class priors during model construction.

Table 4
Expected classification costs and standard errors in the prediction of phosphorylation sites in human proteins

Method	Serine	Threonine	Tyrosine	Overall
With clustering	18.6 ± 0.0	5.9 ± 0.0	10.9 ± 0.0	13.8 ± 0.0
Without clustering	19.4 ± 0.0	6.4 ± 0.0	11.2 ± 0.1	14.4 ± 0.0

Class distributions in the unlabeled data were estimated both for the clustered and non-clustered cases. Clustering was performed using protein's functional class. Confidence intervals of 0.0 indicate values below 0.05.

The results from Fig. 4 suggest that estimating class priors in the unlabeled dataset may be an important step in model learning. Furthermore, clustering based on functional keywords provided a cost decrease of 4.1%. Separated per each phosphorylatable residue, the cost reduction was 4.1% for serine, 7.8% for threonine, and 2.7% for tyrosine sites (Table 4).

4.5. Prediction of phosphorylation sites: clustering using disorder prediction

In this section, we investigate the effects of clustering based on a different type of prior knowledge. We consider a set of 53,630 eukaryotic proteins from Swiss-Prot, again with the task of predicting phosphorylation sites. As a partitioning algorithm $c(\mathbf{x})$ we employ the VL2 predictor of long intrinsically disordered regions [15] which was designed to output the likelihood that a residue belongs to an intrinsically disordered region 30 consecutive residues or more in length.

Unlabeled dataset D_U was constructed using the original set of eukaryotic proteins. Then, $c(\mathbf{x})$ was applied to D_U with the effect that all residues with the prediction score below 0.5 were assigned group label "order", while all remaining residues were assigned group label "disorder". We denote these two partitions as $D_{U,o}$ and $D_{U,d}$. Therefore, in contrast to the setup of Section 4.4, we removed the restriction that all residues from the same protein must belong to the same cluster.

As in the previous section, we compare the clustered with the non-clustered version of the algorithm, applying estimation of class priors in both cases. The average cost for the non-clustered case was calculated by directly using Eq. (1) on the dataset D_U. On the other hand, for the clustered case the total cost was calculated as $C = C_o \cdot p(\text{order}) + C_d \cdot p(\text{disorder})$. Costs C_o and C_d were calculated using Eq. (1) on datasets $D_{U,o}$ and $D_{U,d}$, while $p(\text{order}) = 1 - p(\text{disorder})$ represents the fraction of eukaryotic residues predicted to be ordered by VL2. The results of this experiment are summarized in Tables 5 and 6. Confidence intervals were estimated using bootstrapping.

The results from Table 5 illustrate that partitioning by disorder prediction provided an overall cost reduction of 33.5%:38.1% for serine, 22.6% for threonine, and 10.9% for tyrosine sites. Table 6 also suggests that the estimates of the a priori class probabilities may provide an interesting "side-result" of the classification process: in all three cases, and especially for serine residues, regions predicted to be in long intrinsically disordered regions were estimated to be significantly more likely to undergo reversible phosphorylation than the regions predicted to have fixed secondary structure.

Table 5
Expected classification costs and standard errors in the prediction of phosphorylation sites in eukaryotic proteins

Method	Serine	Threonine	Tyrosine	Overall
With clustering	12.2 ± 0.2	4.8 ± 0.1	4.1 ± 0.1	8.1 ± 0.2
Without clustering	19.7 ± 0.4	6.2 ± 0.2	4.6 ± 0.1	12.2 ± 0.3

Class distributions in the unlabeled data were estimated both for the clustered and non-clustered cases. Clustering was performed on a per residue basis using a predictor of disordered regions VL2 [15].

5. Conclusions

In this study we designed a three-stage framework for automated annotation of protein databases and identified improvements achieved by our methods in each stage. In the first stage, we decided to search for the best feature subset using a filtering approach due to the high dimensionality of the sample and low information content of most individual features (properties of the data that can be unfavorable to the wrappers). In Sections 3.1 and 4.2 we employed Fisher's permutation test as a feature selection filter and compared its performance to that of information gain, χ^2 test, and bi-normal separation. We provided evidence that the permutation test fared well as compared to these techniques.

Fisher's permutation test has larger applicability than merely for the selection of binary features that were exclusively used here. It provides a probabilistic

Table 6
Characteristics of unlabeled datasets constructed from eukaryotic proteins

	Serine		Threonine		Tyrosine	
	Number of examples	Estimated priors	Number of examples	Estimated priors	Number of examples	Estimated priors
D_U	1,527,914	$[0.75\ 0.25]^T$	1,105,315	$[0.93\ 0.07]^T$	626,528	$[0.95\ 0.05]^T$
$D_{U,o}$	947,587	$[0.96\ 0.04]^T$	798,823	$[0.97\ 0.03]^T$	542,261	$[0.97\ 0.03]^T$
$D_{U,d}$	580,327	$[0.40\ 0.60]^T$	306,492	$[0.82\ 0.18]^T$	84,267	$[0.79\ 0.21]^T$

For every dataset, its size and estimated fractions of non-phosphorylated vs. phosphorylated sites are shown. D_U – unlabeled data consisting of S, T, or Y sites extracted from eukaryotic proteins; $D_{U,o}$ – unlabeled data predicted to be ordered; $D_{U,d}$ – unlabeled data predicted to be disordered by VL2 [15].

framework to the feature selection process and is generally insensitive to the underlying distribution of each feature. In addition, it does not require quantization of real-valued features. Thus, in cases when domain knowledge is used to construct multiple additional features, Fisher's permutation test can be used as a general off-the-shelf technique. A disadvantage of non-parametric tests lies in the computational complexity necessary to accurately evaluate the usefulness of the features. This problem may even become greater when datasets are large and when small differences between classes tend to be statistically significant (this may complicate automated threshold selection). However, these downsides can be overcome by combining multiple test statistics and data sampling. In addition, the computational burden can be alleviated using parallel processing.

In the second stage, we addressed the issue of noise and class imbalance. We compared different strategies of model learning that include choices of the number of examples used, class distribution in the training set, as well as of the learning algorithm. Surprisingly, in most cases the best performance results in terms of the AUC scores were achieved by ensembles of logistic regression models. These results can be seen in light of previous studies which recently showed that models based on linear discriminant analysis [69] and logistic regression [70] can be improved using bagging. The instability necessary for variance reduction comes from different selections of the training sets where on average approximately 63% of examples are selected in each bootstrap sample. A large-scale study that compares the performance of logistic regression and decision trees across various domains was done by Perlich et al. [70]. Ensembles of logistic regression models were shown to outperform ensembles of decision trees on small data sets while ensembles of decision trees outperformed logistic regression models on large data sets. The preference of each classification technique was shown to be separable by a simple measure of the signal-to-noise ratio (AUC score). Perlich et al. observed that decision trees generally outperformed logistic regression models when achievable AUC score is above 80%. Our study, however, concentrated on high-dimensional datasets on which logistic regression was found to significantly outperform decision trees regardless of the signal-to-noise ratio.

The results from Section 4.3 suggest that a combination of minority over-sampling and majority under-sampling is beneficial to model learning. In cases of logistic regression, the best results were obtained using simple replication of minority examples. On the other hand, neural networks and decision trees benefited from SMOTE [36]. We believe that, due to the small size of samples, low sample density had negative effect on network learning so that populating the feature space, even with the noisy synthetic data, facilitated the learning process; an effect previously analyzed and quantified by Magdon-Ismail et al. [54] on artificially generated datasets. The other factor that contributed to the learning process is that extra minority examples enabled using larger pools of majority data for the same class distribution in the training set. Finally, performance results achieved by decision trees were inferior to those of other models, including the case of an evidently non-linear concept for dataset DISORDER. The results obtained on all six datasets show, to our surprise, that even ensembles of decision trees could not outperform single logistic regression models or neural networks. Although further experimentation is needed for a definitive answer, we believe that such a performance is likely a consequence of the orthogonal data representation in which no single feature can provide particularly good class separation and correlation between the features may be substantial.

In the third stage, Sections 3.5, 4.4 and 4.5, we showed that estimation of the a priori class distribution of the unlabeled data has to be an integral part of model selection and learning. Learning on balanced samples or samples with the class distribution from the labeled data, can have a serious negative impact on the classification cost. In addition, our results indicate that integrating clustering with estimation of class priors can lead to significant improvements in classifier performance. The clustering methods explored here were based on functional keywords that were available in the Swiss-Prot database, but also on our earlier hypothesis that protein phosphorylation is related to intrinsically disordered protein regions [71]. Therefore, in the former case the predictor of phosphorylation sites was improved based on the assumption that proteins that carry out different functions should be associated with different fractions of phosphorylatable residues. In the latter case, the phosphorylation predictor significantly benefited from the disorder prediction, thus indirectly utilizing a significantly larger dataset available for constructing VL2 [15] for the prediction of phosphorylation sites. We believe that other types of prior knowledge can also be beneficial to predicting phosphorylation sites through this mechanism of combining related tasks.

Acknowledgments

This study was supported by NIH grant 1R01 LM07688 awarded to AKD and ZO and by NSF grant IIS-0219736 awarded to ZO and SV. We thank Lilia M. Iakoucheva and Timothy R. O'Connor for their contribution in constructing phosphorylation and calmodulin datasets.

Appendix

Expected classification costs and standard errors for the three scenarios of training class distributions $\mathbf{p_T}$

Functional class	Serine			Threonine			Tyrosine		
	C_1	C_2	C_3	C_1	C_2	C_3	C_1	C_2	C_3
Transcription	26.7 ± 0.3	25.2 ± 0.1	20.3 ± 0.4	20.9 ± 0.3	5.8 ± 0.1	5.5 ± 0.1	19.0 ± 0.3	18.8 ± 0.5	12.4 ± 0.6
Transport	27.4 ± 0.3	7.9 ± 0.0	7.8 ± 0.1	20.9 ± 0.3	7.8 ± 0.1	7.1 ± 0.2	19.0 ± 0.4	6.4 ± 0.1	5.9 ± 0.2
Structural	26.7 ± 0.3	23.3 ± 0.1	19.4 ± 0.3	20.9 ± 0.3	5.8 ± 0.1	5.3 ± 0.1	19.0 ± 0.3	4.7 ± 0.1	4.3 ± 0.1
Regulation	26.5 ± 0.3	29.0 ± 0.1	21.5 ± 0.4	20.9 ± 0.3	5.8 ± 0.1	5.4 ± 0.1	19.0 ± 0.4	17.2 ± 0.2	12.1 ± 0.4
Inhibitor	27.1 ± 0.3	15.6 ± 0.1	14.2 ± 0.2	20.9 ± 0.3	9.7 ± 0.1	8.5 ± 0.2	18.9 ± 0.4	8.2 ± 0.1	7.2 ± 0.2
Degradation	27.7 ± 0.3	2.1 ± 0.0	2.1 ± 0.0	20.9 ± 0.3	9.7 ± 0.1	8.1 ± 0.2	19.0 ± 0.4	7.4 ± 0.1	6.8 ± 0.2
Cytoskeleton	26.8 ± 0.3	22.3 ± 0.1	18.8 ± 0.3	21.0 ± 0.3	5.8 ± 0.1	5.7 ± 0.1	19.0 ± 0.4	4.7 ± 0.1	4.3 ± 0.1
Cancer	26.7 ± 0.3	25.2 ± 0.1	20.0 ± 0.4	21.0 ± 0.2	2.9 ± 0.0	3.0 ± 0.0	19.0 ± 0.3	6.5 ± 0.1	5.9 ± 0.2
Biosynthesis	27.7 ± 0.3	3.1 ± 0.0	3.1 ± 0.0	20.9 ± 0.3	9.7 ± 0.1	8.7 ± 0.2	19.0 ± 0.4	7.4 ± 0.1	6.9 ± 0.1
Membrane	27.3 ± 0.3	11.7 ± 0.1	11.2 ± 0.1	20.9 ± 0.3	9.7 ± 0.1	8.4 ± 0.2	19.0 ± 0.4	8.2 ± 0.1	6.8 ± 0.2

C_1 is the classification cost corresponding to the case when $\mathbf{p_T} = [1/2\ 1/2]^T$; C_2 corresponds to the case when $\mathbf{p_T} = \mathbf{p_L}$; and C_3 to the case when $\mathbf{p_T}$ is chosen according to the estimated $\mathbf{p}_{U,i}$ (separately estimated for each group i). Confidence intervals of 0.0 indicate values below 0.05.

References

[1] Burley S. An overview of structural genomics. Nat Struct Biol: Struct Genom 2000(Suppl):932–4.

[2] Rost B. Review: protein secondary structure prediction continues to rise. J Struct Biol 2001;134(2–3):204–18.

[3] Tsigelny IF, editor. Protein structure prediction: bioinformatics approach. La Jolla, CA: International University Line; 2002.

[4] Uversky VN. What does it mean to be natively unfolded? Eur J Biochem 2002;269(1):2–12.

[5] Obradovic Z, Peng K, Vucetic S, Radivojac P, Brown CJ, Dunker AK. Predicting intrinsic disorder from amino acid sequence. Proteins 2003;53(S6):566–72.

[6] Pollastri G, Baldi P, Fariselli P, Casadio R. Prediction of coordination number and relative solvent accessibility in proteins. Proteins 2002;47(2):142–53.

[7] Young M, Kirshenbaum K, Dill KA, Highsmith S. Predicting conformational switches in proteins. Protein Sci 1999;8:1752–64.

[8] Fariselli P, Casadio R. Prediction of the number of residue contacts in proteins. Proc Int Conf Intell Syst Mol Biol 2000;8:146–51.

[9] Blom N, Hansen J, Blaas D, Brunak S. Cleavage site analysis in picornaviral polyproteins: discovering cellular targets by neural networks. Protein Sci 1996;5(11):2203–16.

[10] Blom N, Gammeltoft S, Brunak S. Sequence and structure-based prediction of eukaryotic protein phosphorylation sites. J Mol Biol 1999;294(5):1351–62.

[11] Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 1997;25: 3389–402.

[12] Obenauer JC, Cantley LC, Yaffe MB. Scansite 2.0: Proteome-wide prediction of cell signaling interactions using short sequence motifs. Nucleic Acids Res 2003;31:3635–41.

[13] Eddy SR. Profile hidden Markov models. Bioinformatics 1998;14(9):755–63.

[14] Pellegrini M, Marcotte EM, Thompson MJ, Eisenberg D, Yeates TO. Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. Proc Natl Acad Sci USA 1999;96(8): 4285–8.

[15] Vucetic S, Brown CJ, Dunker AK, Obradovic Z. Flavors of protein disorder. Proteins 2003;52:573–84.

[16] Qian N, Sejnowski TJ. Predicting the secondary structure of globular proteins using neural network models. J Mol Biol 1988;202(4):865–84.

[17] Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, et al. The protein data bank. Nucleic Acids Res 2000;28(1): 235–42.

[18] Dunker AK, Brown CJ, Lawson JD, Iakoucheva LM, Obradovic Z. Intrinsic disorder and protein function. Biochemistry 2002;41 (21) :6573–82.

[19] Blum AL, Langley P. Selection of relevant features and examples in machine learning. Artif Intell 1997;97(1–2):245–71.

[20] Guyon I, Elisseeff A. An introduction to variable and feature selection. J Mach Learn Res 2003;3:1157–82.

[21] Hall MA, Holmes G. Benchmarking attribute selection techniques for discrete class data mining. IEEE Trans Knowledge Data Eng 2003;15(6):1437–47.

[22] Kohavi R, John G. Wrappers for feature selection. Artif Intell 1997;97(1–2):273–324.

[23] Breiman L. Classification and regression trees. Belmont, CA: Wadsworth International Group; 1984.

[24] Guyon I, Weston J, Barnhill S, Vapnik V. Gene selection for cancer classification using support vector machines. Mach Learn 2002;46(1–3):389–422.

[25] Yang Y, Pedersen JP. A comparative study on feature selection in text categorization. In: Proceedings of the 14th International Conference on Machine Learning; 1997. p. 412–20.

[26] Forman G. An extensive empirical study of feature selection metrics for text classification. J Mach Learn Res 2003;3: 1289–305.

[27] Fawcett TE, Provost F. Adaptive fraud detection. Data Mining Knowledge Discov 1997;3(1):291–316.

[28] Domingos P. MetaCost: a general method for making classifiers cost-sensitive. In: Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining. San Diego, CA, USA: ACM Press; 1999. p. 155–64.

[29] Weiss GM, Provost F. Learning when training data are costly: the effect of class distribution on tree induction. J Artif Intell Res 2003;19:315–54.

[30] Kubat M, Holte RC, Matwin S. Detection of oil spills in satellite radar images of sea surface. Mach Learn 1998;30:195–215.

[31] Elkan C. The foundations of cost-sensitive learning. In: Proceedings of the 17th International Joint Conference on Artificial Intelligence; 2001. p. 973–8.

[32] Japkowicz N, Stephen S. The class imbalance problem: a systematic study. Intell Data Anal 2002;6(5):429–50.

[33] Kubat M, Matwin S. Addressing the curse of imbalanced data sets: one-sided selection. In: Proceedings of the 14th International Conference on Machine Learning. Nashville, TN: Morgan Kaufmann; 1997. p. 179–86.

[34] Tomek I. Two modifications of CNN. IEEE Transactions on Systems, Man and Cybernetics 1976;SMC-6:769–72.
[35] Kubat M, Holte R, Matwin S. Learning when negative examples abound. In: Proceedings of the European Conference on Machine Learning, 1997, Prague, Czech Republic; 1997. p. 146–53.
[36] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 2002;16(2002):321–57.
[37] Ling C, Li C. Data mining for direct marketing problems and solutions. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining. New York, NY: AAAI Press; 1998.
[38] Solberg A, Solberg R. A large-scale evaluation of features for automatic detection of oil spills in ERS SAR images. In: International Geoscience and Remote Sensing Symposium. Lincoln, NE; 1996. p. 1484–86.
[39] Chan PK, Stolfo SJ. Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining; 1998. p. 164–8.
[40] Radivojac P, Obradovic Z, Brown CJ, Dunker AK. Prediction of boundaries between intrinsically ordered and disordered protein regions. Pac Symp Biocomput 2003;8:216–27.
[41] Nugroho AS, Kuroyanagi S, Iwata A. A solution for imbalanced training sets problem by CombNET-II and its application on fog forecasting. IEICE Trans Inform Syst 2002;E85–D(7):1165–74.
[42] DeRouin E, Brown J, Beck H, Fausett L, Schneider M. Neural network training on unequally represented classes. In: Dagli SRTKaYCS CH, editor. Intelligent engineering systems through artificial neural networks. New York: ASME Press; 1991. p. 135–41.
[43] Japkowicz N, Myers C, Gluck M. A novelty detection approach to classification. In: Proceedings of the 14th Joint Conference on Artificial Intelligence; 1995. p. 518–23.
[44] Shvaytser H. A necessary condition for learning from positive examples. Mach Learn 1990;5(1):101–13.
[45] Barnett V, Lewis T. Outliers in statistical data. 3rd ed. New York: John Wiley and Sons; 1994.
[46] Tukey JW. Exploratory data analysis. Reading, MA: Addison-Wesley; 1977.
[47] Knorr EM, Ng RT, Tucakov V. Distance-based outliers: algorithms and applications. VLDB J 2000;8(3–4):237–53.
[48] Breunig MM, Kriegel H-P, Ng RT, Sander J. LOF: identifying density-based local outliers. In: ACM SIGMOD International Conference on Management of Data 2000 Dallas, TX; 2000. p. 93–104.
[49] Aggarwal CC, Yu PS. Outlier detection for high dimensional data. In: ACM SIGMOD International Conference on Management of Data 2001. Santa Barbara, CA: ACM Press; 2001. p. 37–46.
[50] Ng RT, Han J. Efficient and effective clustering method for spatial data mining. In: 20th International Conference on Very Large Data Bases, 1994, September 12–15, Santiago, Chile; 1994. p. 144–55.
[51] Zhang T, Ramakrishnan R, Livny M. BIRCH: an efficient data clustering method for very large databases. In: ACM SIGMOD International Conference on Management of Data 1996. ACM Press; 1996. p. 103–14.
[52] Efron B, Tibshirani RJ. An introduction to the bootstrap. New York: Chapman & Hall; 1993.
[53] Angluin D, Laird P. Learning from noisy examples. Mach Learn 1988;2:343–70.
[54] Magdon-Ismail M, Nicholson A, Abu-Mostafa YS. Financial markets: very noisy information processing. Proc IEEE 1998;86(11): 2184–95.
[55] Breiman L. Bagging predictors. Mach Learn 1996;24:123–40.
[56] Green WH. Econometric analysis. 4th ed. Upper Saddle River, NJ: Prentice Hall; 2000.
[57] Quinlan J. C4.5: programs for machine learning. San Mateo, CA: Morgan Kaufmann; 1992.
[58] Cybenko G. Approximation by superpositions of a sigmoidal function. MCSS, Math Contr Signals Syst 1989;2:303–314.
[59] Hagan MT, Menhaj MB. Training feedforward networks with the Marquardt algorithm. IEEE Trans Neural Netw 1994;5(6): 989–93.
[60] Riedmiller M, Braun H. A direct adaptive method for faster backpropogation learning: the RPROP algorithm. Proc IEEE Int Conf Neural Netw 1993;1:586–91.
[61] Davidson R, MacKinnon J. Estimation and inference in econometrics. New York: Oxford University Press; 1993.
[62] Whalen AD. Detection of signals in noise. Academic Press: New York, NY; 1971.
[63] Marks F. Protein phosphorylation. New York, Basel, Cambridge, Tokyo: VCH Weinheim; 1996.
[64] Vucetic S, Obradovic Z. Classification on data with biased class distribution. In: Proceedings of the 12th European Conference on Machine Learning (ECML 2001), 2001 September 5–7, Freiburg, Germany; 2001. p. 527–538.
[65] Latinne P, Saerens M, Decaestecker C. Adjusting the outputs of a classifier to new a priori probabilities may significantly improve classification accuracy: evidence from a multi-class problem in remote sensing. In: Proceedings of the Eighteenth International Conference on Machine Learning. Williamstown, MA, USA: Morgan Kaufmann; 2001. p. 298–305.
[66] Kreegipuu A, Blom N, Brunak S. PhosphoBase, a database of phosphorylation sites: release 2.0. Nucleic Acids Res 1999;27(1):237–9.
[67] Boeckmann B, Bairoch A, Apweiler R, Blatter M-C, Estreicher A, Gasteiger E, et al. The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Res 2003;31: 365–70.
[68] Yap K, Kim J, Truong K, Sherman M, Yuan T, Ikura M. Calmodulin target database. J Struct Funct Genom 2000;1:8–14.
[69] Skurichina M, Duin RPW. Bagging, boosting and the random subspace method for linear classifiers. Pattern Anal Appl 2002;5:121–35.
[70] Perlich C, Provost F, Simonoff J. Tree induction vs. logistic regression: a learning-curve analysis. J Mach Learn Res 2003;4: 211–55.
[71] Iakoucheva LM, Radivojac P, Brown CJ, O'Connor TR, Sikes JG, Obradovic Z, Dunker AK. The importance of intrinsic disorder for protein phosphorylation. Nucleic Acids Res 2004;32(3):1037–49.