[Frontiers in Bioscience 13, 691-708, January 1, 2008]

Classification algorithms for phenotype prediction in genomics and proteomics

Habtom W. Ressom[1,2,3], Rency S. Varghese[1,2,3], Zhen Zhang [4], Jianhua Xuan[5], Robert Clarke[1,3,6]

[1]Lombardi Comprehensive Cancer Center, Georgetown University Medical Center, 3970 Reservoir Rd NW, Washington, DC 20057, [2]Department of Biostatistics, Bioinformatics, and Biomathematics, Georgetown University Medical Center, 3970 Reservoir Rd NW, Washington, DC 20057,[3]Department of Oncology, Georgetown University Medical Center, 3970 Reservoir Rd NW, Washington, DC 20057,[4]Department of Pathology, Johns Hopkins Medical Institutions, Baltimore, MD 21287,[5]Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Arlington, VA,[6]Department of Physiology and Biophysics, Georgetown University Medical Center, 3970 Reservoir Rd NW, Washington, DC 20057

TABLE OF CONTENTS

1. ABSTRACT

This paper gives an overview of statistical and machine learning-based feature selection and pattern classification algorithms and their application in molecular cancer classification or phenotype prediction. In particular, the paper focuses on the use of these computational methods for gene and peak selection from microarray and mass spectrometry data, respectively. The selected features are presented to a classifier for phenotype prediction.

2. INTRODUCTION

Molecular cancer classification enables automated grouping of samples into pre-specified categories (e.g. normal, benign, early-stage malignant, and late-stage malignant) on the basis of their molecular profiles. Technologies such as microarray and mass spectrometry have generated a large amount of data, enabling high-throughput molecular profiling for cancer classification. Various statistical and machine learning methods have been used to analyze the high dimensional data generated by these technologies for cancer classification. These methods have been shown to have statistical and clinical relevance in cancer detection for a variety of tumor types.

Most classification algorithms perform sub-optimally with thousands of features (genes/proteins). Thus, feature selection methods are used to identify features that are most predictive of a phenotype. The selected genes/proteins are presented to a classifier or a prediction model. By reducing the dimensionality of the feature space, we can (i) improve classification accuracy, (ii) provide a better understanding of the underlying concepts that generated the data, and (iii) overcome the risk of data overfitting, which arises when the number of features is large and the number of training patterns is comparatively small.

This review paper provides a survey of some computational methods that have been applied for molecular cancer classification. The paper is organized as follows: Sections 3 and 4 introduce pattern classification and feature selection methods, respectively. Section 5 presents methods to evaluate the performance of feature selection and pattern classification algorithms. Section 6 highlights some recent applications of these methods in microarray and mass spectrometry data analysis. Section 7 summarizes the review and discusses future trends in computational methods for feature selection and molecular cancer classification.

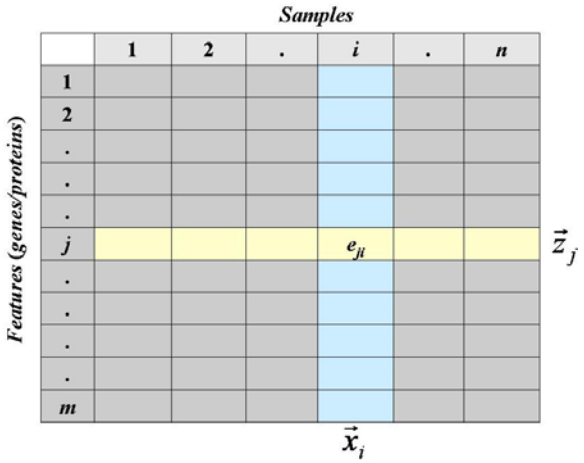

Figure 1. Data matrix with rows representing features (e.g., genes, mass points, proteins, etc.) and the columns representing samples (e.g., cell lines, tissues, subjects, etc.).

3. PATTERN CLASSIFICATION METHODS

In this section, we describe some computational methods that have been applied for molecular cancer classification via high-dimensional data. We explain how these methods are used during (i) the training phase, where the model structure and its parameters are determined using training examples and (ii) testing/operation phase, where the resulting model is applied for classification of testing samples not used for training. Here, we assume that the data are preprocessed and training and testing samples are predetermined.

Let the training examples be represented in a matrix form shown in Figure 1. The rows in the data matrix are features (e.g., genes, mass points, proteins, etc.) and the columns are samples (e.g., cell lines, tissues, subjects, etc.). Each cell in the data matrix represents an expression level. A feature and a sample are represented by an *n*-dimensional vector $\vec{z}$ and an *m*-dimensional vector $\vec{x}$, respectively; and $\vec{y}$ is an *n*-dimensional vector, whose elements are the class labels of all training samples. For example, the i^{th} sample, the j^{th} feature, and the class labels are represented by the following vectors, respectively:

$$\vec{z}_j = \begin{bmatrix} e_{j1} \\ e_{j2} \\ e_{j3} \\ \vdots \\ e_{jn} \end{bmatrix}, \quad \vec{x}_i = \begin{bmatrix} e_{1i} \\ e_{2i} \\ e_{3i} \\ \vdots \\ e_{mi} \end{bmatrix}, \quad \text{and} \quad \vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix},$$

where e_{ji} denotes the expression level for the *j*th feature and the *i*th sample and y_i is the class label for the *i*th sample. For binary classification, 1 and 0 or 1 and -1 are commonly used as class labels.

Linear discriminant analysis (LDA) is used to find the linear combination of features that best separates two or more classes of samples. Consider a set of observations $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$ where each observation has a known class *y*. The classification problem is then to create a predictor, so that any sample with observation $\vec{x}$ can be grouped to one of the predefined classes. Suppose two classes of observations ($y = 0$ and $y = 1$) have means $\vec{\mu}_0$ and $\vec{\mu}_1$ and covariances Σ_0 and Σ_1, respectively. The linear combination of features $\vec{w} \cdot \vec{x}$ will have means $\vec{w} \cdot \vec{\mu}_i$ and variances $\vec{w}^T \Sigma_i \vec{w}$ for $i = 0, 1$. Fisher (1936) defined the separation between these two distributions as the ratio of the variance between the classes to the variance within the classes:

$$S = \frac{\delta^2_{between}}{\delta^2_{within}} = \frac{\vec{w} \cdot (\vec{\mu}_1 - \vec{\mu}_0)^2}{\vec{w}^T (\Sigma_1 + \Sigma_0) \vec{w}}$$

This measure is, in some sense, a measure of the signal-to-noise ratio (SNR) for the class labeling. It can be shown that the maximum separation occurs when $\vec{w} = (\Sigma_0 + \Sigma_1)^{-1}(\vec{\mu}_1 - \vec{\mu}_0)$. Hence, assuming that the probability density functions $p(\vec{x}|y=1)$ and $p(\vec{x}|y=0)$ are both normally distributed, with identical full-rank covariances $\Sigma_0 = \Sigma_1 = \Sigma$, $\vec{w} = \Sigma^{-1}(\vec{\mu}_1 - \vec{\mu}_0)$. Thus, once $\vec{w}$ is determined through the training examples, the probability of an unknown sample $\vec{x}$ being in a class *y*, $p(y|\vec{x})$, is determined as a function of the linear combination of the known observations $\vec{w} \cdot \vec{x}$, i.e., $y = f(\vec{w} \cdot \vec{x})$, where *f* is often a simple threshold function that maps all samples above a certain threshold to class 1 and all other samples to class 0, or the vice versa.

Variants of LDA are used for pattern classification. For example, diagonal LDA (DLDA) is the same as LDA except that the covariance matrices are assumed to be diagonal. The quadratic discriminant analysis (QDA) method is an extension of LDA that allows for the groups of interest to have different population covariance matrices for the predictor variables. QDA uses quadratic equations, rather than linear equations.

Logistic regression is a statistical procedure subsumed by the generalized linear model that is structurally similar to linear regression, except that the outcome variable is a categorical variable as opposed to a continuous variable. In logistic regression it is assumed that the outcome variable follows a binomial distribution and that the log odds (or logit) can be described by a linear function of the logistic regression coefficients. Logistic regression is preferable in unequal grouping conditions. The dependent variable in logistic regression can take the value 1 with a probability of success θ, or the value 0 with probability of failure 1-θ. Logistic regression makes no assumption about the distribution of the independent variables. Thus, the variables do not have to be normally distributed, linearly related or of equal variance within each group. The relationship between the features and class labels is not a linear function in logistic regression; instead,

the logistic regression function is used, which is the logit transformation of θ,

$$\log\left[\frac{\theta(\vec{x}_i)}{1-\theta(\vec{x}_i)}\right] = \alpha + \beta_1 e_{1i} + \beta_2 e_{2i} + ... + + \beta_m e_{mi} \qquad i = 1,..,n$$

where α is the constant of the equation and β's are coefficients. α and β's are usually determined by maximum likelihood. Since logistic regression calculates the probability or success over the probability of failure, the results of the analysis are presented in the form of an odds ratio, which is defined as the ratio of the probability of occurrence of an event to the probability of the event not occurring.

Compound covariate predictor (CCP) is a weighted linear combination of expression levels for features that are univariately significant at a pre-specified significance level (1). The univariate t-statistics for comparing two classes are used as weights:

$$\theta_i = t_1 e_{1i} + t_2 e_{2i} + \cdots + t_m e_{mi} \qquad i=1,..,n$$

where t_j is the two-sample t-statistic for feature j. Once, the θ_i's are determined for each sample in the training set, the class label y of an unknown sample $\vec{x}$ is determined by comparing its corresponding θ with a threshold α, which is the midpoint of the means of θ_i's for the two classes:

$$y(\vec{x}) = \begin{cases} 1 & \text{if } \theta > \alpha \\ 0 & \text{otherwise} \end{cases} \qquad \text{where} \qquad \alpha = \frac{\mu_{\theta,y=0} + \mu_{\theta,y=1}}{2}$$

where $\mu_{\theta y=0}$ and $\mu_{\theta y=1}$ are the means of the θ_i's with class label y=0 and y=1, respectively, in the training data set.

Nearest Centroid Classifier assumes that the target classes correspond to individual (single) clusters and uses the cluster means (or centroids) to determine the class of a new sample point (2). A prototype pattern for class c_j is defined as the arithmetic mean:

$$\vec{\mu}_{c_j} = \frac{1}{n_{c_j}} \sum_{\vec{x}_i \in c_j} \vec{x}_i$$

where $\vec{x}_i$'s are the labeled training samples; n_{c_j} is the number of training samples labeled as class c_j. During operation, the class label of an unknown sample $\vec{x}$ is determined as $y(\vec{x}) = \arg\min_{c_j} d(\vec{\mu}_{c_j}, \vec{x})$, where $d(.)$ is a distance function. For a two-class problem (y=0 or y=1), the corresponding $\vec{\mu}_{c_j}$'s are denoted by $\vec{\mu}_0$ and $\vec{\mu}_1$, respectively. The class label of an unknown sample $\vec{x}$ is determined by comparing $d(\vec{\mu}_1, \vec{x})$ vs. $d(\vec{\mu}_2, \vec{x})$ (i.e., the sample will be assigned to the class with the smaller distance).

Naïve-Bayes classifier (NBC) relies on the basic statistical theory of probabilities and conditional probabilities to assign a sample to a pre-specified class. Let $c_j, j = 1, 2, 3\ldots c$ be our list of possible classes and let p_{ci} be the probability of a pattern belonging to c_i (a priori knowledge). Given a sample $\vec{x}$, the probability of it belonging to c_i is denoted by $p(c_i \mid \vec{x})$. Now, $\vec{x}$ belongs to class i, if $p(c_i \mid \vec{x}) > p(c_j \mid \vec{x}), \ \textit{for} \ j = 1,2,...c$ and $i \neq j$. Let $p(\vec{x} \mid c_i)$ be the probability of obtaining the pattern vector $\vec{x}$ in each of the possible groups given we know the pattern must belong to one of the n groups. According to Bayes's law:

$$p(c_i \mid \vec{x}) = \frac{p(\vec{x} \mid c_i) p_{c_j}}{\sum_{j=1}^{n} p(\vec{x} \mid c_j) p_{c_j}}$$

For two classes, c_1 and c_2, it can be shown that the Bayes classification rule is given by: "*if* $\frac{p(\vec{x} \mid c_1)}{p(\vec{x} \mid c_2)} > \frac{p_{c_2}}{p_{c_1}}$, *then* $\vec{x}$ *belongs to* c_1, *otherwise, assign* $\vec{x}$ *to class* c_2." In most practical situation, $p(\vec{x} \mid c_i)$ is estimated by assuming that it follows the "normal" distribution.

Weighted voting (WV) assumes each feature to be independent and contributes a weight or vote for a class. The class receiving the greatest number of votes is the predicted class. The algorithm finds decision boundaries half way between the class means. For two classes (0 and 1), each feature j casts a vote as $V_j = S_j(e_j - b_j)$, where $b_j = \frac{\mu_{j,0} + \mu_{j,1}}{2}$, $S_j = \frac{\mu_{j,0} - \mu_{j,1}}{\sigma_{j,0} + \sigma_{j,1}}$, e_j is the value of feature j, $\mu_{j,y}$ and $\sigma_{j,y}$ are the mean and standard deviation of the values for the jth feature in the class labeled with y (0 or 1). The final vote for an unknown sample $\vec{x}$ is determined as $y(\vec{x}) = sig\left(\sum_{j=1}^{m} V_j\right)$.

K-nearest neighbor (KNN) is used to predict the response of an observation using a nonparametric estimate of the response distribution of its k nearest (i.e., in predictor space) neighbors. KNN classification is based on the assumption that the characteristics of members of the same class should be similar and thus, observations located close together in covariate (statistical) space are members of the same class or at least have the same posterior distributions on their respective classes (3). Given a set of training samples $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n\}$ and an unknown sample $\vec{x}$, KNN finds k training samples closest to $\vec{x}$. Let these samples and their corresponding class labels be defined by the sets $\{\vec{x}_1^m, \vec{x}_2^m, \ldots, \text{and } \vec{x}_k^m\}$ and $\{y_1^m, y_2^m, \ldots y_k^m\}$, respectively. KNN classifies $\vec{x}$ to the class which has the greatest number of representatives in the latter set. In other words, the classification is performed by taking the majority vote among k nearest neighbors of $\vec{x}$.

Learning vector quantization (LVQ) has a first competitive layer and a second linear layer. The

competitive layer consists of competitive neurons that learn to cluster input vectors. The linear layer transforms the competitive layer's clusters into target classifications defined by the user.

Initially, random weight vectors are generated and a group of arbitrarily selected competitive neurons are assigned to each class. Then, LVQ picks an input vector $\vec{x}_i$ at random from the input space. In terms of distance between the weight vector of the neuron and $\vec{x}_i$, the closest competitive neuron is identified. The weight vector of the winner neuron is moved closer to the input vector, if its class label matches with the true class for $\vec{x}_i$. The weight vector will be moved away from the input vector, if the class assignment is incorrect. To achieve this, the competitive layer must have enough neurons and each a set of competitive neurons must be assigned to each class. Let $\vec{w}_j, j = 1,2,...,l$ be the set of weight vectors; $\vec{x}_i, i = 1,2,...,n$ represent the set of input vectors; c_{wc} denote the class that $\vec{w}_c$ is associated with; and c_i denote the class label of the input vector $\vec{x}_i$. Suppose that $\vec{w}_c$ is the closest weight vector to the input vector $\vec{x}_i$. The weight vectors are updated as follows:

If $c_i = c_{wc}$, then $\vec{w}_c(t+1) = \vec{w}_c(t) + \alpha_t[\vec{x}_i - \vec{w}_c(t)]$,
where $0 < \alpha_t < 1$
If $c_i \neq c_{wc}$, then $\vec{w}_c(t+1) = \vec{w}_c(t) - \alpha_t[\vec{x}_i - \vec{w}_c(t)]$

Note that the other weight vectors are not modified and α_n decreases monotonically as the number of iteration increases. During operation an unknown sample is assigned to the class labels of its closest weight vector.

Decision trees are rooted, usually binary trees, with simple classifiers placed at each internal node and a classification at each leaf. In order to evaluate a particular tree *T* with respect to an input $\vec{x}$, each classifier in the tree is assigned the argument $\vec{x}$. The outputs of the simple classifiers at the nodes determine a unique path from the root to a leaf of the decision tree; at each internal node, the left edge to a child is taken if the output of the function associated with that internal node is +1, and vice versa if it is -1. This path is known as the evaluation path. The value of the function $T(\vec{x})$ is the classification associated with the leaf reached by the evaluation path. Decision trees are generally learned by means of a top down growth procedure, which starts from the root node and greedily chooses a split of the data that maximizes some cost function, usually a measure of the "impurity" of the subsamples implicitly defined by the split. After choosing a split, the subsamples are then mapped to the two children nodes. This procedure is then recursively applied to the children, and the tree is grown until some stopping criterion is met. The tree is then used as a starting point for a bottom up search, performing a pruning of the tree. This eliminates nodes that are redundant or are unable to "pay for themselves" in terms of the cost function.

Random forest (4) is an ensemble of unpruned classification or regression trees, induced from bootstrap samples of the training data, using random feature selection in the tree induction process. It is a classification method based on "growing" an ensemble of decision tree classifiers. In order to classify a new object, the input is analyzed using each of the classification trees in the forest. Each tree gives a classification, "voting" for that class. The forest chooses the classification having the most votes (over all the trees in the forest). A measure of the importance of classification variables is also calculated by considering the difference between the results from original and randomly permuted versions of the data set. Prediction is made by aggregating (majority vote for classification or averaging for regression) the predictions of the ensemble. Random forest generally exhibits a substantial performance improvement over the single tree classifier such as classification and regression tree (CART). Izmirlian (5) discussed how the random forest approach can be successfully applied for in proteomics profiling study to construct a classifier and discover peak intensities most likely responsible for the separation between classes.

Fuzzy classification methods express classification rules in qualitative values that humans can easily understand. The fuzzy system has the advantage that it does not need to be retrained when using measurements obtained from a different type of microarray. Also, as pointed out by Wang *et al.* (6), it may be especially advantageous to introduce fuzzy sets in cancer classification, where frequently unlabeled tumor samples may not necessarily be clear members of one class or another. Using crisp techniques, an ambiguous sample will be assigned to one class only, an assignment that is not warranted. On the other hand, fuzzy techniques will specify to what degree the object belongs to each class. In addition, fuzzy systems allow us to handle imprecise and noisy data by dividing the range of variables into intervals, where each interval is a fuzzy set defined as "low", "medium", or "high" instead of crisp values. A fuzzy classifier may have four principal units: fuzzification, knowledge base, decision-making (inference), and defuzzification. The fuzzy system accepts a set of inputs (e_{1i}, e_{2i}, ... e_{mi}) as its information about the outside world (also referred to as crisp data). The fuzzification unit converts these actual values into fuzzy sets using membership functions to determine the degree of truth for each rule premise. The knowledge-base unit contains membership functions defining fuzzy sets and a set of fuzzy rules. A fuzzy rule has two components, an *if* part (also referred to as premise) and a *then* part (also referred to as conclusion). Such rules can be used to represent knowledge and association, which are inexact and imprecise in nature, expressed in qualitative values that humans can easily understand. For example, one might say, *"If gene x_1 is up-regulated and gene x_2 is down-regulated, then the probability of disease y is high."* The inference unit executes these fuzzy rules. The truth-value for the premise of each rule is computed and then applied to the conclusion part of each rule. When all the rules are executed, a fuzzy region will be created for the output variable *y*. With the process of defuzzification, a crisp value of the output will be generated as a solution.

Multilayer perceptrons (MLPs) consist of a number of interconnected processing elements (neurons) arranged in layers. The inputs to the network are expression levels of a sample and the network output is the predicted class label. Assuming an MLP with two hidden layers, the outputs of the neurons in the first hidden layer, second hidden layer, and output layer, respectively can be written as:

$$p_k = f\left(\sum_{j=1}^{m} w_{kj}^{(p)} e_{ji}\right), k=1,..l\ ;\ q_k = g\left(\sum_{j=1}^{l} w_{kj}^{(q)} p_j\right),\ k=1,..s;$$

and $\theta = h\left(\sum_{j=1}^{s} w_j^{(\theta)} q_j\right)$

where p_k and q_k are the outputs of the kth neuron in the first and second hidden layers, respectively. θ is the output of the network. $f(.)$, $g(.)$, and $h(.)$ are activation functions which transform the activation level of a neuron into an output signal. Typically, an activation function could be a threshold function, a sigmoid function (an S-shaped, symmetric function that is continuous and differentiable), a Gaussian function, or a linear function. l and s are the number of neurons in the first and second hidden layers, respectively. The network size (i.e., l and s) and the network weights (i.e., $w_{kj}^{(p)}$'s, $w_{kj}^{(q)}$'s. and $w_j^{(\theta)}$'s) are adjusted or trained to achieve a desired overall behavior of the network in terms of predicting the phenotype of samples in the training set.

The back-propagation algorithm (Rumelhart and McClelland, 1986) is commonly used to train MLP networks. The algorithm consists of two passes, forward and backward. During the forward pass, the input signals are applied to the network and their effect propagates through the different layers and generates the network outputs at the output layer. Note that the synaptic weights of the network are all kept fixed during the forward pass. During the backward pass, the synaptic weights are adjusted in accordance with an error correction rule. The network outputs are subtracted from the desired outputs to produce error signals. The error signals are propagated backward through the network against the direction of the synaptic connections to determine how the synaptic weights should be adjusted in order to decrease a pre-defined error function. A typical error function is the sum of squares of the error signals between the network outputs and the desired outputs. Thus, during the backward pass, the back-propagation algorithm alters the synaptic weights in the direction of the steepest gradient of the error function to reduce the error function. The forward and backward passes are repeated until a pre-specified stopping criterion is achieved or the error function is significantly reduced. A number of methods are proposed to improve the performance of the steepest gradient method described above, which is based on the first derivatives of the error function with respect to the synaptic weights. Newton's method is used to speed-up training by using second derivatives of the error function with respect to the synaptic weights. The Gauss-Newton method is designed to approach second order training speed without calculating the second derivates. The Levenberg-Marquardt learning algorithm speeds up the learning process and produces enhanced learning performance by combining the standard gradient technique with the Gauss-Newton method.

It is important to mention that neural learning is strictly about minimizing the error between the desired output and the network output. Good generalization capability of the resulting network is often needed to be able to provide reasonable approximations given previously unseen data. One of the major problems that affect network generalization is overfitting, which may be caused by an inappropriate number of nodes or when the network is overtrained for too many epochs. A commonly used approach to prevent the network from overfitting is the early stopping method, where a validation data set different from the training data set is used to estimate network's performance during training. After each epoch, the network is tested using the validation data set. When validation error increases for a specified number of epochs - indicating that the network is over-fitting the training data - the training is stopped and the network weights are set to the values found where validation error was minimized. Another method that prevents overfitting is the use of regularization (weight decay), to constrain the magnitude of the weights of the network. Regularization keeps the network weights small values to make the network less likely to overfit.

During operation, a vector of an unknown sample is presented to the MLP whose network structure (e.g. network size) and weights are predetermined via the training data set. The output of the network is used to predict the phenotype of the unknown sample. A suitable threshold α is chosen to convert the numeric network output θ into categorical data (e.g. $y = 0$ or $y = 1$).

Radial basis function (RBF) networks are special feedforward networks that have a single hidden layer. The activation functions of the neurons in the hidden layer are radial basis functions, while the neurons in the output layer have simple linear activation functions. Radial basis functions are a set of predominantly nonlinear functions such as Gaussian functions that are built up into one function. Each Gaussian function responds only to a small region of the input space where the Gaussian is centered. Hence, while an MLP network uses hyperplanes defined by weighted sums as arguments to sigmoidal functions, an RBF network uses hyperellipsoids to partition the input space in which measurements or observations are made. Thus, RBF networks find the input to output map using local approximators. The key to a successful implementation of RBF networks is to find suitable centers for the Gaussian functions. Theoretically, the centers and width of the Gaussian functions can be determined with supervised learning, for example, steepest gradient method. They can also be determined through unsupervised learning by clustering the training data points. Once the centers and width of the Gaussian functions are obtained, the weights for the connection

between the hidden layer and the output layer are easily and quickly determined using methods such as linear least squares, as the output neurons are simple linear combiners.

Probabilistic neural network (PNN) uses sums of spherical Gaussian function centered at each training vector to estimate the class probability density function. As a result, it can make a classification decision in accordance with the Bayes' strategies for decision rules and provide probability and reliability measures for each classification. A single suitable smoothing factor (i.e., a common standard deviation) is used for all the Gaussian functions. A limitation of PNN is that all training vectors must be stored and used to classify new vectors. Hence, it requires large memories. An advantage of PNN is that it can follow changing on non stationary statistics quickly and simply by either adding to or replacing old training samples with new samples as they become available.

Support vector machines (SVMs) are learning kernel-based systems that use a hypothesis space of linear functions in high dimensional feature spaces. Unlike artificial neural networks, which try to define complex functions in the input feature space, the kernel methods perform a nonlinear mapping of the complex data into high dimensional feature spaces and then use simple linear function to create linear decision boundaries. Thus, the problem of choosing network architecture is replaced here by the problem of choosing a suitable kernel for the data projection. Parameters of an SVM model are determined based on structural risk minimization to search for one target known as the optimal hyperplane.

Given a training set of instance label pairs ($\vec{x}_i$, y_i), $i = 1,\ldots,n$ where $\vec{x}_i \in R^n$ and $y \in \{1,-1\}^n$, SVMs require the solution of the following optimization problem (7, 8):

$$\min_{w,b,\xi} \frac{1}{2}\vec{w}^T\vec{w} + C\sum_{i=1}^{l}\xi_i$$

subject to $y_i(\vec{w}^T\phi(\vec{x}_i)+b) \geq 1-\xi_i, \quad \xi_i \geq 0.$

where $\vec{w}$ and b are the weight vector and bias of the hyperplane and ξ_i's are nonnegative scalar variables called slack variables that measure the deviation of a data point from the ideal condition of pattern separability.Here, training vectors $\vec{x}_i$ are mapped into a higher dimensional space by the function ϕ, where $\phi(\vec{x}_i)^T\phi(\vec{x}_j)$ is called the kernel function, denoted by $K(\vec{x}_i,\vec{x}_j)$. The most commonly used kernels are:

Linear: $K(\vec{x}_i,\vec{x}_j) = \vec{x}_i^T\vec{x}_j$;

Polynomial: $K(\vec{x}_i,\vec{x}_j) = (\gamma\vec{x}_i^T\vec{x}_j + r)^d, \gamma > 0$;

RBF: $K(\vec{x}_i,\vec{x}_j) = \exp(-\gamma\left\|\vec{x}_i - \vec{x}_j\right\|^2), \gamma > 0.$; and

Sigmoid: $K(\vec{x}_i,\vec{x}_j) = \tanh(\gamma\vec{x}_i^T\vec{x}_j + r)$

where γ , r, and d are kernel parameters.

For a given set of training samples and kernel, SVM finds a linear separating hyperplane with the maximal margin in this higher dimensional space. C > 0 is the penalty parameter of the error term.

During the operation phase, the optimal hyperplane $\vec{w}\cdot\vec{x}+b=0$ and the corresponding decision function $d(\vec{x}) = \vec{w}\cdot\vec{x}+b$ are used to determine the class labels for new samples. Here, $\vec{w}$ and b are the optimal values obtained by solving the above optimization problem.

The one-vs.-all (OVA) approach extends the functionality of SVM from binary to multi-class classification. The OVA constructs a binary classifier for each group. Thus, for a k-group classification, k binary SVMs are needed. Each binary SVM classifier creates a decision boundary that separates the group it represents from all other groups. The k binary SVM classifiers compete to categorize an unknown spectrum into their corresponding group. The SVM with the highest decision value (farthest from the decision boundary) "wins" the competition, assigning the unknown spectrum to its group.

4. FEATURE SELECTION

Biological experiments from laboratory technologies like microarray and mass spectrometry generate data with a very high number of variables (features), in general much larger than the number of samples. Feature selection can be viewed as a major bottleneck of supervised learning and data mining (9, 10). When the number of features significantly outnumbers the number of samples, it is highly desirable to discard the irrelevant features. Therefore, feature selection provides a fundamental step in the analysis of high-dimensional data (11). By selecting only a subset of features, the prediction accuracy can possibly be improved and more insight into the nature of the prediction problem can be gained.

Feature selection for classification can be formalized as a combinatorial optimization problem, finding the feature set that maximizes the quality of the hypothesis learned from these features. Combinatorial search is a computationally intensive alternative to feature ranking. To seek an optimum subset of *m* features or less, all combinations of *m* features or less are tried. The combination that yields the best classification performance (on a test set or by cross-validation) is selected. Due to its computational intractability, the feature selection problem has been tackled by means of heuristic algorithms based on various machine learning techniques (9).

One can distinguish three main approaches for feature selection: *filter, wrapper*, and *embedded* (10). The filter approach relies on general characteristics of the training data set to select some features without involving any learning algorithm. This, it provides generic selection of features, not tuned by a given learning algorithm and it is

usually fast and easy to interpret. However, it has the following limitations: (1) it selects features based on "relevance" criterion, not on their classification capability; (2) redundant features can exist; (3) features that have strong discriminating power jointly, but are weak individually are ignored.

In the wrapper method, features are selected by taking into account their contribution to the performance of a given type of classifier. Thus, the wrapper method allows selection of features based on the "usefulness" criterion. It searches for a combination of useful features from the entire set of features. It tends to find features better suited to the classifier, but it also tends to be more computationally expensive than the filter approach (12). Also, wrapper methods are prone to overfitting unless an internal cross-validation method is used during feature selection.

In the embedded approach, feature selection is part of the training procedure of a classifier. The implementation of this approach depends on the type of the classifier. For example, in ANNs and decision trees pruning method can lead to a classifier which retains the most useful features.

The performance of the three feature selection approaches depends on the application domain. The wrapper approach is favored in many works since the selection of the features is directly linked to the performance of a chosen type of classifier. When the number of features becomes very large, the filter model is usually chosen due to its computational efficiency.

In this section we briefly describe filter-based methods such as *t-statistic*, *signal-to-noise ratio* (13, 14), and *correlation*. Wrapper-based methods such as *forward addition, backward elimination*, and evolutionary and swarm-intelligence-based approaches are also discussed. Finally, embedded methods such as *recursive feature elimination* (RFE) (15), shrunken centroid, and CART are presented.

T-statistic assesses each feature, whether the means of two groups are statistically different from each other. The *t*-statistic for each feature *j* can be obtained using the following equation (16):

$$t = \frac{\mu_{j,1} - \mu_{j,0}}{\sqrt{\frac{\sigma_{j,1}^2}{n_1} + \frac{\sigma_{j,0}^2}{n_0}}}$$

where n_1 and n_0 is the number of samples in the group labeled as 1 and 0, respectively, $\mu_{j,1}$ and $\mu_{j,0}$ are the means of the classes 1 and 0 for the *j*th feature. Similarly $\sigma_{j,1}$ and $\sigma_{j,0}$ are the standard deviations for the two classes for the *j*th feature. The probability (*p*-value) that the two groups come from the same distribution is calculated; *p* varies from 0 (not likely) to 1 (certain). The higher the probability, the more likely it is that the corresponding feature does not distinguish the two groups, and that any differences are just due to random chance. The lower the probability, the more likely it is that that the feature is relevant. Since the number of features to choose from is very large, various methods are used to account for multiple testing. For example, false discovery rate (FDR) estimates the expected proportion of features that are wrongly selected as differentially expressed. Benjamin and Hochberg (17) introduced the notion of FDR and proposed a method to control FDR by adjusting the p-values. Korn *et al.* (18) proposed a non-parametric method based on a permutation test to estimate FDR.

Other non-parametric and parametric methods have been used as alternatives to the t-test. These include the Wilcoxon rank sum test and random variance t-test. The Wilcoxon rank sum test is a non-parametric method that outperforms the t-test when the data do not follow normal distribution. The random-variance t-test is an improvement over the standard t-test as it permits sharing information among features about within-class variation without assuming that all features have the same variance (19).

Signal-to-noise ratio (SNR) finds the features that will help discriminate between two groups by calculating a score which gives the highest score to those features whose expression levels differ most on average in the two groups while favoring those with small deviations in scores in the respective classes. The score for each feature *j* is calculated as:

$$S_j = \frac{\mu_{j,1} - \mu_{j,0}}{\sigma_{j,1} + \sigma_{j,0}}$$

where $\mu_{j,1}$ and $\mu_{j,0}$ are the means of the classes 1 and 0 for the *j*th feature (13, 14). Similarly $\sigma_{j,1}$ and $\sigma_{j,0}$ are the standard deviations for the two classes for the *j*th feature. Features that give the most positive values are most correlated with class 1, and that give the most negative values are most correlated with class 0. The features with the highest magnitude of S_j scores are then selected as top features.

Correlation-based method ranks features based on the Pearson correlation coefficient (*R*) or the coefficient of determination (R^2) between each feature $\vec{z}_j$ and the vector for the class label in the training data set. Features with high correlation (positive or negative) are considered to be relevant. Rather than ranking individual features, a correlation-based feature selection machine learning method proposed in (20) selects as subset of features by removing redundant features. This was accomplished by applying a heuristic algorithm that takes into account the usefulness of individual features for predicting the class along with the level of inter-correlation among them. The assumption is that good feature subsets contain features highly correlated with the class, yet uncorrelated with each other.

Sequential search methods are used in a wrapper setting. Instead of seeking optimal solution through exhaustive search methods that can be impractical for very

high-dimensional problems, sequential search methods such as forward addition and backward elimination look for sub-optimal set of features. Forward addition starts with no features and adds features based on their contribution to prediction. Backward elimination starts by including all the features for classification and then eliminating those that do not make a significant contribution to prediction. Both suffer from the so-called "nesting effect." For example, in backward elimination, once features are discarded they can not be reselected. In forward addition, once a feature is selected it cannot be discarded.

Individual discriminatory gene (IDG) and jointly discriminatory gene (JDG) methods were proposed by Xuan *et al.* (21) based on the weighted Fisher criterion (wFC). The algorithm consists of two steps: First, all features are ranked on the basis of their individual discriminatory power measured by the 1D wFC. A feature will be selected as an individually discriminatory gene (IDG) if its discriminatory power is above an empirical threshold. Second, from the IDG pool, we select jointly discriminatory gene (JDG) subsets (with various sizes) whose joint discriminatory power is the maximum among all sets of the same size. The joint discriminatory power is also determined by the multi-dimensional version of wFC. Furthermore, the JDG sets are refined by testing on a trained MLP, which ultimately determines the 'optimal' diagnostic gene subset that minimizes the generalization error. From the curve of classification rate versus JDG subsets, we pick the optimal JDG subset that corresponds to the maximal classification rate as the final inputs for the MLP. This step boosts the MLP performance, and also determines its number of inputs.

Evolutionary methods such as genetic algorithms (GAs) have been used for guiding the selection of a subset of features that leads to more accurate prediction. A commonly used approach is to combine an evolutionary algorithm with a classification algorithm. Examples include GA-KNN (22), GA-SOC (23), and GA-SVM (24). In (25), a detailed survey is presented on the use of evolutionary algorithms for classification of gene expression data.

Swarm intelligence methods such as particle swarm optimization (PSO) and ant colony optimization (ACO) have been recently proposed for feature selection (26-29). PSO and its recent variants, such as cooperative PSO as proposed in (30), require less computation time and are more resilient to the curse of dimensionality than GAs. ACO methods were inspired by the observation of ant colonies, which use pheromone trails to discover the shortest paths between food sources and their nest (31). ACO can be applied to solve difficult discrete or combinatorial optimization problems search as the traveling salesman problem, sequential ordering, and vehicle routing. Both PSO and ACO are used for feature selection in a wrapper setting where they are combined with classification algorithms. Examples include PSO-SVM, ACO-ANN, etc. In particular, ACO allows the integration of wrapper with a filter method. For example, in (26) Mutual Information Evaluation Function (MIEF) is incorporated into ACO to evaluate local features, i.e. a filter approach. Also, the algorithm is used in a wrapper setting by using ANN to evaluate the performance of the features.

SVM-recursive feature elimination (SVM-RFE) is a special purpose feature selection algorithm for SVMs. It recursively removes features based upon the absolute magnitude of the hyperplane elements $d(\vec{x}) = \vec{w} \cdot \vec{x} + b$. SVM-RFE is applied to eliminate elements of $\vec{w}$ that have small magnitude, since they do not contribute much in the classification function (15). The SVM is trained with all features; then for each feature *j*, $S_j = |w_j|$ is calculated, where w_j the value of the *j*th element of $\vec{w}$. Then, the *Sj*'s are sorted from largest to smallest value and the features corresponding to the indices that fall in the bottom 10% of the sorted list are removed. The SVM is retrained on this smaller feature set, and the procedure is repeated until a desired number of features is obtained.

Unified maximum separability analysis (UMSA) incorporates data distribution information into structural risk minimization learning algorithm of SVMs. It replaces the constant *C* in the optimization problem formulation of SVM with an individualized p_i for each training sample to constrain its maximum influence in the solution. In addition to data distribution information, p_i can also be used as a mechanism to incorporate other information such as sample quality or certainty of sample class assignment. The linear version of UMSA is applied to identify a direction along which two classes of data are best separated. This direction is represented as a linear combination of the original variables. The weight assigned to each variable in this combination measures the contribution of the variable toward the separation of the two classes of data.

Nearest shrunken centroid is a special purpose feature selection algorithm for the nearest centroid algorithm (32, 33). We illustrate the algorithm here as described in (2). The algorithm tries to shrink the class prototypes (μ_{c_j}) towards the overall mean: $\vec{\mu} = \frac{1}{n}\sum_{i=1}^{n} \vec{x}_i$. It calculates $d_j = \frac{\vec{\mu}_{c_j} - \vec{\mu}}{n_j(\vec{s})}$, where $n_j = \sqrt{\frac{1}{n_{c_j}} - \frac{1}{n}}$, $\vec{s}$ is a vector of pooled within class variances for each feature and division is done component wise. From this equation, the class centroid is calculated as $\vec{\mu}_{C_j} = \vec{\mu} + n_j(\vec{s} \cdot \vec{d}_j)$, where (.) denotes component wise multiplication. By decreasing d_j we can move the class centroid towards the overall centroid. When a component of the class centroid is equal to the corresponding component of the overall mean for all cases, the feature no longer plays a part in classification and is effectively removed. As $\vec{d}_i$ shrinks progressively more features are removed. The shrinkage is called soft thresholding that produces $\vec{d}_i'$ as follows:

$$d_j'(i) = \begin{cases} 0 & \text{if } |d_j(i)| - \delta \le 0 \\ sign(d_j(i))(|d_j(i)| - \delta) & \text{otherwise} \end{cases}$$

where $d_j(i)$ is the ith component of the vector $\vec{d}_i$. The shrunken centroid is then computed by replacing $\vec{d}_i$ with $\vec{d}_i'$ in $\vec{\mu}_{C_j} = \vec{\mu} + n_j(\vec{s} \cdot \vec{d}_j)$. After shrinking the centroids, samples are classified by the usual nearest centroid rule, but using the shrunken class centroids.

The nearest shrunken centroid method is used by the Prediction Analysis for Microarrays (PAM) software to identify subsets of genes that best characterize each class. Using the software, one can train the classifier, perform cross validation to get an idea of the value to use for thresholding, and do predictions.

Significance analysis of microarrays (SAM) is an algorithm developed by Tusher *et al.* (34) for identifying significant genes in microarray data. It assigns a score to each gene on the basis of change in gene expression relative to the standard deviation of repeated measurements. For genes with scores greater than an adjustable threshold, SAM uses permutations of the repeated measurements to estimate and control the percentage of genes identified by chance, i.e., the false discovery rate (FDR). FDR is defined in SAM as the median number of false positive genes divided by the number of significant genes.

Classification and regression tree (CART) analysis performs a binary recursive partitioning of samples. It is binary, because each group of samples, represented by a parent node in a decision tree, can only be split into two child nodes. The binary partitioning process can be applied recursively until a pre-specified criterion is reached. As illustrated in (35), CART analysis consists of four basic steps: (1) Tree building: a tree is built using recursive splitting of nodes. Each resulting node is assigned a predicted class, based on the distribution of classes in the learning data set which would occur in that node and the decision cost matrix. The assignment of a predicted class to each node occurs whether or not that node is subsequently split into child nodes. (2) Stopping of the tree building process: at this point a "maximal" tree has been produced. The tree may overfit the information contained within the training examples. (3) Pruning: this involves the process of creating a sequence of simpler and simpler trees. (4) Optimal tree selection: the tree which fits (without overfitting) the information in the training examples is selected from among the sequence of pruned trees.

5. EVALUATION OF FEATURE SELECTION AND PATTERN CLASSIFICATION METHODS

An important weakness of many machine learning-based classification algorithms is that they are not based on a probabilistic model. There is no probability level or confidence interval associated with predictions derived from using them to classify a new set of data. The confidence that an analyst can have in the accuracy of the results produced by a given classifier is based purely on its historical accuracy—how well it has predicted the desired response in other, similar circumstances. Thus, after learning is completed, a machine-learned paradigm is evaluated for its performance through previously unseen testing data set (also known as a blind validation set). The purpose of this testing is to prove the adequacy or to detect the inadequacy of features or a classifier. Inadequate performance could be attributed to insufficient or redundant features, inappropriate selection of model structure for the classifier, too few or too many model parameters, insufficient training, overtraining, error in the program code, or complexity of the underlying system such as presence of highly nonlinear relationships, noise, and systematic bias. The aim of evaluating is a classifier is to insure that it serves as a general model. A general model is one whose input-output relationships (derived from the training data set) apply equally well to new sets of data (previously unseen test data) from the same problem not included in the training set. Thus, the main goal of machine learning-based modeling is thus the generalization to new data of the relationships learned on the training set (36).

Various methods have been used to test the generalization capability of a classifier. These include the *k*-fold cross-validation, bootstrapping, and hold-out methods. In *k*-fold cross-validation, we divide the data into *k* subsets of (approximately) equal size. We train the model l times, each time leaving out one of the subsets from training, but using only the omitted subset to compute the classification error. If *k* equals the sample size, this is called "leave-one-out" cross-validation. In the leave-one-out method, one sample is selected as a validation sample and feature selection and classifier building are performed using the remaining data set. The resulting model is tested on the validation sample. The process is repeated until all samples appear in the validation set. In the hold-out method, only a single subset (also known as validation set) is used to estimate the generalization error. Thus, the hold-out method does not involve crossing. In bootstrapping, a sub-sample is randomly selected from the full training data set with replacement. Common bootstrapping methods include bagging and boosting. Bagging can be used with many classification methods and regression methods to reduce the variance associated with prediction, and thereby improve the prediction process. In bagging, many bootstrap samples are drawn from the available data, some prediction method is applied to each bootstrap sample, and then the results are combined by voting. Boosting can also be used to improve the accuracy of classification. Unlike bagging, the samples used at each step are not all drawn in the same way from the same population, but rather the incorrectly predicted cases from a given step are given increased weight during the next step. Hence, boosting uses a weighted average of results obtained from applying a prediction method to various samples.

Once the features are selected and a pattern classifier is built, the performance of the classifier in phenotype prediction is estimated via a blind validation (independent) data set. If an independent data set is not available, the cross-validation method (e.g. leave-one-out

Table 1. Confusion matrix

		Predicted	
		Positive	Negative
Actual	Positive	*TP*	*FN*
	Negative	*FP*	*TN*

approach) can be used to estimate the performance of a classifier. The most common performance measures for classifiers are a confusion matrix and a receiver operating characteristic (ROC) curve.

A confusion matrix presents information about actual and predicted classifications made by a classifier to assess the performance of the classifier. Table 1 shows the confusion matrix for a binary classifier, where *TP*, *TN*, *FP*, and *FN* denote the number of true positive, true negative, false positive, and false negative samples, respectively. A false positive is when the outcome is incorrectly classified as "positive", when it is in fact "negative". A false negative is when the outcome is incorrectly classified as negative when it is in fact positive. True positives and true negatives are correct classifications. Various performance measures are used including the following:

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP}$$

$$Positive\ \ predictive\ \ value = \frac{TP}{TP + FP}$$

$$Negative\ \ predictive\ value = \frac{TN}{TN + FN}$$

$$Overall\ \ classification\ accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

The ROC is a plot of the sensitivity of a classifier against 1-specificity for multiple decision thresholds. In many cases, a classifier has a threshold that can be adjusted to increase sensitivity at the cost of specificity or the vice versa. Each threshold setting provides a (1-specificity, sensitivity) pair and a series of such pairs can be used to plot an ROC curve. For example, in SVM the distance from the optimal hyperplane can be used a threshold to generate data for ROC plot. The area under the ROC curve (AUROC) is a reflection of how good the classifier is at distinguishing between cases and controls. The greater the AUROC, the better the classifier is.

6. RECENT APPLICATIONS IN MOLECULAR CANCER CLASSIFICATION

In this section, we summarize some of the recent studies where machine learning and statistical methods were used for selection of markers and classification of high dimensional DNA microarray and mass spectrometry data.

6.1. DNA microarray

Several studies on gene selection for the molecular classification of diseases using gene expression profiles have been reported. For example, Golub *et al.* (13) constructed the weighted voting method and used it to distinguish between two types of human acute leukemias, acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML). The data consisted of expression measurements on 6817 genes obtained from Affymetrix GeneChip. They selected 50 genes on the basis of SNR in 38 training samples and applied the weighted voting method to predict 34 new samples in the testing data set as either AML or ALL.

Khan *et al.* (37) used an ANN-classifier approach to select a subset of genes for the multiclass prediction of small round blue cell tumors (SRBCTs). The data consisted of expression measurements on 6,567 genes (2,308 genes after filtering for minimal level of expression) obtained from glass-slide cDNA microarrays. The tumors are classified as Burkitt lymphoma (BL), Ewing sarcoma (EWS), neuroblastoma (NB), or rhabdomyosarcoma (RMS). A total of 63 training samples and 25 test samples were provided, although five of the latter were not SRBCTs. The dimensionality of the microarray data was reduced by principal component analysis (PCA) and used the principal components as an input to ANN. PCA replaces redundant or highly correlated features with a smaller number of uncorrelated features capturing most of the information. A sensitivity analysis of ANN's input-output relations was applied and identified 96 genes that resulted on a test error of 0%. Tibshirani *et al.* (32) used the nearest shrunken centroid method to classify the SRBCTs data, where a smaller gene set (43 genes) achieved comparable classification performance (37).

Berrar *et al.* (38) used PNN for multiclass cancer classification using two gene expression data sets, the leukemia data set (13) and the NCI60 data set (39). They compared the performance of the PNN with two machine learning methods, a decision tree and a neural network. To evaluate the performance of the classifiers, they used a lift-based scoring system that allows a fair comparison of different models. They reported that PNN outperformed the other models. The results demonstrate the successful application of the PNN model for multiclass cancer classification.

Dudoit *et al.* (40) compared the performance of various predictors including LDA, CART, and KNN using three DNA microarray data sets: the leukemia (ALL/AML) data set of Golub *et al.* (13); the lymphoma data set of Alizadeh *et al.* (41); and the 60 cancer cell line (NCI 60) data set of Ross *et al.* (42). They concluded that the rankings of the classifiers were similar across data sets and the main conclusion, for these data sets, is that simple classifiers such as diagonal LDA and nearest neighbors perform remarkably well compared to more sophisticated methods such as aggregated classification trees.

Diaz-Uriarte and Alvarez de Andres (43) investigated the use of random forest for classification of microarray data (including multi-class problems) and proposed a new method of gene selection in classification problems based on random forest. To select genes, they

iteratively fit random forests, at each iteration building a new forest after discarding those variables (genes) with the smallest variable importance; the selected set of genes is the one that yields the smallest out-of-bag (OOB) error rate. Using simulated and nine microarray data sets they show that random forest has comparable performance to other classification methods, including DLDA, KNN, and SVM, and that the new gene selection procedure yields very small sets of genes (often smaller than alternative methods) while preserving predictive accuracy.

O'Neill *et al.* and Wei *et al.* (44, 45) narrowed the number of genes by differentiating a trained ANN to assess the relative dependence of the classification output on each active input neuron. The differentiation process involves slightly perturbing the activation of each active input neuron, one at a time, to note the specific change in the output value.

Bicciato *et al.* (46) performed gene selection and cancer classification using auto associative neural networks, a specific type of ANN trained to generate an identity association in which the network outputs approximate the inputs using nonlinear transfer functions. By inspecting the network's weight matrices, they assessed the contribution of each gene and selected specific marker genes.

To enhance the classification power of ANN-based microarray data analysis, Linder *et al.* (47) proposed a cascade of ANNs known as subsequent ANN (SANN), where the classification made by the first ANN was used as a preselection followed by a final categorization by a successive ANN. SANN was used for the multiclass classification of microarray data and obtained better classification accuracy than a classical ANN.

SVMs have demonstrated superior performance in classifying microarray data. They provide several advantages including their flexibility in choosing a similarity function, sparseness of solution, ability to handle large feature spaces, and their ability to identify outliers (48). In particular, they have been used successfully in multiclass cancer diagnosis using microarray data (49). Furey *et al.* (50) used SVMs to explore microarray data for mislabeled or questionable tissue results. In (15), SVM-RFE is applied for gene selection yielding improved classification performance.

New variants of SVMs are emerging to improve performance in microarray data classification. For example, Komura *et al.* (51) proposed multidimensional SVMs (MD-SVMs) that generate multiple orthogonal axes based on a margin between classes. This extends the classical SVM that uses a decision function in a one-dimensional space. Statnikov *et al.* (52) applied multicategory SVM (MC-SVM) that displayed less sensitivity to the curse of dimensionality than MLPs. Zhang *et al.* (53) developed a novel type of regularization in SVMs for simultaneous gene selection and cancer classification. This goal was achieved by imposing a special nonconvex penalty on the hinge loss function in the SVM model. The approach automatically eliminates redundant genes and yields a classifier with a compact set of genes.

Duan *et al.* (16) proposed a new feature selection method that uses a backward elimination procedure similar to that implemented in SVM-RFE. Unlike the SVM-RFE method, at each step, the proposed approach computes the feature ranking score from a statistical analysis of weight vectors of multiple linear SVMs trained on subsamples of the original training data. The method was tested on four gene expression data sets for cancer classification. It was concluded that this feature selection method selects better gene subsets than the original SVM-RFE and improves the classification accuracy.

Fuzzy systems are applied in the classification of tumors using microarray data in the studies presented in (54) and (55). These fuzzy systems provide short rules that are easy to interpret and exhibit classification performances that compare favorably with other methods such as logistic regression models. Wang *et al.* (6) used a fuzzy c-means (FCM) clustering to obtain a confidence measure for tumor classification. First, they used a self-organizing map (SOM), where gene expression profiles are summarized into two-dimensional SOM with optimally selected map units (estimated by stress function). Each component plane describes a gene expression structure of a tumor sample or a class, and the component plane is displayed by taking from each map the value of the component, and depicting this as a color on the grid. To estimate the number of SOM reference vectors that best fit the data distribution of a high dimensional input space, they then used a forward searching algorithm with a stress function to detect the boundaries of SOM reference vectors. Feature selection is performed on the weighted/mean component plane, by either automatic feature selection (pair-wise Fisher's linear discriminant) or manual feature selection. In the latter method, FCM clustering was applied to the optimally selected SOM and its fuzzy membership values used to construct a weighted SOM component plane of each tumor class.

Evolutionary algorithms provide global optimization methods for improved gene selection and phenotypic classification. In (56), GA is applied to search for a subset of genes whose expressions can distinguish samples from classes. The effectiveness of a subset of expressed genes is evaluated in terms of its discrimination power by SVMs. The algorithm is run multiple times using data selected by the bootstrapping method. Genes that are selected more frequently in multiple runs are used for cancer classification using SVM. This method is based on previous work by Li *et al.* (57), which combined GA with KNN for gene selection and cancer classification using microarray data.

6.2. Protein mass spectrometry

Zhu *et al.* (58) used a statistical algorithm that can select a subset of k biomarkers from the marker list that could best discriminate between the groups in a training data set via the best k-subset discriminant method with high sensitivity and specificity.

Lilien *et al.* (59) developed an algorithm called Q5 for probabilistic classification of healthy vs. disease whole serum samples using mass spectrometry. The algorithm employs PCA followed by LDA on whole spectrum surface-enhanced laser desorption/ionization time-of-flight (SELDI-TOF) mass spectrometry data, and is demonstrated on four real data sets from complete, complex SELDI spectra of human blood serum.

Levner (2) examined the performance of the nearest centroid classifier coupled with various feature selection algorithms including filter-based feature ranking methods (T-statistic, Kolmogorov-Smirnov test, and SNR), wrapper approaches (sequential forward selection and a modified version of sequential backward selection), and embedded approaches (shrunken nearest centroid and a boosting based feature selection method). In addition, the study tested dimensionality reduction approaches such as PCA coupled with LDA. Evaluation of these feature selection methods was done using stratified cross validation with an internal leave-one-out cross-validation loop for automated feature selection. For classification the nearest centroid method was used. Through five cancer MS data sets, it was shown that sequential forward selection and boosted feature selection algorithms produce the most consistent results across all data sets.

Rogers *et al.* (60) trained neural-network models based on either presence/absence of peaks or peak intensity values from SELDI spectra to distinguish samples from patients before undergoing nephrectomy for clear cell renal cell carcinoma (RCC), normal volunteers, and outpatients attending with benign diseases of the urogenital.

Wagner *et al.* (61) used matrix-assisted laser desorption/ionization time-of-flight (MALDI-TOF) spectra obtained from 41 serum samples (24 diagnosed with lung cancer and 17 health individuals). A linear SVM exhibited especially robust performance when the number of peaks was varied from four to thirteen, and when the peaks were selected from the training set alone. Experiments with the samples randomly assigned to the two classes confirmed that misclassification rates were significantly higher in such cases than those observed with the true data. Wagner *et al.* (62) also used a two-stage linear SVM-based procedure to select a small set of peaks from SELDI-TOF spectra that achieved good classification accuracy in distinguishing four groups, BPH, early (localized) cancer, late (metastasized) cancer and controls.

Vlahou *et al.* (63) used the biomarker patterns software (BPS), which is based on CART to discriminate ovarian cancer from benign diseases and healthy controls using SELDI spectra. Serum protein mass spectrum profiles from 139 patients with ovarian cancer, benign pelvic diseases, or healthy women were analyzed. A decision tree, using five protein peaks, resulted in an accuracy of 81.5% in the cross-validation analysis and 80% in a blinded set of samples in differentiating the ovarian cancer from the control groups.

Rajapakse *et al.* (64) illustrated the importance of feature selection in cancer classification by using SVM-RFE to select informative mass points from two cancer data sets, ovarian and lung cancer. They showed that SVM-RFE can select a good small subset of peaks with which the classifier achieves high prediction accuracy with much better performance than with the feature subset selected by T-statistics. They found that the best peak subset selected by SVM-RFE always have in the top ranked peaks by T-statistics while it includes some peaks that are ranked low by T-statistics. However, these peaks together give much better classification performance than the same number of most top ranked peaks by T-statistics. They concluded that selecting a small subset of peaks not only improves the efficiency of the classification algorithms, but also improves the cancer classification accuracy, even for classification algorithms like SVMs, which are capable of handling large number of input.

Li *et al.* (65) reported that the UMSA method enabled the identification of three discriminatory biomarkers that achieved 93% sensitivity and 91% specificity in detecting breast cancer patients from the non-cancer controls. Koopmann *et al.* (66) also applied UMSA to analyze serum samples from patients with and without pancreatic cancer using SELDI MS. Using a case-control study design, serum samples from 60 patients with resectable pancreatic adenocarcinoma were compared with samples from 60 age- and sex-matched patients with nonmalignant pancreatic diseases, as well as 60 age- and sex-matched healthy controls. They showed that SELDI profiling of serum is significantly better than the current standard serum biomarker CA19–9 at distinguishing patients with pancreatic cancer from those with pancreatitis and from healthy controls.

Petricoin *et al.* (23) applied a combination of GA and self-organizing clustering (GA-SOC) for variable selection that can distinguish ovarian cancer patients from healthy women. The GA-SOC, which is implemented in ProteomeQuest software, begins with a random generation of a population of many subsets of SELDI-TOF mass spectra with precise m/z candidate values. The user arbitrarily fixes the number of features (i.e., m/z values) that will create the best model. In their study, the number of features varies with the biologic state and ranges from 5 to 20. For the same data set, Lilien *et al.* (59) applied PCA for dimensionality reduction and LDA coupled with a nearest centroid classifier for classification.

Using MALDI TOF data, Wu *et al.* (67) compared features selected via the t-statistic approach with those selected by the random forest method. SVM, random forests, LDA/QDA, KNN, and bagged/boosted decision trees were subsequently used to classify the data. Using 15 and 25 feature sets, they reported that the latter approach improved the classification accuracy in distinguishing ovarian cancer patients from healthy individuals.

Adam *et al.* (68) identified the most informative peaks from SELDI-TOF MS data using ROC curves. The selected peaks were used by a decision tree algorithm to

differentiate between healthy individuals and those with prostate cancer. Qu *et al.* (69) obtained improved performance by using boosting. Specifically, they used the ROC curve method to identify relevant features. For subsequent feature selection and classification, they used decision stumps together with Ada-Boost and its variant, boosted decision stump feature selection (BDSFS) method. A key difference between the two methods is that BDSFS selects features without replacement, whereas boosted decision stumps (BDS) allows for selection of the same feature multiple times. The former method resulted in much smaller feature set than the latter. Using the same data set, Wagner *et al.* (62) applied a filter-based ANOVA F-statistic to rank the pre-selected peaks. They selected relevant features in sets of increasing size. Classification was performed with KNN, LDA/QDA, and SVM using 100-fold randomized cross-validation strategy. Linear SVM achieved the best accuracy using just eight peaks. Lilien *et al.* (59) used PCA for dimensionality reduction and LDA for classification. Petricoin *et al.* (23) also applied GA-SOM for classification of prostate cancer.

Ressom *et al.* (70, 71) applied PSO-SVM for biomarker selection and sample classification from SELDI-QqTOF and MALDI-TOF spectra in liver cancer studies, with high prediction accuracy. The algorithm combines PSO with SVM to identify the optimal features from a set of potential features. The algorithm randomly selects N number of particles initially from a list of L candidate features, where each particle consists of n features. The performance of each particle in distinguishing cases from controls is then evaluated by building an SVM classifier for each particle using the cross-validation method. The estimated classification accuracy is then used to select the most-fit particles, which contribute to the next generation of N candidate particles. Thus, on the average, each successive population of candidate particles fits better than its predecessor. This process continues until the performance of the SVM classifier converges. Recently, Ressom *et al.* (72) used a hybrid ACO-SVM algorithm that combines a wrapper-based method with a filter-based feature ranking, thereby reducing the computation time. Both PSO-SVM and ACO-SVM selected a panel of peaks from MALDI-TOF spectra that yielded over 90% sensitivity and specificity in detecting hepatocellular carcinoma (HCC) patients in a testing data set.

7. DISCUSSION

This review introduces some statistical and machine learning methods and their use in marker selection and classification of phenotype samples. As large volume and high dimensional data are being generated by the rapidly expanding use of microarray and mass spectrometric technologies, the number of reported applications of genomic and proteomic pattern classification algorithms is expected to increase. However, with increasing demand comes the need for further improvements that can make implementation of these algorithms for high dimensional data analysis more efficient. Key improvements include: (i) careful study design to minimize the effect of factors that may introduce bias to the data; (ii) enhanced computational power to handle the high dimensionality and large volume data; (iii) improved high-throughput technologies with less background noise and technical variability; (iv) enhanced quality control and protocol development/implementation; (v) improved data preprocessing methods to minimize the impact of background noise, sample degradation, and variability in sample preparation and instrument settings (v) improved visualization tools to assess data quality and interpret results; (vi) adequate data storage and retrieval systems; (vii) advances in statistical and machine learning methods to enhance their speed and make them more accessible to the user.

Careful study design is needed to make sure that a protocol is in place that enables appropriate randomization and replication are needed to avoid bias in sample collection and sample preparation (73). Ransohoff (74) indicated that bias will increasingly be recognized as the most important 'threat to validity' that must be addressed in the design, conduct and interpretation of such research. Bias can occur if the cancer and non-cancer groups are handled in systematically different ways, introducing an apparent 'signal' into one group but not the other. Such differences might be introduced at several stages, including specimen collection, handling and storage, or during data generation. In (75), Diamandis questioned why the features and classification performance vary so drastically across studies. This concern is based on the observations that different SELDI-TOF approaches combined with different machine learning techniques for pattern recognition produce highly variable results in terms of relevant features and classification accuracy. Such variation may be attributed to a large number of features relevant to the task of discriminating healthy individuals from those afflicted with cancer. Baggerly *et al.* (76) indicated the cause for inconsistent result could be the chemical/electronic noise and/or bias introduced during the acquisition of the MS spectra.

Zhang (77) noted that systematic biases from pre-analytical variability, which are attributed to samples could be collected under different protocols for different purposes, and analytical variability caused by sample preparation methods are often specific to institutions (sites). Hence, the use of specimens from multiple institutions combined with sound study is suggested as a means to address such biases. It is also indicated that the typical way of pooling multiple data sets together, followed by randomly dividing them into training and testing sets may still turn out to be overly optimistic with results unsustainable in actual "field use." With the large number of simultaneously measured variables, it is possible for a complex multivariate model to pick up from a pooled dataset the different types of systematic biases that existed in the original individual data sets. Hence, unless the number of sites is large and diverse enough to form a true representative sample of the target population, the "mix-and-split" use of multi-site samples is not recommended. An alternative and more conservative approach is to conduct independent discovery sessions using the data sets separately, followed by inter-institution validation.

Computational methods described in this review and many other supervised methods take advantage of prior knowledge in making distinctions between one type of phenotype sample and another. However, their success is highly dependent on the availability and quality of prior knowledge from previous experiments. If inaccurately labeled data are used for learning, the classification result will be impaired. Note that like other empirical models, supervised methods are only as good as the data set to which they are applied; hence, the quality of the data collected is very important. Unsupervised methods may be used for outlier screening.

Preprocessing of the raw data is of crucial importance and significantly influences the quality of the classification results. For example, the problem of detecting real peaks in a mass spectrum is a central one that deserves careful consideration of various preprocessing steps including smoothing, baseline correction, normalization, and alignment of spectra. In microarray data analysis, preprocessing steps such as image quantification, background adjustment, normalization, and in some platforms such as Affymetrix summarization of probe-level measurements play a significant role in obtaining reliable expression measurements. Other data preprocessing steps include outlier screening, handling of missing values, and elimination of features associated to known factors and covariates other than the variable we hope to predict.

It is important to realize the difference between variable selection, feature selection, and dimensionality reduction. Both variable and feature selection keep an optimum subset of variables/features and discards others. The term variable selection is often used to indicate the selection is made from the original (measured) variables. A feature selection, however, could mean that the selection is made out of features that are derived from the original variables. On the other hand, a dimensionality reduction can be made by mapping a subset or the entire set of the variables/features onto a low-dimensional space. This is commonly done in exploratory analysis, where the purpose is to improve visualization. It can be also used to reduce the complexity of a problem and to make the task of building a classifier easier. A well-known linear transformation used to reduce dimensionality is PCA, which transforms the input variables to a new set of variables (features). The new variables (also known as principal components) are computed as a linear combination of the original variables and are orthogonal to each other. PCA reduces input dimensionality by providing a subset of the principal components that captures most of the information in the original data. A classifier with the selected principal components as inputs may provide better accuracy than a classifier with a large dimension of original variables consisting of co-regulated genes. Nonlinear PCA such as kernel PCA can also be used for dimensionality reduction with the same goal as classical PCA, but adding the capability to look for nonlinear combinations (78).

Various software tools for microarray and mass spectrometry data analysis are currently available. These include Bioconductor, GenePattern, BRB Array Tools, CART, ClinProTools, Q5 algorithm, Proteinmarker Detection Software, Engene, Genesis, RankGene, ProteomeQuest, and Biomarker Pattern Software. It is expected that more tools that focus on feature selection methods will become available. While classification methods with good generalization capability exist (e.g. SVM, random forest), the selection of features in high dimensional and low-sample size data is still the object of vigorous research. Researchers have become aware of the need to cross-validate not only the process of building a classifier but also the feature selection scheme to ensure the generalization capability of the resulting feature set and the classifier. Also, better ways of estimating prediction accuracy are needed, mainly when the number of samples is limited and/or no independent samples are available. The trade-off between achieving good prediction accuracy versus the parsimoniousness of features is an object of discussion, particularly in terms of then need for simpler and cheaper diagnostic tests vis-à-vis the high sensitivity and specificity needed, if the markers or classifiers are deployed clinically.

7. ACKNOWLEDGEMENT

This work was supported in part by an NCI grant R03-CA119313 awarded to HWR.

8. REFERENCES

1. Satagopan JM and Panageas KS: A statistical perspective on gene expression data analysis. *Stat Med*, 22(3), 481-99 (2003)

2. Levner I: Feature selection and nearest centroid classification for protein mass spectrometry. *BMC Bioinformatics*, 6(1), 68 (2005)

3. Cover TM and Hart PE: Nearest neighbor pattern classification. *Transactions on Information Theory*, 13, 21-27 (1967)

4. Breiman L: Random Forest. *Machine learning*, 45, 5-32 (2001)

5. Izmirlian G: Application of the random forest classification algorithm to a SELDI-TOF proteomics study in the setting of a cancer prevention trial. *Ann N Y Acad Sci*, 1020, 154-74 (2004)

6. Wang J, Bo TH, Jonassen I, Myklebost O and Hovig E: Tumor classification and marker gene prediction by feature selection and fuzzy c-means clustering using microarray data. *BMC Bioinformatics*, 4, 60 (2003)

7. Boser BE, Guyon IM and Vapnik V: A training algorithm for optimal margin classifiers. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. Pittsburg, (1992)

8. Cortes C and Vapnik V: Support vector network. *Machine learning*, 20, 273-297 (1995)

9. Guyon I and Elisseff A: An introduction to variable and feature selection. *Machine learning*, 3(special issue on variable and feature selection), 1157-1182 (2003)

10. Marchiori E, Heegaard NHH, West-Nielsen M and Jimenez CR: Feature Selection for Classification with Proteomic Data of Mixed Quality. *Proceedings of the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB'05)*, 385-391 (2005)

11. Xing EP: Feature Selection in Microarray Analysis. Kluwer Academic Publishers, (2003)

12. Langley P: Selection of relevant features in machine learning. *Proceedings of the AAAI Fall Symposium on Relevance*, 140 -144. (1994)

13. Golub TR, Slonim DK, Tamayo P, Huard C, Gaasenbeek M, Mesirov JP, Coller H, Loh ML, Downing JR, Caligiuri MA, Bloomfield CD and Lander ES: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science*, 286(5439), 531-7 (1999)

14. Slonim D, Tamayo P, Mesirov JP, Golub TR and Lander ES: Class prediction and discovery using gene expression data. *Proceedings of the 4th Annual International Conference on Computational Molecular Biology (RECOMB), Universal Academy Press*, 263-272 (2000)

15. Guyon I, Weston J, Barnhill S and Vapnik V: Gene Selection for cancer classification using support vector machines. *Machine learning*, 46, 389-422 (2002)

16. Duan KB, Rajapakse JC, Wang H and Azuaje F: Multiple SVM-RFE for gene selection in cancer classification with expression data. *IEEE Trans Nanobioscience*, 4(3), 228-34 (2005)

17. Benjamini Y and Hochberg Y: Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, 57(Series B), 289-300 (1995)

18. Korn EL, Troendle JF, L.M. M and Simon R: Controlling the number of false discoveries: application to high-dimensional genomic data. *Journal of Statistical Planning and Inference*, 124, 379-398 (2004)

19. Wright GW and Simon RM: A random variance model for detection of differential gene expression in small microarray experiments. *Bioinformatics*, 19(18), 2448-55 (2003)

20. Hall MA: Correlation-based Feature Selection for Machine Learning In: *Department of Computer Science*. The University of Waikato, Hamilton, NewZealand (1999)

21. Xuan J, Dong Y, Khan J, Hoffman EP, Clarke R and Wang Y: Robust gene selection by weighted Fischer criterion for multiclass prediction in gene expression profiling. *Proceedings of the 17th International Conference on Pattern Recognition, Cambridge, UK*, 291-294 (2004)

22. Jirapech-Umpai T and Aitken S: Feature selection and classification for microarray data analysis: evolutionary methods for identifying predictive genes. *BMC Bioinformatics*, 6(1), 148 (2005)

23. Petricoin EF, Ardekani AM, Hitt BA, Levine PJ, Fusaro VA, Steinberg SM, Mills GB, Simone C, Fishman DA, Kohn EC and Liotta LA: Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, 359(9306), 572-7 (2002)

24. Huerta EB, Duval B and Hao JK: A hybrid GA/SVM approach for gene selection and classification of microarray data. Springer-Verlag, (2006)

25. Wahde M and Szallasi Z: A survey of methods for classification of gene expression data using evolutionary algorithms. *Expert Rev Mol Diagn*, 6(1), 101-10 (2006)

26. Al-Ani A: Feature Subset Selection Using Ant Colony optimization. *International Journal of Computational Intelligence*, 2(1), 53-58 (2005)

27. Ressom H, Varghese RS, Saha D, Orvisky E, Goldman L, Petricoin EF, Conrads TP, Veenstra TD, Abdel-Hamid M, Loffredo CA and Goldman R: Particle swarm optimization for analysis of mass spectral serum profiles. *Proceedings of Genetic and Evolutionary Computation Conference (GECCO)*, 1 431-438 (2005)

28. Ressom HW, Varghese RS, Orvisky E, Drake SK, Hortin GL, Abdel-Hamid A, Loffredo CA and Goldman R: Ant Colony Optimization for Biomarker Identification from MALDI-TOF Mass Spectra. *Proceedings of the 28th IEEE Engineering in Medicine and Biology Society Annual International Conference, New York City, NY*, 4560-4563 (2006)

29. Ressom HW, Varghese RS, Orvisky E, Drake SK, Hortin GL, Abdel-Hamid A, Loffredo CA and Goldman R: Biomarker identification and rule extraction from mass spectral serum profiles. *Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, Toronto, ON* 164-170 (2006)

30. van den Bergh F and Engelbrecht AP: A cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3), 225-239 (2004)

31. Dorigo M, Di Caro G and Gambardella LM: Ant algorithms for discrete optimization. *Artif Life*, 5(2), 137-72 (1999)

32. Tibshirani R, Hastie T, Narasimhan B and Chu G: Class prediction by nearest shrunken centroids with applications

to dna microarays. *Stastistical Science*, 18(1), 104-117 (2003)

33. Tibshirani R, Hastie T, Narasimhan B and Chu G: Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc Natl Acad Sci U S A*, 99(10), 6567-72 (2002)

34. Tusher VG, Tibshirani R and Chu G: Significance analysis of microarrays applied to the ionizing radiation response. *Proc Natl Acad Sci U S A*, 98(9), 5116-21 (2001)

35. Lewis R: An Introduction to Classification and Regression Tree (CART) Analysis. *Presented at the 2000 Annual Meeting of the Society for Academic Emergency Medicine in SanFrancisco, California* (2000)

36. Wang Z, Wang Y, Xuan J, Dong Y, Bakay M, Feng Y, Clarke R and Hoffman EP: Optimized multilayer perceptrons for molecular classification and diagnosis using genomic data. *Bioinformatics*, 22(6), 755-61 (2006)

37. Khan J, Wei JS, Ringner M, Saal LH, Ladanyi M, Westermann F, Berthold F, Schwab M, Antonescu CR, Peterson C and Meltzer PS: Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nat Med*, 7(6), 673-9 (2001)

38. Berrar DP, Downes CS and Dubitzky W: Multiclass cancer classification using gene expression profiling and probabilistic neural networks. *Pac Symp Biocomput*, 5-16 (2003)

39. Irizarry RA, Hobbs B, Collin F, Beazer-Barclay YD, Antonellis KJ, Scherf U and Speed TP: Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, 4(2), 249-64 (2003)

40. Dudoit S, Fridlyand J and Speed TP: Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, 97, 77-87 (2002)

41. Hastie T, Tibshirani R, Eisen MB, Alizadeh A, Levy R, Staudt L, Chan WC, Botstein D and Brown P: 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biol*, 1(2), RESEARCH0003 (2000)

42. Ross DT, Scherf U, Eisen MB, Perou CM, Rees C, Spellman P, Iyer V, Jeffrey SS, Van de Rijn M, Waltham M, Pergamenschikov A, Lee JC, Lashkari D, Shalon D, Myers TG, Weinstein JN, Botstein D and Brown PO: Systematic variation in gene expression patterns in human cancer cell lines. *Nat Genet*, 24(3), 227-35 (2000)

43. Diaz-Uriarte R and Alvarez de Andres S: Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7, 3 (2006)

44. O'Neill MC and Song L: Neural network analysis of lymphoma microarray data: prognosis and diagnosis near-perfect. *BMC Bioinformatics*, 4, 13 (2003)

45. Wei JS, Greer BT, Westermann F, Steinberg SM, Son CG, Chen QR, Whiteford CC, Bilke S, Krasnoselsky AL, Cenacchi N, Catchpoole D, Berthold F, Schwab M and Khan J: Prediction of clinical outcome using gene expression profiling and artificial neural networks for patients with neuroblastoma. *Cancer Res*, 64(19), 6883-91 (2004)

46. Bicciato S, Pandin M, Didone G and Di Bello C: Pattern identification and classification in gene expression data using an autoassociative neural network model. *Biotechnol Bioeng*, 81(5), 594-606 (2003)

47. Linder R, Dew D, Sudhoff H, Theegarten D, Remberger K, Poppl SJ and Wagner M: The 'subsequent artificial neural network' (SANN) approach might bring more classificatory power to ANN-based DNA microarray analyses. *Bioinformatics*, 20(18), 3544-52 (2004)

48. Brown MP, Grundy WN, Lin D, Cristianini N, Sugnet CW, Furey TS, Ares M, Jr. and Haussler D: Knowledge-based analysis of microarray gene expression data by using support vector machines. *Proc Natl Acad Sci U S A*, 97(1), 262-7 (2000)

49. Ramaswamy S, Tamayo P, Rifkin R, Mukherjee S, Yeang CH, Angelo M, Ladd C, Reich M, Latulippe E, Mesirov JP, Poggio T, Gerald W, Loda M, Lander ES and Golub TR: Multiclass cancer diagnosis using tumor gene expression signatures. *Proc Natl Acad Sci U S A*, 98(26), 15149-54 (2001)

50. Furey TS, Cristianini N, Duffy N, Bednarski DW, Schummer M and Haussler D: Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10), 906-14 (2000)

51. Komura D, Nakamura H, Tsutsumi S, Aburatani H and Ihara S: Multidimensional support vector machines for visualization of gene expression data. *Bioinformatics*, 21(4), 439-44 (2005)

52. Statnikov A, Aliferis CF, Tsamardinos I, Hardin D and Levy S: A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*, 21(5), 631-43 (2005)

53. Zhang HH, Ahn J, Lin X and Park C: Gene selection using support vector machines with nonconvex penalty. Institute of Statistics Mimeo (2005)

54. Ohno-Machado L, Vinterbo SA and Weber G: Classification of gene expression data using fuzzy logic. *Journal of Intelligent and Fuzzy System*, 12(1), 19-24 (2002)

55. Vinterbo SA, Kim EY and Ohno-Machado L: Small, fuzzy and interpretable gene expression based classifiers. *Bioinformatics*, 21(9), 1964-70 (2005)

56. Chen X-W: Gene Selection for Cancer Classification Using Bootstrapped Genetic Algorithms and Support Vector Machines. *Proceedings of the 2nd IEEE Computer Society Bioinformatics Conference*, 504-504 (2003)

57. Li L, Weinberg CR, Darden TA and Pedersen LG: Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. *Bioinformatics*, 17(12), 1131-42 (2001)

58. Zhu W, Wang X, Ma Y, Rao M, Glimm J and Kovach JS: Detection of cancer-specific markers amid massive mass spectral data. *Proc Natl Acad Sci U S A*, 100(25), 14666-71 (2003)

59. Lilien RH, Farid H and Donald BR: Probabilistic disease classification of expression-dependent proteomic data from mass spectrometry of human serum. *J Comput Biol*, 10(6), 925-46 (2003)

60. Rogers MA, Clarke P, Noble J, Munro NP, Paul A, Selby PJ and Banks RE: Proteomic profiling of urinary proteins in renal cancer by surface enhanced laser desorption ionization and neural-network analysis: identification of key issues affecting potential clinical utility. *Cancer Res*, 63(20), 6971-83 (2003)

61. Wagner M, Naik D and Pothen A: Protocols for disease classification from mass spectrometry data. *Proteomics*, 3(9), 1692-8 (2003)

62. Wagner M, Naik DN, Pothen A, Kasukurti S, Devineni RR, Adam BL, Semmes OJ and Wright GL, Jr.: Computational protein biomarker prediction: a case study for prostate cancer. *BMC Bioinformatics*, 5(1), 26 (2004)

63. Vlahou A, Schorge JO, Gregory BW and Coleman RL: Diagnosis of Ovarian Cancer Using Decision Tree Classification of Mass Spectral Data. *J Biomed Biotechnol*, 2003(5), 308-314 (2003)

64. Rajapakse JC, Duan KB and Yeo WK: Proteomic cancer classification with mass spectrometry data. *Am J Pharmacogenomics*, 5(5), 281-92 (2005)

65. Li J, Zhang Z, Rosenzweig J, Wang YY and Chan DW: Proteomics and bioinformatics approaches for identification of serum biomarkers to detect breast cancer. *Clin Chem*, 48(8), 1296-304 (2002)

66. Koopmann J, Zhang Z, White N, Rosenzweig J, Fedarko N, Jagannath S, Canto MI, Yeo CJ, Chan DW and Goggins M: Serum diagnosis of pancreatic adenocarcinoma using surface-enhanced laser desorption and ionization mass spectrometry. *Clin Cancer Res*, 10(3), 860-8 (2004)

67. Wu B, Abbott T, Fishman D, McMurray W, Mor G, Stone K, Ward D, Williams K and Zhao H: Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data. *Bioinformatics*, 19(13), 1636-43 (2003)

68. Adam BL, Qu Y, Davis JW, Ward MD, Clements MA, Cazares LH, Semmes OJ, Schellhammer PF, Yasui Y, Feng Z and Wright GL, Jr.: Serum protein fingerprinting coupled with a pattern-matching algorithm distinguishes prostate cancer from benign prostate hyperplasia and healthy men. *Cancer Res*, 62(13), 3609-14 (2002)

69. Qu Y, Adam BL, Yasui Y, Ward MD, Cazares LH, Schellhammer PF, Feng Z, Semmes OJ and Wright GL, Jr.: Boosted decision tree analysis of surface-enhanced laser desorption/ionization mass spectral serum profiles discriminates prostate cancer from noncancer patients. *Clin Chem*, 48(10), 1835-43 (2002)

70. Ressom HW, Varghese RS, Abdel-Hamid M, Abdel-Latif Eissa S, Saha D, Goldman L, Petricoin EF, Conrads TP, Veenstra TD, Loffredo CA and Goldman R: Analysis of mass spectral serum profiles for biomarker selection. *Bioinformatics* 21(21), 4039-4045 (2005)

71. Ressom HW, Varghese RS, Orvisky E, Drake SK, Hortin GL, Abdel-Hamid M, Loffredo CA and Goldman R: Analysis of MALDI-TOF serum profiles for biomarker selection and sample classification. *Proceedings of the 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, La Jolla, CA*, 378-384 (2005)

72. Ressom HW, Varghese RS, Drake SK, Hortin GL, Abdel-Hamid M, Loffredo CA and Goldman R: Peak selection from MALDI-TOF mass spectra using ant colony optimization. *Bioinformatics*, 23(5), 619-26 (2007)

73. Zhang Z and Chan DW: Cancer proteomics: in pursuit of "true" biomarker discovery. *Cancer Epidemiol Biomarkers Prev*, 14(10), 2283-6 (2005)

74. Ransohoff DF: Bias as a threat to the validity of cancer molecular-marker research. *Nat Rev Cancer*, 5(2), 142-9 (2005)

75. Diamandis EP: Mass spectrometry as a diagnostic and a cancer biomarker discovery tool: opportunities and potential limitations. *Mol Cell Proteomics*, 3(4), 367-78 (2004)

76. Baggerly KA, Morris JS and Coombes KR: Reproducibility of SELDI-TOF protein patterns in serum: comparing datasets from different experiments. *Bioinformatics*, 20(5), 777-85 (2004)

77. Zhang Z: Bioinformatics tools for differential analysis of proteomic expression profiling data from clinical samples. Taylor and Francis CRC Press, (2005)

78. Pochet N, De Smet F, Suykens JA and De Moor BL: Systematic benchmarking of microarray data classification: assessing the role of non-linearity and dimensionality reduction. *Bioinformatics*, 20(17), 3185-95 (2004)

Abbreviations: ACO: ant colony optimization; ALL: acute lymphoblastic leukemia; AML: acute myeloid leukemia; ANN: artificial neural network; ANOVA: analysis of variance; AUROC: area under ROC; BDS: boosted decision stumps; BDSFS: boosted decision stump feature selection; BL: Burkitt lymphoma; BPH: benign prostatic hyperplasia; CART: classification and regression tree; CCP: compound covariate predictor; cDNA: complementary DNA; DLDA: diagonal LDA; DNA: deoxyribonucleic acid; EWS: Ewing sarcoma; FCM: fuzzy c-means; FDR: false discovery rate; FN: false negative; FP: false positive; GA: genetic algorithm; HCC: hepatocellular carcinoma; IDG: individual discriminatory gene; JDG: jointly discriminatory gene; KNN: k- nearest neighbor; LDA: linear discriminant analysis; LVQ: learning vector quantization; MALDI-TOF: matrix assisted laser desorption ionization time-of-flight; MC-SVM: multicategory SVM; MD-SVM: multidimensional SVM; MIEF: mutual information evaluation function; MLP: multi-layer perceptron; MS: mass spectrometry; NB: neuroblastoma; NBC: Naïve-Bayes classifier; NCI : National cancer institute; OOB: out-of-bag; OVA: one-vs.-all; PCA: principal component analysis; PNN: probabilistic neural network; PSO: particle swarm optimization; QDA: quadratic discriminant analysis; RBF: radial basis function; RCC: renal cell carcinoma; RFE: recursive feature elimination; RMS: rhabdomyosarcoma; ROC: receiver operating characteristic; SANN: subsequent ANN; SELDI-QqTOF: surface-enhanced laser desorption/ionization-quadrupole-time-of-flight; SELDI-TOF: surface-enhanced laser desorption/ionization time-of-flight; SNR: signal-to-noise ratio; SOC: self-organizing clustering; SOM: self-organizing map; SRBCT: small round blue cell tumor; SVM: support vector machine; TN: true negative; TP: true positive; UMSA: unified maximum separability analysis; wFC: weighted Fisher criterion.

Key Words: Feature Selection, Classification, Gene Expression, Microarray, Mass Spectrometry, Review

Send correspondence to: Dr Habtom W. Ressom, 4000 Reservoir Road NW, Room 174, Building D, Washington D.C 20057, Tel: 202-687-2283, Fax: 202-687-0227, E-mail: hwr@georgetown.edu

http://www.bioscience.org/current/vol13.htm