J Sign Process Syst (2010) 61:3–20
DOI 10.1007/s11265-008-0273-8

Feature Selection for Genomic Signal Processing: Unsupervised, Supervised, and Self-Supervised Scenarios

S. Y. Kung · Yuhui Luo · Man-Wai Mak

Received: 2 March 2008 / Revised: 15 July 2008 / Accepted: 2 September 2008 / Published online: 9 October 2008

Abstract An effective data mining system lies in the representation of pattern vectors. For many bioinformatic applications, data are represented as vectors of extremely high dimension. This motivates the research on feature selection. In the literature, there are plenty of reports on feature selection methods. In terms of training data types, they are divided into the unsupervised and supervised categories. In terms of selection methods, they fall into filter and wrapper categories. This paper will provide a brief overview on the state-of-the-arts feature selection methods on all these categories. Sample applications of these methods for genomic signal processing will be highlighted. This paper also describes a notion of self-supervision. A special method called vector index adaptive SVM (VIA-SVM) is described for selecting features under the self-supervision scenario. Furthermore, the paper makes use of a more powerful symmetric doubly supervised formulation, for which VIA-SVM is particularly useful. Based on several subcellular localization experiments, and microarray time course experiments, the VIA-SVM algorithm when combined with some filter-type metrics appears to deliver a substantial dimension reduction (one-order of magnitude) with only little degradation on accuracy.

Keywords Feature selection · Genomics · Unsupervised · Supervised · Self-supervised · Microarray · Sequence · Filter · Wrapper

1 Introduction: Why Feature Selection?

In genomic applications, features usually correspond to genes, proteins (sequences), or signal motifs. Let N denote the number of training data samples, M the original feature dimension, the full raw feature can be expressed as a set of M-dimensional vectors:

$$\mathbf{x}(t) = \left[x_1(t), x_2(t), \ldots, x_M(t)\right]^T, \quad t = 1, \ldots, N.$$

The subset feature can be denoted as an m-dimensional vector process

$$\mathbf{y}(t) = \left[y_1(t), y_2(t), \cdots, y_m(t)\right]^T \tag{1}$$

$$= \left[x_{s_1}(t), x_{s_2}(t), \cdots, x_{s_m}(t)\right]^T \tag{2}$$

where $m \leq M$ and s_i stands for index of a selected feature.

For many genomic applications, the feature dimension can be extremely high. For example, the feature dimension of gene expression data is often in the order of thousands. This motivates exploration into feature

Based on SY Kung's Keynote Paper, Proceedings, IEEE Workshop on Machine Learning for Signal Processing, Thessaloniki, Greece, August 27–29, 2007.

The research was conducted in part while S.Y. Kung was on leave with the National Chung-Hsing University as a Chair Professor.

S. Y. Kung · Y. Luo (✉)
Princeton University, Princeton, NJ, USA
e-mail: yuhuiluo@princeton.edu

S. Y. Kung
National Chung Hsing University, 250 Kuo Kuang Rd., Taichung 402, Taiwan, Republic of China

M.-W. Mak
Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong SAR

selection and representation, both aiming at reducing the feature dimensionality to facilitate the training and prediction of genomic data. The challenge lies in how to reduce feature dimension while conceding minimum sacrifice on accuracy.

The traditional dimension reduction involves projection which maps high-dimensional feature spaces into low dimension by finding some optimal linear combinations of the features. As an alternative, dimension reduction may also be accomplished by feature selection which involves retaining selectively the most useful features.

1.1 Computational Perspectives: Reduction of Dimensionality

The extreme dimension of features motivates (if not necessitates) feature selection process because high dimensionality in feature spaces increases uncertainty in classification. Two serious adverse effects are:

- **Data over-fitting.** It is well known that data over-fitting may happen when the vector dimension is relatively too large when compared with the size of training data. An excessive dimensionality could severely jeopardize the generalization capability due to over-fitting and unpredictability of the numerical behavior. Feature reduction is an effective way to alleviate the overtraining problem.
- **Suboptimal search.** Relatively, the computational resources available for genomic processing are never sufficient, given the astronomical amounts of genomic data needing to be processed. High dimensionality in feature spaces increases uncertainty in the numerical behaviors. As a result, a computational process often converges to a solution far inferior to the true optimum, which may compromise the prediction accuracy.
- **Computation loads.** Such an extreme dimensionality has a serious and adverse effect on the computation loads. First, high dimensionality in feature spaces increases the computational cost in both the (1) learning phase and (2) prediction phase.

Here, let us use a subcellular localization example as an evidence to support such a non-monotonic performance curve and highlight the importance of feature selection.

Example 1 Subcellular Localization. Profile alignment SVMs [1] are applied to predict the subcellular location of proteins in an eukaryotic protein dataset provided by Reinhardt and Hubbard [2]. The dataset comprises 2427 annotated sequences extracted from SWISSPROT 33.0, which amounts to 684 cytoplasm, 325 extracellular, 321 mitochondrial, and 1097 nuclear proteins. 5-Fold cross validation was used to obtain the prediction accuracy. The accuracy and testing time for different number of features selected by a Fisher-based method [3] are shown in Fig. 1. This example offers an evidence of the non-monotonic performance property based on real genomic data.

1.2 Biological Perspectives: Feature Selection

There are genomic applications where feature extraction methods that rely on combinations of features do not apply. In this case, feature selection has a special appeal. Some are exemplified as follows.

1. **Presence of co-expressed genes**: The presence of co-expressed genes implies that there exists abundant redundancy among the genes. Such redundancy plays a vital role and has a great influence on how to select features as well as how many to select.
2. **Plenty of irrelevant genes**: From the biological view point, only a small portion of genes are strongly indicative of a targeted disease. The remaining "housekeeping" genes would not contribute relevant information. Moreover, their participation in

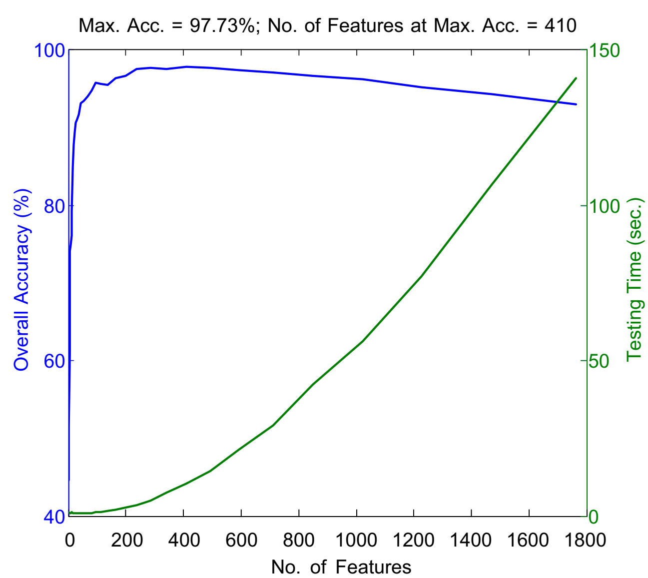

Figure 1 Real data supporting the Monotonic increasing property. *Upper curve*: Performance reach a peak by selecting an optimal size instead of the full set of the features available. *Lower curve*: the computational time goes up (more than linear rate) as the number of features increases.

the training and prediction phases could adversely affect the classification performance.

3. **Identification of biomarkers** The selective genes may pave a way to identify those genes most relevant to a targeted disease, known as bio-markers. Concentrating on such a compact subset of selected genes would facilitate a better interpretation and understanding of the cause-effect pertaining to the disease. A plausible application example is selection of critical genes or sequences for discriminating cancer/non-cancer cases. Because the size of the selected genes is small, it is more affordable to go through more advanced dry or wet experiments for further validation.

2 Genomic Applications: Overview

2.1 Feature Selection for Microarray Data

In genomic applications, to facilitate analysis, interpretation, and classification, it is advantageous to convert biological data into an $M \times N$ matrix $\mathbf{Z} \in R^{M\times N}$:

$$\mathbf{Z} = \begin{bmatrix} \mathbf{z}_{11} & \mathbf{z}_{12} & \cdots & \mathbf{z}_{1N} \\ \mathbf{z}_{21} & \mathbf{z}_{22} & \cdots & \mathbf{z}_{2N} \\ \cdots & \cdots & \cdots & \cdots \\ \mathbf{z}_{M1} & \mathbf{z}_{M2} & \cdots & \mathbf{z}_{MN} \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1^T \\ \mathbf{y}_2^T \\ \vdots \\ \mathbf{y}_M^T \end{bmatrix}$$
$$= \left[\mathbf{x}_1 \; \mathbf{x}_2 \cdots \mathbf{x}_N \right]. \quad (3)$$

In this paper, depending on the data types and classification tasks, features can be selected either along the x-direction of (i.e., selecting columns) or along the y-direction (i.e., selecting rows).

1. **Regular supervised scenario:** For clinical applications of microarrays, M in Eq. 3 is the number of genes and N is the number of samples. In other words, the M-dimensional column vector $\mathbf{x}_j$ represents j-th clinical sample. Typically, the number of genes is significantly larger than the number of samples. This could cause the curse of dimensionality problem if the goal is to classify the samples. One approach to alleviating this problem is to select m out of M features along the y-direction of the matrix, where $m \ll M$ (see Fig. 2a). The feature selection problem now becomes:

   Given differential signal levels over the x-direction (e.g. over different classes of samples/conditions), find the relevant features along the y-direction (e.g. critical genes).

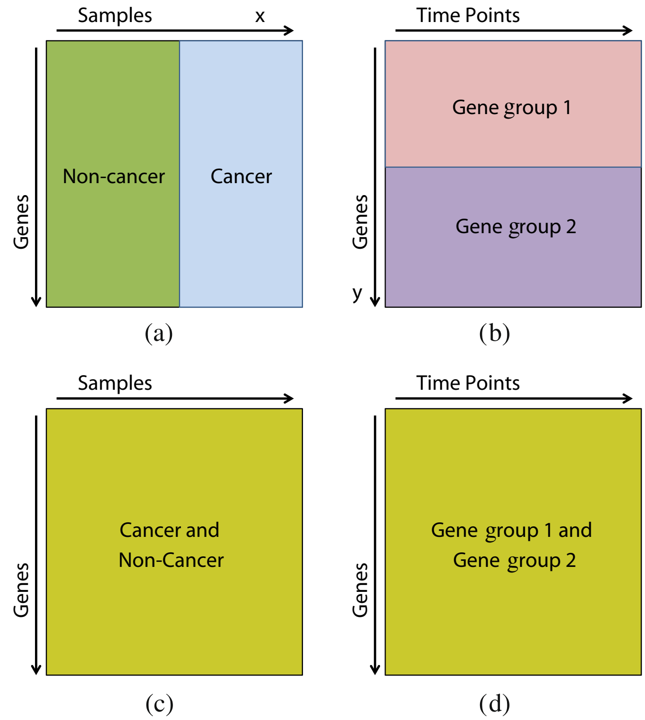

Figure 2 Different prior knowledge of **a** regular supervised, **b** self-supervised, **c** regular unsupervised, and **d** self-unsupervised data. In the regular training data, the assignment of class labels (either known a priori in the supervised case or through class discovery in the unsupervised case) are done to the column vectors (in the x-direction). In the self-supervised or self-unsupervised situation, the assignment of class labels (either known a priori or through class discovery) are done to the features (in the y-direction), instead of the column vectors.

2. **Self-supervised (SS) scenario:** Suppose the expression level of M genes are measured at N time points. Given a gene expression matrix $\mathbf{Z}$ and the class labels of the M genes in the matrix, a gene classifier can be created. However, among the genes in $\mathbf{Z}$, many of them may show little variation across the time points. Therefore, it is important not to use these irrelevant genes for training the classifier. In this scenario, the class labels are known in the y-direction (e.g., ribosomal vs. non-ribosomal genes), but not in the x-direction, as shown in Fig. 2b. This is called self-supervised learning. The feature selection problem now becomes:

   Given differential signal levels over the y-direction (e.g. over different classes of genes), find the relevant features along the y-direction (e.g. critical genes).

3. **Regular Unsupervised scenario:** Like its supervised counterpart, we have N samples in the x-direction and M genes (or features) in the y-direction. However, in this case, the class labels for the samples are not known ahead of time, and we wish to perform class discovery and uncover biologically significant groupings of the samples into similar classes (e.g. different types of cancers). (See Fig. 2c)
4. **Self-Unsupervised scenario:** Similar to its supervised counterpart, we once again have time-course microarray data, with the time points in the x-direction and the genes (or features) in the y-direction (see Fig. 2d). We don't have the labels of the genes ahead of time, and we want to group the genes which display similar time-course expression levels together and attempt to find the biological significance of these groupings (e.g. genes that are active in the same cell cycle).

2.2 Feature Selection for Sequence Data

Because the elements of biological sequences are represented by alphabets (20 amino acids for proteins and 4 nucleotides for DNA) instead of numerical values and most machine learning tools can only process data in vectorial forms, it is necessary to convert biological sequences to vectors for classification—a process known as vectorization.

The most prominent approach to vectorization is the one that uses k-mers and motif counts. Its advantage lies in the fact that there is now a matrix data representation. In particular, the approach converts a set of M sequences into an $M \times N$ matrix, where N is the number of k-mers patterns or motif counts.

For this paper, we will reserve the y-direction of the matrix $\mathbf{Z}$ for the sequence indices, and the x-direction of the matrix represents different k-mer or motif patterns. Depending on the direction along which the features are selected, we have two kinds of feature selection for sequence data. The first is selection of the most relevant k-mers along the x-direction, and the other is selection of the most relevant sequences along the y-direction. This is to be elaborated below:

1. *Selection of k-mers and motif counts*. Each feature in a vector represents the frequency of occurrences of a particular alphabet combinations in that sequence [4, 5]. The features are derived from the sequences independently, i.e., for each sequence, its corresponding vector is derived from the contents of the sequence only. For large k or long motifs, the feature dimension will be too large for reliable classification and therefore feature selection is imperative. For this type sequence representation, selection is along the x-direction of the data matrix $\mathbf{Z}$ in Eq. 3, and only n out of N features will be selected.[1]
2. *Selection of Sequences*. In some applications, we are given M sequences together with their class labels. The goal is to train a sequence classifier to predict the classes of query sequences. Among these M training sequences, some of them may be redundant and some may not be relevant to the classification task. Therefore, it is important to weed them out. Selection is therefore along the y-direction of $\mathbf{Z}$, and m out of M sequences are to be selected.

Research has shown that feature selection is an important step in many biological applications of sequence analysis, including enzyme classification [6], motif finding [6, 7], remote homology detection [5, 7], subcellular localization [8–10], and protein fold prediction [11].

In the context of sequence classification, reducing the dimensionality along the x-direction of $\mathbf{Z}$ or selecting relevant sequences along the y-direction are strategies to achieve two goals.

1. *Improve classification performance*. From the x-direction perspective, removing irrelevant k-mers patterns can help the classifier to capture the most discriminative characteristics of the sequences for classification. From the y-direction perspective, removal of redundant sequences can help the training of classifiers because it avoids the redundant sequences from dominating the decision boundaries.
2. *Reduce retrieval time*. From the x-direction perspective, reducing the number of k-mers patterns and motif counts means reducing the number of inputs to the classifier and reducing the time to create a feature vector from a query sequence. From the y-direction perspective, computation saving occurs when the pairwise approach (see Section 6.3.4) is adopted because constraining the training sequences to a small but relevant set means reducing the number of pairwise alignments, which represents a significant computation saving.

Note that for sequence data, feature selection along the x-direction produces some relevant motifs as a

[1] For such a huge dimensionality, a preliminary Signal-to-Noise ratio (SNR)-based filtering method can be applied to weed out those k-mers patterns (i.e. columns) that are below certain low threshold.

by-product because for k-mers and motif counts, the selected features represent the biologically relevant k-mers and motif patterns.

3 Criteria and Approaches

3.1 Feature Selection Criteria

Two criteria are often considered in feature selection:

1. **Relevance and Signal Strength:** The ranking of an *individual* feature hinges upon its relevance. For example, in microarray data, a particular gene that exhibits very distinctive responses across all the samples or experimental conditions is considered highly relevant. For unsupervised learning, the variance can be used as a measurement of relevance. For supervised learning, a feature's SNR across classes can be used as a score for its relevance. Typically, the metric is defined as:

   $$SNR = \frac{signal}{noise},$$

   where *signal* and *noise* represent inter-class distinction and intra-class perturbation respectively.
2. **Redundancy:** Optimal feature selection depends not only on the individual ranking scores but also *inter-feature relationships*, i.e. the mutual redundancy among features. Data redundancy is prevalent in genomic data. These can include co-expressed genes which exhibit very similar behaviors across different classes. If we wish to obtain computational savings, then redundant genes can be removed from our feature set. Another common scenario includes instances where data entries are replicated across different databases. If any of these entries contain errors, the problem of noisy/bad data is compounded even further by the repetition of error. Thus, removing redundancies can alleviate the problem of erroneous annotations.

3.2 Individual vs. Group Ranking Approaches

Suppose we have M features available. Feature selection techniques which only take into account *individual* feature rankings are computationally efficient, on the order of $O(M)$. However, they fail to take into account the inter-feature redundancy that abound in genomic data. For example, it is very possible that the two highest-rank individual features share a great degree of similarity. As a result, the selection of both features would amount to a waste of resources.

This problem can be fully resolved (possibly overdone) by adopting a group ranking view of the selected features, where the overall relevance of an entire group is considered together, taking into account mutual redundancies and similarities. The price of group ranking, however, is its excessively high computational cost. In order to find the best combination of features, an exhaustive search would consider every one of the 2^M possible combinations. It is clear, then, that we need to find a compromise between minimizing the computational cost and maximizing the effectiveness of the features selected.

3.3 Consecutive Search Approaches

One such compromise is consecutive search, which has two main approaches:

1. **Forward Search:** Such a search usually begins at an empty feature set, and iteratively adds features based on how much added value it brings to the existing subset, instead of purely on its individual merit or strength.
2. **Backward Elimination:** This approach begins with the full feature set, and iteratively removes features in a way such that it minimizes the information loss with respect to the remaining set. In short, at each step, it removes the feature whose absence has the smallest negative impact.

The consecutive search approach offers improved accuracy by taking into account both relevance and redundancy. It also offers substantial computational saving when compared with the comprehensive and exhaustive evaluation of the group scores. However, the downside is clear too. The order of feature selection can significantly affect the inter-feature redundancy revealed, which in turns affects the final outcome of the selection. In other words, the decision on selecting or eliminating any feature will depend on whether it is evaluated earlier or later in the process.

3.4 Filter vs. Wrapper Approaches

There are two main approaches used in selecting and evaluating features: *filter* and *wrapper* [12]:

- **Filter Approach** The filter method selects features using a mathematical score on the data set, independent of any classification algorithm applied to the data afterwards. It has computational simplicity and is less prone to overfitting, which makes it a promising and popular selection approach. For example, an SNR-type criterion based on the Fisher discriminant analysis [13] is often used.

- **Wrapper Approach** The wrapper method incorporates the actual learning algorithm into its feature selection decision, by feeding a set of features into the algorithm and evaluating their quality from the learner's output. An exhaustive search is computationally prohibitive, so common approaches to implementing the wrapper method include making use of a linear classification assumption, or performing consecutive search as above.

3.5 Other Approaches

There are also many other approaches available that seek a compromise between individual rankings and group rankings. For a detailed study of these approaches, such as branch and bound, floating sequential search, the reader is referred to [14].

4 Unsupervised Feature Selection

4.1 PCA-Type and Clustering-Type Algorithms

Traditional unsupervised learning algorithms are basically divided into two types: principal component analysis (PCA) and cluster discovery (e.g. k-means). Conceptually speaking, both types are very useful for unsupervised feature selection.

4.1.1 PCA-Type Algorithms

Traditional feature representation techniques such as PCA have been used successfully for reducing dimensionality in unsupervised data. The downside is that the newly extracted features from PCA are combinations of the original features. If we want to make predictions on new data in clinical applications, using the principal components would require that we measure all the genes (i.e. the original features) [15]. Obtaining a subset of the original features, on the other hand, will allow us to measure a smaller subset of genes for clinical applications; in addition, these subsets also retain their original biological meaning and lend themselves to more intuitive analysis.

For unsupervised feature selection, PCA can still be used as an intermediate step, however. For example, [16–18] use the results from PCA or PCA-related methods to select features, where the numerical execution of PCA is often performed by SVD [19].

4.1.2 Clustering-Type Algorithms

Clustering algorithms are frequently used for class discovery in genomics. In these cases, class labels (e.g. particular types of cancers) are not known a priori, and our goal is to discover biologically significant groupings of samples that capture some "natural" structure inherent in the data. For unsupervised data, dimension reduction is important since data points become more sparse in high dimensions and "distances between data points become relatively uniform" [20]. This makes it difficult for clustering algorithms that depend on minimizing Euclidian distances (such as k-means) to create meaningful clusters.

Just as feature selection and dimensionality reduction are essential for clustering, conversely clustering algorithms have been found instrumental for unsupervised feature selection. For example, we can perform feature selection by clustering time-course microarray data along the y-direction and select the most representative gene in each cluster. We can also use the clustering algorithm along the x-direction to evaluate the quality of the chosen features (i.e. see Section 4.2.2 for the unsupervised wrapper approach).

4.2 Filter and Wrapper Approaches

Like their supervised counterpart, unsupervised feature selection can be thought of as either *filter* or *wrapper* type approaches. Although there are some exceptions, the filter and PCA/SVD approaches usually go hand in hand (for example, see Eq. 4). So do the clustering and wrapper approaches (see Fig. 4).

4.2.1 Filter Approaches

Filter approaches to unsupervised feature selection are computationally simple approaches that attempt to rank features (in either a univariate way or a multivariate way) based on only the data distributions. Examples of ranking features in unsupervised data include utilizing the variance, entropy, density, or reliability of each feature [21].

In most problems (supervised and unsupervised), unsupervised *gene filtering* is often performed as a preprocessing step, where genes with flat expression levels over time or with low absolute values are removed [23]. For example, Fig. 3 shows the histogram of the variances of the gene profiles before and after the filtering for the yeast cell cycle data [22]. Note that a large number of low-variance genes have been removed after the filtering process. Beyond that, however, more sophisticated unsupervised gene selection tech-

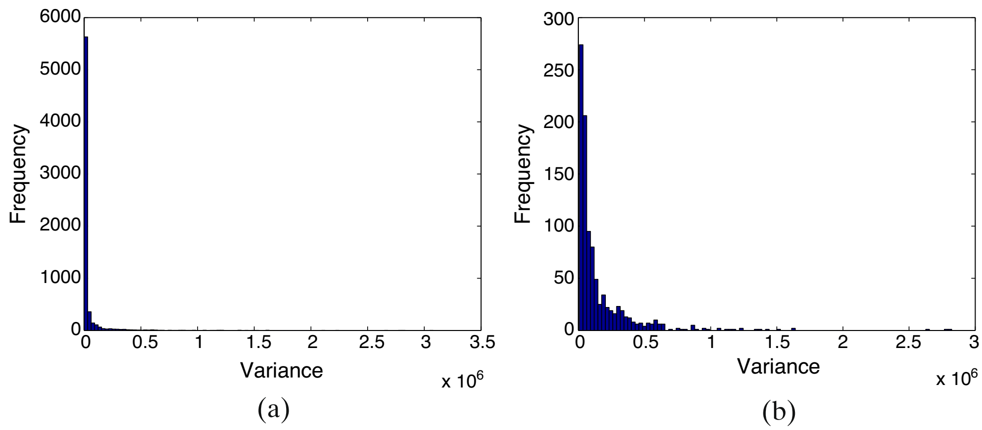

Figure 3 Histogram of the variances of the gene profiles of Tamayo et al.'s [22] yeast cell cycle data (un-normalized). **a** Before filtering. **b** After filtering.

niques have also been developed. Here, we will highlight a selection of techniques developed for genomic applications.

One of the earlier methods developed was "gene shaving" [16], which sought subsets of genes with large variations across samples (or conditions) that have similar expression patterns by iteratively discarding ("shaving off") the genes with the smallest variations. At each iteration, it finds a subset of genes and finds the largest principal component of the subset (called the *eigen gene*). Then, it calculates inner products of all the genes with this eigen gene and a fraction of the genes with the smallest absolute inner product values are removed from the subset.

The two-way ordering method proposed in [17] also used an iterative process to discard genes. It calculates the similarity between the genes using $w_{ij} = \exp c_{ij}/\bar{c}$, where c_{ij} is the Pearson correlation coefficients, and $\bar{c}$ is the average correlation. This similarity value was used to produce a weighted bipartite graph from the microarray data. The graph is then ordered in a way to simultaneously move the most similar genes and samples closer and the dissimilar genes and sampler further away from each other. The irrelevant genes with little discriminative power will be moved towards the middle and can then be discarded.

In [18], SVD-Entropy was used to select features. Suppose we have a matrix A that has singular values σ_j. Then the dataset entropy of the matrix is defined as:

$$E(A) = -\frac{1}{\log N} \sum_{j=1}^{N} V_j \log V_j \tag{4}$$

where $V_j = \sigma_j^2 / \sum_k \sigma_k^2$. A value of $E(A) = 0$ (low entropy) indicates an extremely ordered set and a value of $E(A) = 1$ indicates a highly disordered set. Then, leave-one-out comparison is used to define the contribution of each feature as the difference between the original dataset entropy $E(A)$ and the dataset entropy with feature i removed, $E(A')$. This can then facilitate a ranking system, where simple ranking (individual features), forward selection and backwards elimination can be used for feature selection.

4.2.2 Wrapper Approaches

Wrapper approaches in unsupervised feature selection work through an iterative process described in Fig. 4. Several difficulties arise in the wrapper approach for unsupervised feature selection. We must optimize the number of clusters at the same time as the feature space. In addition, if we use the quality of the clusters as a way of selecting the best feature subset, then finding

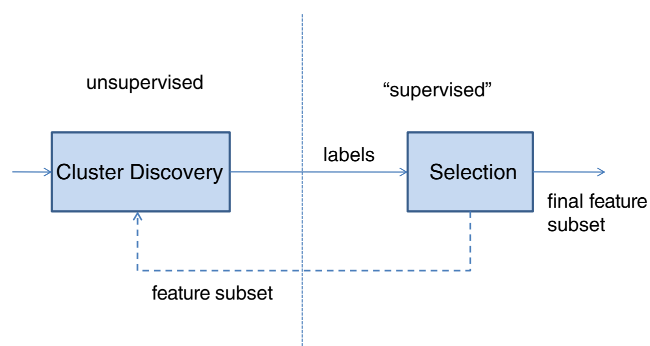

Figure 4 Typical procedure for the unsupervised wrapper approach, where a feature subset is fed into a clustering algorithm whose results are used to perform feature selection. At the *left side*, an unsupervised cluster discovery procedure is performed, generating "class labels" for each cluster. Once we have these class labels, we can solve the feature selection problem using a "supervised" feature approach (as shown in the *right side*). For example, the label information can be used for a ranking criterion such as Eq. 5. The *dashed feedback line* represents the possibility of using an iterative process, such as [24], where the features can be further pruned.

the "best" objective function or separatibility measure is also not immediately obvious.

An application of the wrapper approach in genomics is CLIFF [24], which alternates feature filtering using independent feature evaluation, information gain and Markov blanket filtering and a clustering algorithm based on normalized cuts. In the independent feature evaluation phase, the feature vectors are modelled as a mixture of Gaussians, and the features are ranked using the Bayesian error, $\epsilon_{Bayes} = \pi_0 P(h(x) = 1|z_x = 0) + \pi_1 P(h(x) = 0|z_x = 1)$. To measure information gain, the probabilities are measured by empirical proportion. For example, for a given clustering that produces a partition, S_c, out of the entire set, S, the probability $P(S_c) = |S_c|/|S|$. Suppose a feature induces a partitioning, $E_1, \cdots, E_K$. Then the information gain is

$$I_{gain} = H\big(P(S_1), \cdots, P(S_c)\big) - \sum_{k=1}^{K} P(E_k) H\big(P(S_1|E_k), \cdots, P(S_c|E_k)\big) \quad (5)$$

where H is defined as the entropy. Both independent feature evaluation and information gain are used to select the most relevant features. Then, Markov Blanket filtering is used to remove the redundant features, by finding a subset G of the features F such that for any clustering C, $P(C|F = f)$ and $P(C|G = f_G)$ are very similar. Here, f_G is simply the projection of f onto the variables in G.

Another application is the Bayesian Class Discovery method proposed in [25]. Here, the clustering algorithm is the EM (Expectation Maximization) algorithm, where the M step was replaced with LDA (Linear Discriminant Analysis). During this new M-Step, automatic feature selection is then performed by using ridge regression using the l_1 penalty, also known as the Least Absolute Shrinkage and Selection Operator (LASSO). With LASSO, many of the regression coefficients are shrinked to 0, which correspond to features that should be removed. Finally, resampling-based stability analysis is used to determine the parameters for optimizating the regression problem. This method takes into account difficulties in unsupervised feature selection that arise with multiple highly-ranked hypotheses on how to cluster the data.

While wrapper approaches typically follow Fig. 4, explicit clustering is not always required. For example, recently, [26] proposed Laplacian Linear Discriminant Analysis-based Recursive Feature Elimination (LLDA-RFE), which is a multivariate feature selection technique closely related to the Laplacian Score [27] and based on similar principles as the Q-α algorithm [28]. They extend the supervised LDA approach to work for the unsupervised case and wrap their feature selection algorithm around the unsupervised LDA approach, rather than performing an explicit clustering step.

Suppose for a moment that the labels of the training vectors are actually known (i.e. under supervised assumption), then the between-class scatter matrix $\mathbf{S}_b$ and the within-class scatter matrix $\mathbf{S}_w$ can be defined as:

$$\mathbf{S}_b = \frac{1}{n} \sum_{k=1}^{c} n_k \left(\mathbf{m}^{(k)} - \mathbf{m}\right) \left(\mathbf{m}^{(k)} - \mathbf{m}\right)^T$$

and

$$\mathbf{S}_w = \frac{1}{n} \sum_{k=1}^{c} \sum_{j=1}^{\mathbf{n}_k} \left(\mathbf{x}_j^{(k)} - \mathbf{m}^{(k)}\right) \left(\mathbf{x}^{(k)} - \mathbf{m}^{(k)}\right)^T$$

where c is the number of classes, n_k is the number of samples in class k, and n is the total number of samples.

The goal of the classical Fisher's discriminant (also known as LDA) finds the projection matrix $\mathbf{W}$ that maximizes the Fisher criterion:

$$J(\mathbf{W}) = trace \left\{ \frac{\mathbf{W}^T \mathbf{S}_b \mathbf{W}}{\mathbf{W}^T \mathbf{S}_w \mathbf{W}} \right\}, \quad (6)$$

subject to the orthogonality constraint: $\mathbf{W}^T\mathbf{W} = I$.

In many genomic applications, such as microarray experiments, the feature dimensionality is greater than the number of samples. In this case, $\mathbf{S}_w$ becomes singular. Such a singularity problem is usually regarded to be a serious liability of the classical LDA. To overcome this problem, the maximum margin criterion (MMC) was proposed by Li et al. [29]. In the MMC approach, one maximizes

$$J_{MMC}(\mathbf{W}) = trace \mathbf{W}^T \Delta \mathbf{W} :$$

where

$$\Delta = \mathbf{S}_b - \mathbf{S}_w.$$

Denote the total scatter matrix defined as:

$$\mathbf{S}_t = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$$

where it is assumed (WLOG) that the total mean vector $c = 0$. According to [30], $\mathbf{S}_t = \mathbf{S}_b + \mathbf{S}_w$, therefore,

$$\Delta = \mathbf{S}_b - \mathbf{S}_w = \mathbf{S}_t - 2\mathbf{S}_w$$

Thus the matrix Δ can be expressed as

$$\Delta = \mathbf{S}_t - 2\mathbf{S}_w = \frac{1}{n}\mathbf{X}\{I - 2(I - \mathbf{W}_L)\}\mathbf{X}^T : \quad (7)$$

where $\mathbf{W}_L$ is a matrix describing the intra-class relationship. For more details, the reader is referred to [26]. Here we shall treat the two-class case only. In this case, the entries $\mathbf{W}_L$ are is either $\frac{1}{n_1}$, $\frac{1}{n_2}$ or zero. More importantly, all the column entries must sum into 1, making it a probability-like distribution. (Likewise for all the row entries.)

For unsupervised case, just like Eq. 7, all the column entries again sum into 1. However, the entries of $\mathbf{W}_L$ are function of the distance between $\mathbf{x}_i$ and $\mathbf{x}_j$. By this approach, the explicit clustering is no longer necessary, which is the major advantage of the LLDA approach. The basic idea of LLDA may be applied to visualization or classification. For the visualization case, $\mathbf{W}$ is a $3 \times n$ matrix and for classification, $\mathbf{W}$ is a $1 \times n$ vector, denoted as $\mathbf{w}$.

Let $\delta_1, \cdots, \delta_n$ be the eigenvalues of Δ. Then, the weight of each feature j is defined as:

$$\sum_{i=1}^{d} \sqrt{\delta_i}\,|\mathbf{W}_{ji}| \quad (8)$$

where d is the number of positive eigenvalues. Once the weights are obtained, the features with the lowest weights are removed recursively. Even though both algorithms use the word RFE to describe the feature selection process, there is a critical difference. The process described in Fig. 6 finds an optimal decision boundary vector $\mathbf{w}$ whose coefficients are used for feature selection. LLDA-RFE never explicitly produces an optimal decision boundary vector before selection and instead ranks the features using weights in Eq. 8.

4.3 Application to Microarray Data

Unsupervised feature selection methods have been successfully applied to a variety of microarray data and yielded biologically relevant results. In [16], gene shaving was applied to data from patients with diffuse large B-cell lymphoma and was able to identify a cluster of genes highly indicative of survival. The two-way ordering method [17] was applied to the colon cancer dataset [31] and the leukemia dataset [13], and the features selected resulted in improved cluster accuracy and in the case of the colon cancer dataset, showed considerable overlap with the features selected using supervised methods. The SVD-Entropy method [18] was applied to two leukemia datasets [13, 32] and a virus dataset [33], where improved performance was found over variance selection methods and gene shaving. For the dataset in [13], significant GO enrichment was found as well.

When applied to the leukemia dataset [13], CLIFF [24] improved performance over clustering without feature selection and produced results close to the original labelling of the data. Bayesian class discovery [25] has been shown to find biologically relevant partitions on the leukemia dataset [13]. In experiments on seven microarray datasets [13, 31, 32, 34–37], LLDA-RFE [26] was found to outperform Laplacian Score and have favorable performance against SVD-Entropy, and on some datasets, even outperforms the supervised Fisher score.

5 Supervised Feature Selection

Traditional supervised feature selection methods can be divided into filter and wrapper approaches.[2]

5.1 Filter Approach

The predominant type of filter criterion in supervised feature selection is the SNR-type score [13]:

$$\text{Signed-SNR} = \frac{\mu^+ - \mu^-}{\sigma^+ + \sigma^-}, \quad (9)$$

where μ^+, μ^-, σ^+, and σ^- represent the class-conditional means and standard derivations of any single feature, respectively. Another example of the SNR metric is the symmetric divergence (SD) [41]:

$$\text{SD} = \frac{1}{2}\left(\frac{(\sigma^+)^2}{(\sigma^-)^2} + \frac{(\sigma^-)^2}{(\sigma^+)^2}\right) + \frac{1}{2}\left(\frac{(\mu^+ - \mu^-)^2}{(\sigma^+)^2 + (\sigma^-)^2}\right) - 1. \quad (10)$$

There are also other filter approaches—such as the t-test [42], Fisher discriminative ratio (FDR) [3], Bayesian technique [43, 44], BSS/WSS [45], and TNom

[2]In addition to the SNR-type filter and SVM-RFE, there exist an extremely large number of application studies based on microarray data. Two recent ones are the MRMR [38] and Markov blanket [39], which are based on the Multivariate techniques. Another recent approach is the VIA-SVM [40], which is more amendable to the self-supervised scenario explained in Section 6.

[46]—that have been applied for gene selection. All these methods select a small group of features based on whether or not the genes are significantly differentially expressed (as measured by their chosen criterion). Although some are based on statistical tests, factoring in multiple testing or false positive issues are not essential, since no claims are made on the statistical significance of the genes; the metrics are only used to decide on the inclusion of the genes for classification [15].

Because all of these methods are based on the notion of SNR, they produce comparable performance. In fact, it has been shown that the performance of signed-SNR, SNR, and FDR are fairly close [47].

5.2 Wrapper Approach

In the wrapper approach, the classification method is predetermined and the selected features are bounded to the type of linear classifier adopted [48, 49].

- The earliest and well-known example of a linear classifier is **Fisher-type approaches** such as Fisher's discriminant [30]. The goal of the classical Fisher's discriminant (also known as LDA) is to find the vector projection $\mathbf{w}$ that maximizes the Fisher criterion:

$$J(\mathbf{w}) = trace\left\{\frac{\mathbf{w}^T\mathbf{S}_b\mathbf{w}}{\mathbf{w}^T\mathbf{S}_w\mathbf{w}},\right\} \tag{11}$$

  If the numerical (singularity) problem is of grave concern, the MCC criterion may be used instead:

$$J_{MMC}(\mathbf{w}) = trace\left\{\mathbf{w}^T\Delta\mathbf{w}\right\} = \mathbf{w}^T\Delta\mathbf{w}.$$

  In this case, $\mathbf{w}$ is the normalized eigenvector corresponding to the largest eigenvalue. Once the optimal decision (slope) vector $\mathbf{w}$ is found, the best features can be readily determined by the wrapper method.
- One prominent method in this category is the **SVM approach** with recursive feature elimination (RFE) proposed by Guyon et al. [48]. The RFE algorithm eliminates unimportant features recursively based on the weights of linear SVMs, hence the name SVM-RFE. More precisely, the algorithm begins with using the full-feature training vectors $\mathbf{y} \in \Re^M$ to train a linear SVM. Briefly, the features are ranked by sorting the square of the SVM's weights $\{w_i^2\}_{i=1}^M$ in descending order, where the weight vector is given by

$$\mathbf{w} = [\mathbf{w}_1, \ldots, \mathbf{w}_M] \sum\nolimits_{k\in\mathcal{S}} \alpha_k l_k \mathbf{y}_k, \qquad \mathbf{w} \in \Re^M \tag{12}$$

  where α_k are the Lagrange multipliers, $\mathcal{S}$ contains the indexes of support vectors (SVs), and $l_k \in \{+1, -1\}$ is the class label of SV $\mathbf{y}_k$. Note that the decision boundary is controlled by $\mathbf{w}$, which is in turn controlled by α. The features corresponding to the larger $|w_i|$ are selected first, as exemplified in Fig. 5.

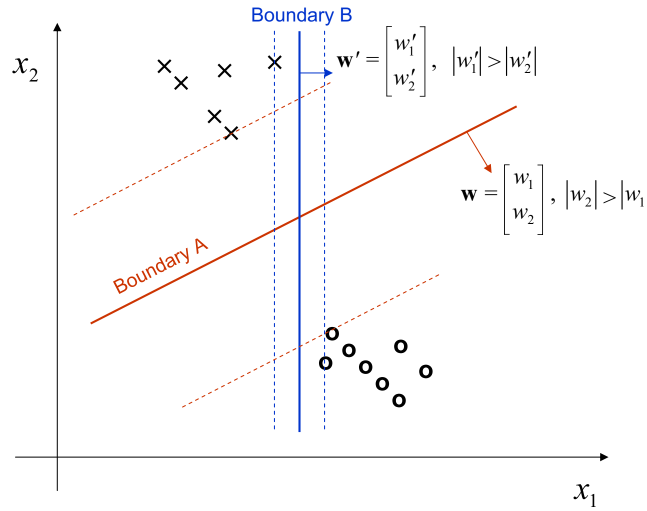

Figure 5 Selecting one out of two features by SVM-RFE. A linear SVM will use Boundary A for classification and therefore feature x_2 will be selected by ranking the square of SVM weights $\{w_1^2, w_2^2\}$ in descending order.

The RFE flowchart is shown in Fig. 6. The wrapper approach uses classification accuracies to rank the discriminative power of all of the possible feature subsets so that the selected subset is likely to produce the best performance. This conventional wrapper approach will be referred to as the reflexive type of wrapper approaches. For example, in the case of clinical applications of microarray data, features are selected along the y-direction and classification is along the x-direction, thus the name "reflexive".

The idea of SVM-RFE can be intuitively explained by considering a two-feature case as shown in Fig. 5. The figure shows two possible ways of separating the two classes of data. Boundary B (with weight vector $\mathbf{w}'$) is undesirable because of the small margin. On the other hand, Boundary A (with weight vector $\mathbf{w}$) is more desirable because of the large margin. In fact, a linear SVM will use Boundary A to classify the data. Notice that the weight vector $\mathbf{w} = [w_1 \;\; w_2]^{\mathrm{T}}$ in Fig. 5 has the property $w_2^2 > w_1^2$, which suggests that x_2 is a more discriminative feature.

5.3 Applications to Microarray Data

Supervised feature selection has been applied to microarray data extensively. Here, we use the cancer

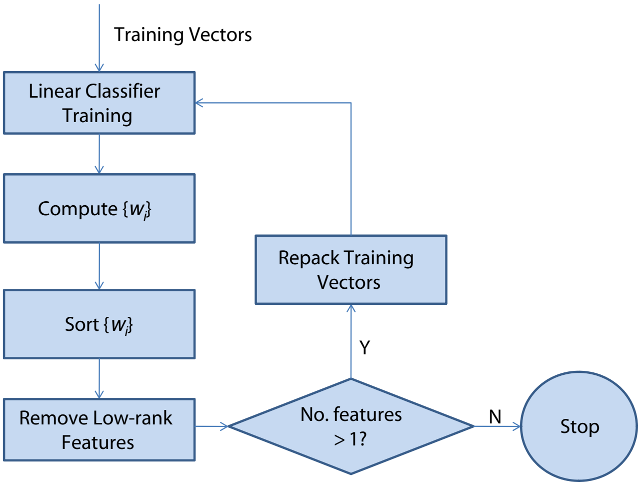

Figure 6 Flowchart for RFE. The procedure is applicable to any types of linear classifiers, including the Fisher-type and the SVM-type discussed in this section.

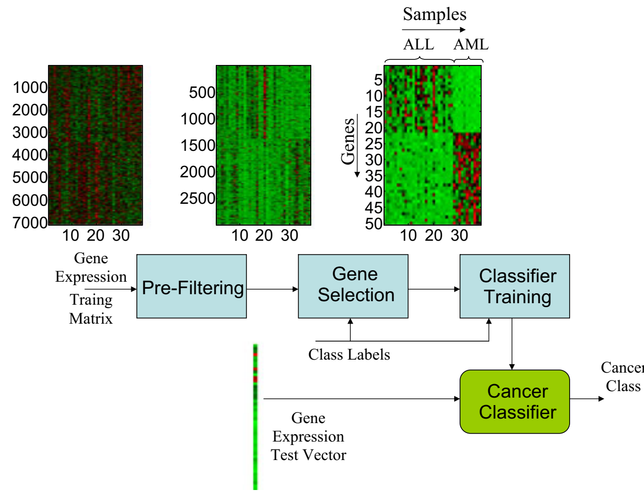

Figure 7 Procedures for building a cancer classification system based on gene expression data. The normalized gene expression matrices at various stages of the system building process are also shown.

classification problem provided by Golub et al. [50] as an illustrative example.

In [50], the microarray data contains 47 samples of acute lymphoblastic leukemia (ALL) and 25 samples of acute myeloid leukemia (AML). For each sample, the expression level of 7192 genes (of which 6817 human genes are of interest, the other are the controls) are measured, forming an expression matrix of size 7192 × 72. The matrix is further divided into a training set (containing 27 ALL and 11 AML cases) and test set (containing 20 ALL and 14 AML cases). This dataset[3] has now become a benchmark for gene selection algorithms and microarray cancer classification algorithms.

5.3.1 Gene Pre-Filtering

Among the 7129 genes in Golub's dataset, a majority of them are irrelevant to the classification task. In fact, many genes have expression value well beyond meaningful level. Therefore, it is imperative to weed out quickly those genes with small variation and extremely large expression values. Following [51], we removed gene i if it meets any of the following conditions:

(1) $\max_j g_{ij} - \min_j g_{ij} \leq 500$

(2) $\max_j g_{ij} > 16,000$

(3) $\left|\dfrac{\max_j g_{ij}}{\min_j g_{ij}}\right| \leq 5$

where g_{ij} is the expression level of gene i at training sample j. The gene expression image in Fig. 7 and the correlation matrix in Fig. 8b show that after this pre-filtering step, the two-class pattern begins to emerge. After this step, 2,729 genes remain in the training expression matrix, i.e., the matrix size is reduced to $2,729 \times 38$.

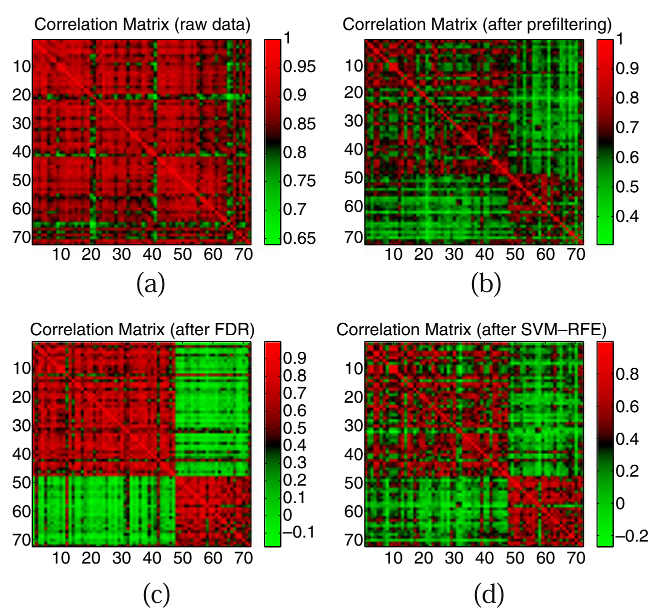

Figure 8 Correlation matrix of Leukemia dataset based on **a** raw data, **b** genes after filtering, **c** genes after signed-SNR selection, and **d** genes after SVM-RFE selection.

[3]Downloadable from the official site http://www.genome.wi.mit.edu/mpr

5.3.2 Gene Selection

Because the pre-filtering step does not make use of the class labels for selecting genes, there remain many irrelevant genes in the expression matrix. More irrelevant genes can be weeded out by using a supervised feature selection approach, which can be divided into *filter* and *wrapper* types (c.f. Section 5.1):

Following Golub's work [50], we selected 50 out of 2,729 genes. The two-class pattern in Fig. 7 becomes apparent, suggesting that the selected features are relevant for the classification task.

5.3.3 Training and Evaluation of Classifiers

Fifty-input linear SVMs were used to classify the AML against ALL patterns, i.e., one SVM for classifying features selected by signed-SNR and another for classifying features selected by SVM-RFE. Figure 9 shows the scores obtained by the SVMs together with the decision thresholds that lead to maximum accuracy. Confirming Golub's result, the accuracy is 100% for the SVM that bases on signed-SNR selected genes. The accuracy for SVM-RFE, however, is 97.1%.

To have a more detailed comparison between the capability of signed-SNR and SVM-RFE in selecting relevant genes, Fig. 10 plots the accuracy against the number of selected genes. Evidently, signed-SNR is superior to SVM-RFE for a wide range of feature dimension.

It is important to know the ranking of individual genes in case the number of allowable genes is very limited. To this end, signed-SNR and SVM-RFE were used to find the top five genes for the classification task. The accession number of the selected genes are shown in Table 1. Although the genes found by both selection methods are very different, these two sets of genes lead to the same prediction accuracy, which is 94.1%. It was also noticed that all of the genes found by signed-SNR are part of the 50 genes used in Golub et al's experiments. The SVM-RFE, on the other hand, has one gene not found by Golub et al.

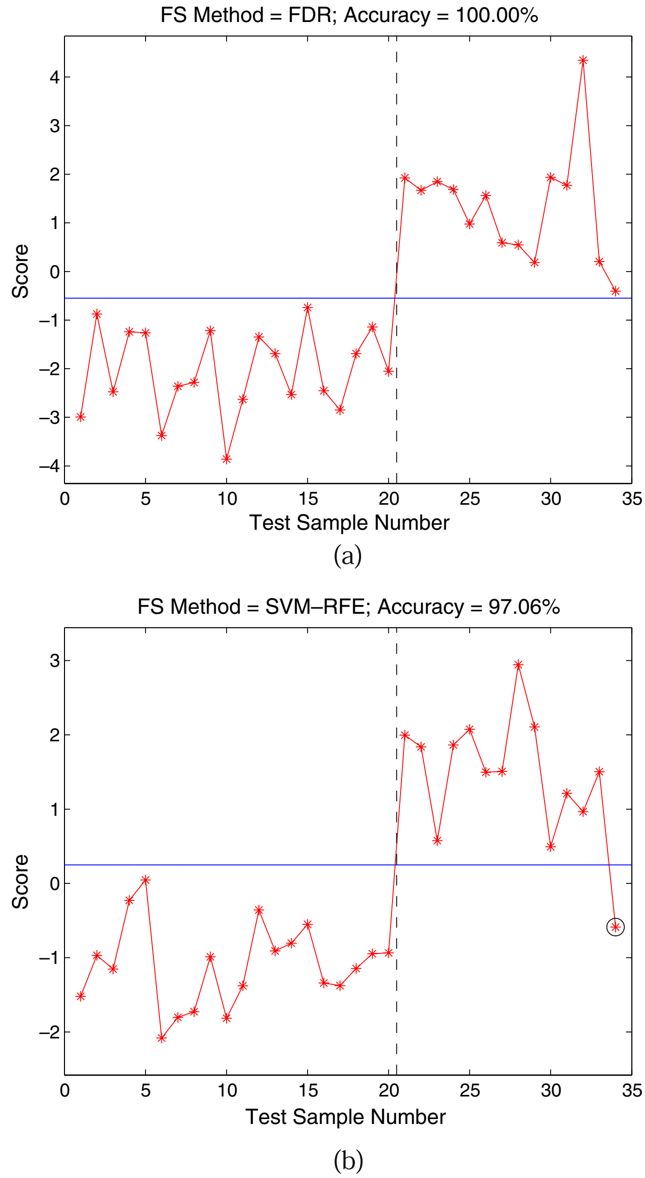

Figure 9 Prediction scores and decision threshold (blue horizontal line) of SVM classifiers based on 50 genes selected by **a** signed-SNR and **b** SVM-RFE. On the *left* (*right*) of the *vertical dashed line* are 20 ALL (14 AML) test samples. Incorrect predictions are highlighted by *black circles*.

6 Self-Supervised Scenario

6.1 Why is SS Formulation Biologically Appealing?

The SS formulation naturally arises in many genomic applications. For example, in time-course microarray data, the expression levels of M genes are measured over N time points. Therefore, the i-th row in Eq. 3 is

$$\mathbf{y}_i^T = \left[y_i(1), y_i(2), \ldots, y_i(N)\right] \quad i = 1, \ldots, M \tag{13}$$

Each of these vectors is assigned a class label. One interesting question is "Among these M genes, which are the most representatives for differentiation into different gene groups?" This question leads to a self-supervised formulation for feature selection, where features are genes.

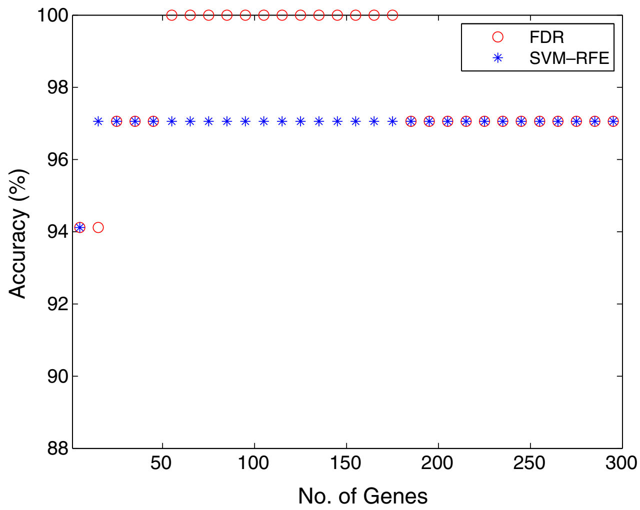

Figure 10 Prediction accuracy of SVM classifiers based on different numbers of genes selected by signed-SNR and SVM-RFE.

Example 2 (SS Formulation for Yeast Expression Data) In http://genomics.stanford.edu:16080/yeast_cell_cycle/cellcycle.html, the genes in the yeast data provided by [22] are divided into five stages of a cell cycle: Early G1, Late G1, S phase, G2 phase, and M phase. This results in the SS formulation with a 421×16 expression matrix (421 genes at 16 time points).

Example 3 (SS Formulation for Sequence Selection) As mentioned in Section 2.1, a sequence can be converted into a vector via the k-mers or motif counts. As an example, a protein sequence can be represented by a 20-dimensional vector when $k = 1$ or by a 3,200,000-dimensional vector when $k = 5$. Then, referring to Eq. 3, M sequences can be represented by an $M \times N$ matrix. Suppose that the sequences have known class labels and that the goal is to select m relevant sequences out of the M sequences, we have basically a self-supervised problem.

Table 1 The accession numbers of the genes selected by signed-SNR and SVM-RFE when the maximum number of genes to be selected is set to 5.

Rank	Signed-SNR	SVM-RFE
1	U22376	X04085
2	M55150	M19507*
3	U50136	M28130
4	X95735	U46751
5	U82759	X17042

All genes are part of Golub et al.'s gene set, except for the one with an asterisk.

6.2 Supervised Versus Self-Supervised Approaches

The two different supervision scenarios naturally require different kinds of filter and wrapper approaches.

- **Regular Supervised Scenario**:
  1. *Filter approaches*. SNR-based criteria are used for ranking features, see Section 5.1.
  2. *Reflexive wrapper approach*. The class labels along the x-direction of $\mathbf{Z}$ in Eq. 3 are used to guide the selection along the y-direction of $\mathbf{Z}$. One important example is the SVM-RFE, see Section 5.2.

- **Self-Supervised Scenario**:
  1. *Filter Approaches*. The relevance of each row vector, say $\mathbf{y}_i$, in $\mathbf{Z}$ is ranked by the variance of its element. Because the labels of the row vectors are not used, this is equivalent to unsupervised feature selection discussed in Section 4.2. However, if we can convert the SS scenario into a symmetric doubly supervised (SDS) one (see Section 6.3.4), the differential expression of individual features along the y-direction offers an effective SNR metric for feature selection.
  2. *Direct Wrapper Approach*. This is a relatively new approach in that classification and selection are both along the y-direction of the data matrix, i.e., class labels along the y-direction of $\mathbf{Z}$ are used to guide the selection of features (e.g., sequences or genes) along the y-direction. This approach will be more natural and appropriate for sequence selection mentioned earlier. One implementation is the VIA-SVM, which is to be elaborated in Section 6.3 below.

6.3 The VIA-SVM Scheme

The vector-index-adaptive SVM (VIA-SVM) [10], designed specifically for the self-supervised formulation, selects a subset of critical vectors from a pool of SVs. For simplicity, we shall consider the two-class case first, as shown in Fig. 11a.

First, the SVs are deemed to be a good candidate pool. The reason is mainly due to the proximity/importance of the SVs to the decision boundary. The next phase is to select a subset of "critical vectors" from the pool of SVs. To this end, the SVs are further subdivided into two groups, according to whether the

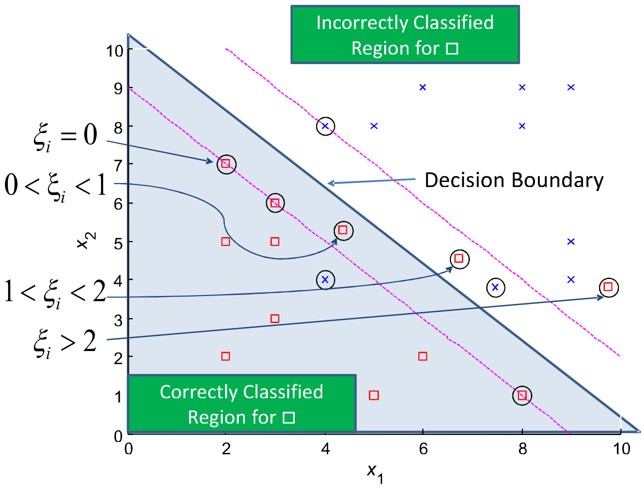

Figure 11 SVs are divided into two types: correctly classified ($0 \le \xi_i < 1$ and incorrectly classified ($\xi_i > 1$). For clarity of presentation, only the SVs correspond to Class 1 (marker *square*) are highlighted.

slack variables ξ_i are above or below certain threshold θ, i.e., if $\xi_i < \theta$ then the corresponding SVs will be selected.

6.3.1 Feature Selection Based on Adaptive Threshold

Sometimes, it is desirable to have some flexibility on the number of selected SVs. This can be achieved by adjusting the threshold θ. The smaller the threshold the less SVs will be selected. This adaptive scheme is well supported by our simulation study in [10], which showed that correctly classified SVs performed well except when the SVs are very close to the decision boundary (i.e. these SVs had a much higher averaged value of ξ than the rest of the group).

6.3.2 Feature Selection Based on Correctly Classified SVs

An important special case is to set the selection threshold to 1, i.e., $\theta = 1$, which corresponds to the decision boundary as shown in Fig. 11.

1. **Correctly Classified SVs**: The selected SVs correspond to $0 \le \xi_i < 1$. They are situated on the correct side of the decision boundary.
2. **Incorrectly Classified SVs**: The removed SVs are in the region $\xi_i \ge 1$.

6.3.3 VIA-SVM for Multiclass Training and Testing

Assume that the data have M training samples and R classes. The row vectors of $\mathbf{Z}$ are used to train R SVMs from which R sets of SV indexes $\mathcal{S}_r$ ($r = 1, \ldots, R$) are determined. This results in a set of SVs $\mathbf{z}_j$ for each class, where $j \in \mathcal{S}_r$. Then, for the r-th class, the indexes in $\mathcal{S}_r$ are used as a candidate pool for selecting the rows of $\mathbf{Z}$. (This process is repeated for all classes.)

6.3.4 Extension from SS to SDS

For vectorial data, one can apply the VIA-SVM directly to the original SS matrix, cf. Eq. 3. However, our theoretical and experimental studies suggest that the real usage of VIA-SVM hinges upon an extension of the original SS formulation to the so-called SDS formulation. Moreover, the SDS formulation also copes very well with nonvectorial data, such as sequence data.

Let us now further describe how to convert the SS formulation into a SDS formulation by a pairwise approach. Mathematically, a symmetric score matrix $\mathbf{S}_{\mathbf{yy}} \in \Re^{M \times M}$ can be obtained as follows:

$$\mathbf{S}_{\mathbf{yy}} = \begin{bmatrix} S(\mathbf{y}_1, \mathbf{y}_1) & S(\mathbf{y}_1, \mathbf{y}_2) & \cdots & S(\mathbf{y}_1, \mathbf{y}_M) \\ S(\mathbf{y}_2, \mathbf{y}_1) & S(\mathbf{y}_2, \mathbf{y}_2) & \cdots & S(\mathbf{y}_2, \mathbf{y}_M) \\ \cdots & \cdots & \cdots & \cdots \\ S(\mathbf{y}_M, \mathbf{y}_1) & S(\mathbf{y}_M, \mathbf{y}_2) & \cdots & S(\mathbf{y}_M, \mathbf{y}_M) \end{bmatrix}, \quad (14)$$

where $S(\mathbf{y}_i, \mathbf{y}_j)$ represents a similarity score between $\mathbf{y}_i$ and $\mathbf{y}_j$.[4]

For example, for sequence data, we can use an alignment score such as the Smith-Waterman [52] score. The SDS formulation for sequence data then allows us to convert variable-length sequences into fixed-length vectors with dimension equal to the number of sequences (M) in the training set. Then, feature selection can be applied to select m (where $m \ll M$) features along the y-direction of the matrix to form M training vectors of m dimensions. These m-dimensional vectors are then used to train a classifier, e.g., an m-input SVM. During the retrieval phase, a query sequence is compared with the m selected sequences to form an m-dimensional test vector which is to be fed to the classifier. Because sequence alignment can be time consuming, reducing the number of inputs from M to m represents a significant computation saving during the retrieval phase.

[4]Here, we use bold face to represent both vectorial data such as gene expression profiles and non-vectorial data such as sequences.

Rich Information Pertaining to SDS Formulation. Under the SDS formulation, the data matrix is symmetric. Therefore, the class labels are known for rows and columns of the symmetric matrix (i.e. the class labels exist not only for the y-direction but also for the x′-direction due to the symmetry property).

A promising approach is to design a fusion strategy which may fully take advantages of the rich and diversified information embedded in the SDS matrix. Two prominent examples are:

1. In the original SS formulation, only VIA-SVM can be applied to do feature selection. Now that the class labels are available also for the x′-direction, the reflexive wrapper approach such as SVM-RFE can also be applied.
2. It is now possible to combine VIA-SVM and filter approaches such as SNR to improve the selection. This motivates us to propose an SVM-filtering fusion scheme to combine various information made available by the SDS formulation. For this, an overselect-and-prune strategy is proposed, which is discussed below.

To fuse the information pertaining to the SDS formulation, the following overselect-and-prune strategy is adopted:

1. **Over-Selection Phase.** This phase involves a quick and coarse (suboptimal) evaluation. This phase can be implemented by filtering or by selecting more SVs in VIA-SVM.
2. **Pruning Phase.** This phase serves as a fine-tuning process. It can be achieved by relevance filtering based on features' SNR.

With SDS, feature selection via SNR-based filtering is now allowed.

The theoretical justification on why VIA-SVM and SNR complement well with each other is briefly explained here. Note that SNR is an individual measurement whose score is independently computed. However, the SNR metric fails to take the inter-feature interaction into account. On the other hand, VIA-SVM is based on a collective decision after considering all the training vectors (features). Therefore, the two types of information are inherently different and can complement each other.

6.3.5 Simulation Studies Using VIA-SVM

Our VIA-SVM algorithm was applied to both microarray data and sequence data in [10]. Here, we will highlight the significant results from our studies.

In our finding, VIA-SVM performs well on the microarray yeast cell cycle data [22] over a wide range of feature size, even when the number of features is reduced to a single digit. Compared to the performance clustering the data in its original self-supervised (time-course) formulation using k-means as a method of feature selection with a nearest-neighbor classifier, our SDS formulation using VIA-SVM and SVM-RFE consistently and noticeably outperform the SS formulation.

On the other hand, our sequence data provided by Huang and Li [53] starts with more than 3000 features, and VIA-SVM can successfully reduce the dimension by more than one order of magnitude (i.e. 10 times). In our simulations, VIA-SVM is superior to SVM-RFE in two aspects: (1) It outperforms SVM-RFE at almost all feature dimension, particularly at low feature dimensions and (2) it automatically bounds the number of selected features within a small range. A drawback of SVM-RFE is that it requires a cutoff point for stopping the selection. On the other hand, VIA-SVM is insensitive to the penalty factor in SVM training and can avoid the need to set a cutoff point for stopping the feature selection process.

When the over-select-and-prune cascaded fusion architecture was adopted, the strategy produced more compact feature subsets without significant reduction in prediction accuracy. We also note that although VIA-SVM is inferior to SVM-RFE for large feature-set size, the combination of SD (a filtering metric) and VIA-SVM performs better at small feature-set size.

7 Conclusion and Future Work

This paper reviews the applications and techniques of feature selection for genomic signal processing. Many prominent techniques for unsupervised, supervised, and self-supervised scenarios. The paper also provides a number of experimental results primarily on microarray (sample and time-course) data and gene/protein sequence selection. More works will be needed on motif selection applications before any concrete results can be included. Other areas to explore the benefits of feature selection may include text/literature mining, SNPs (Single Nucleotide Polymorphisms), and integrating different data sources [54].

Acknowledgements This work was in part supported by The Research Grant Council of the Hong Kong SAR (Project No. PolyU 5241/07E, PolyU 5251/08E, and A-PH18).

References

1. Guo, J., Mak, M.-W., & Kung, S. Y. (2006). Eukaryotic protein subcellular localization based on local pairwise profile alignment SVM. In *2006 IEEE international workshop on machine learning for signal processing* (*MLSP'06*) (pp. 391–396).
2. Reinhardt, A., & Hubbard, T. (1998). Using neural networks for prediction of the subcellular location of proteins. *Nucleic Acids Research, 26*, 2230–2236.
3. Pavlidis, P., Weston, J., Cai, J., & Grundy, W. N. (2001). Gene functional classification from heterogeneous data. In *Int. conf. on computational biology* (pp. 249–255). Pittsburgh: PA.
4. Leslie, C., ESKIN, E., & Noble, W. S. (2002). The spectrum kernel: A string kernel for SVM protein classification. In Altman, R. B., Dunker, A. K., Hunter, L., Lauredale, K., & Klein, T. E. (Eds.) *Proc. of the pacific symposium on biocomputing*. River Edge: World Scientific.
5. Leslie, C. S., Eskin, E., Cohen, A., Weston, J., & Noble, W. S. (2004). Mismatch string kernels for discriminative protein classification. *Bioinformatics, 20*(4), 467–476.
6. Ben-Hur, A., & Brutlag, D. (2004). Sequence motifs: Highly predictive features of protein function. *Neural Information Processing Systems 2004*.
7. Kuang, R., Ie, E., Wang, K., Wang, K., Siddiqi, M., Freund, Y., & Leslie, C. (2004). Profile-based string kernels for remote homology detection and motif extraction. *Computational Systems Bioinformatics Conference, 2004. CSB 2004. Proceedings. 2004 IEEE* (pp. 152–160).
8. Gao, Q., & Wang, Z. (2006). Feature subset selection for protein subcellular localization prediction. *Lecture Notes in Computer Science*, (Vol. 4115, p. 433).
9. Su, Y., Murali, T. M., Pavlovic, V., Schaffer, M., & Kasif, S. (2003). *RankGene: Identification of diagnostic genes based on expression data* (vol. 19). Oxford: Oxford University Press.
10. Kung, S. Y., & Mak, M.-W. (2008). Feature selection for self-supervised classification with applications to microarray and sequence data. *IEEE Journal of Selected Topics in Signal Processing: Special Issue on Genomic and Proteomic Signal Processing, 2*, 297–309.
11. Huang, C., Lin, C., & Pal, N. (2003). Hierarchical learning architecture with automatic feature selection for multiclass protein fold classification. *NanoBioscience, IEEE Transactions on, 2*, 221–232.
12. Kohavi, R., & John, G. H. (1997). Wrappers for feature selection. *Artificial Intelligence, 97*(1–2), 273–324.
13. Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., et al. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science, 286*, 531–537.
14. Kudo, M., & Sklansky, J. (2000). Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition, 33*(1), 25–41.
15. Simon, R. (2003). Diagnostic and prognostic prediction using gene expression profiles in high-dimensional microarray data. *British Journal of Cancer, 89*(9), 1599–1604.
16. Hastie, T., Tibshirani, R., Eisen, M., Alizadeh, A., Levy, R., Staudt, L., et al. (2000). 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology, 1*(2), research0003.1–research0003.21.
17. Ding, C. (2003). Unsupervised feature selection via two-way ordering in gene expression analysis. *Bioinformatics, 19*(10), 1259–1266.
18. Varshavsky, R., Gottlieb, A., Linial, M., & Horn, D. (2006). Novel unsupervised feature filtering of biological data. *Bioinformatics, 22*(14), e507–e513.
19. Golub, G. H., & Loan, C. F. V. (1996) *Matrix computations*. Baltimore: Johns Hopkins University Press.
20. Steinbach, M., Ertöz, L., & Kumar, V. (2003). The challenges of clustering high dimensional data. In: *New vistas in statistical physics: Applications in econophysics, bioinformatics, and pattern recognition*. New York: Springer.
21. Guyon, I., Elisseefi, A., & Kaelbling, L. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*(7–8), 1157–1182.
22. Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., et al. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proceedings of the National Academy of Sciences, 96*, 2907–2912, Mar.
23. Kohane, I. S., Kho, A. T., & Butte, A. J. (2003) *Microarrays for an integrative genomics*. Cambridge: MIT.
24. Xing, E., & Karp, R. (2001). CLIFF: Clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. *Bioinformatics, 17*(90001), 306–315.
25. Roth, V., & Lange, T. (2004). Bayesian class discovery in microarray datasets. *Biomedical Engineering, IEEE Transactions on, 51*(5), 707–718.
26. Niijima, S., & Okuno, Y. (2007). Laplacian linear discriminant analysis approach to unsupervised feature selection. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 10*, 20 Oct. doi:10.1109/TCBB.2007.70257.
27. He, X., Cai, D., & Niyogi, P. (2005). Laplacian score for feature selection. *Advances in Neural Information Processing Systems, 18*, 507–514.
28. Wolf, L., & Shashua, A. (2005). Feature selection for unsupervised and supervised inference: The emergence of sparsity in a weight-based approach. *The Journal of Machine Learning Research, 6*, 1855–1887.
29. Li, H., Jiang, T., & Zhang, K. (2006). Efficient and robust feature extraction by maximum margin criterion. *Neural Networks, IEEE Transactions on, 17*, 157–165.
30. Fukunaga, K. (1990). *Introduction to statistical pattern recognition*. London: Academic.
31. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., et al. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences of the United States of America, 96*(12), 6745.
32. Armstrong, S., Staunton, J., Silverman, L., Pieters, R., den Boer, M., Minden, M., et al. (2002). MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics, 30*(1), 41–47.
33. Fauquet, C., Desbois, D., Fargette, D., & Vidal, G. (1988). Classification of furoviruses based on the amino acid composition of their coat proteins. *Viruses with fungal vectors* (pp. 19–38). Wellesbourne: Association of Applied Biologists.

34. Pomeroy, S., Tamayo, P., Gaasenbeek, M., Sturla, L., Angelo, M., McLaughlin, M., et al. (2002). Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature, 415*(6870), 436–442.
35. van 't Veer, L. J., Dai, H., van de Vijver, M. J., He, Y. D., Hart, A. A. M., Mao, M., et al. (2002). Gene expression profiling predicts clinical outcome of breast cancer. *Nature, 415*, 530–536.
36. Beer, D. G., Kardia, S. L., Huang, C.-C., Giordano, T. J., Levin, A. M., Misek, D. E., et al. (2002). Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Natural Medicines, 8*, 816–824.
37. Khan, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., et al. (2001) Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Natural Medicines, 7*, 673–679, June.
38. Ding, C., & Peng, H. (2003). Minimum redundancy feature selection from microarray gene expression data. *Bioinformatics Conference, 2003. CSB 2003. Proceedings of the 2003 IEEE* (pp. 523–528).
39. Gevaert, O., Smet, F. D., Timmerman, D., Moreau, Y., & Moor, B. D. (2006). Predicting the prognosis of breast cancer by integrating clinical and microarray data with bayesian networks. *Bioinformatics, 22*, 184–190.
40. Kung, S. Y., & Mak, M.-W. (2008). *Machine learning for bioinformatics: An introduction to engineers*. Cambridge: Cambridge University Press.
41. Mak, M.-W., & Kung, S. Y. (2006). A solution to the curse of dimensionality problem in pairwise scoring techniques. In *Int. conf. on neural information processing* (pp. 314–323).
42. Jafari, P., & Azuaje, F. (2006). An assessment of recently published gene expression data analyses: Reporting experimental design and statistical factors. *BMC Medical Informatics, 6*(27), 27–35.
43. Baldi, P., & Brunak, S. (2001) *Bioinformatics: The machine learning approach* (2nd ed). Cambridge: MIT.
44. Fox, R. J., & Dimmic, M. W. (2006). A two-sample bayesian t-test for microarray data. *BMC Bioinformatics, 7*, 126.
45. Dudoit, S., Fridlyand, J., & Speed, T. P. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association, 97*, 77–88.
46. Ben-Dor, A., Bruhn, L., Friedman, N., Nachman, I., Schummer, M., & Yakhini, Z. (2000). Tissue classification with gene expression profiles. *Journal of Computational Biology, 7*, 559–583.
47. Mak, M.-W., & Kung, S. Y. (2008). Fusion of feature selection methods for pairwise scoring svm. *Neurocomputing, special issue for ICONIP'06*.
48. Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning, 46*, 389–422.
49. Zhang, X. G., Lu, X., Shi, Q., Xu, X. Q., Leung, H. C. E., Harris, L. N., et al. (2006). Recursive SVM feature selection and sample classification for mass-spectrometry and microarray data. *BMC Bioinformatics, 7*(197), 197–210.
50. Golub, T. R., Slonim, D. K., Tamayo, C. H. P., Gaasenbeek, M., Mesirov, J. P., Coller, H., et al. (1999). Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science, 286*, 531–537, Oct.
51. Dudoit, S., Fridlyand, J., & Speed, T. P. (2000). Comparison of discrimination methods for the classification of tumors using gene expression data. Technical Report 576, Dept. of Statistics, University of California, Berkeley, Berkeley, CA 94720-3860.
52. Smith, T. F., & Waterman, M. S. (1981). Comparison of biosequences. *Advances in Applied Mathematics, 2*, 482–489.
53. Huang, Y., & Li, Y. D. (2004). Prediction of protein subcellular locations using fuzzy K-NN method. *Bioinformatics, 20*(1), 21–28.
54. Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics, 23*(19), 2507.

S. Y. Kung is a Professor at Department of Electrical Engineering in Princeton University. His research areas include VLSI array processors, system modeling and identification, neural networks, wireless communication, sensor array processing, multimedia signal processing, bioinformatic data mining and biometric authentication. He was a founding member of several Technical Committees (TC) of the IEEE Signal Processing Society, and was appointed as the first Associate Editor in VLSI Area (1984) and later the first Associate Editor in Neural Network (1991) for the IEEE Transactions on Signal Processing. He has been a Fellow of IEEE since 1988. He served as a Member of the Board of Governors of the IEEE Signal Processing Society (1989–1991). Since 1990, he has been the Editor-In-Chief of the Journal of VLSI Signal Processing Systems. He was a recipient of IEEE Signal Processing Society's Technical Achievement Award for the contributions on "parallel processing and neural network algorithms for signal processing" (1992); a Distinguished Lecturer of IEEE Signal Processing Society (1994); a recipient of IEEE Signal Processing Society's Best Paper Award for his publication on principal component neural networks (1996); and a recipient of the IEEE Third Millennium Medal (2000). He has

authored and co-authored more than 400 technical publications and numerous textbooks including “VLSI and Modern Signal Processing”, Prentice-Hall (1985), “VLSI Array Processors”, Prentice-Hall (1988); “Digital Neural Networks”, Prentice-Hall (1993); “Principal Component Neural Networks”, John-Wiley (1996); and “Biometric Authentication: A Machine Learning Approach”, Prentice-Hall (2004).

Yuhui Luo received her B.A.Sc. with Honours in Electrical Engineering from the University of Toronto in 2007. She is currently a doctoral student at Princeton University's Electrical Engineering Department, working under the supervision of Prof. SY Kung. Her research interests include bioinformatics and digital signal processing.

Man-Wai Mak received a BEng(Hons) degree in Electronic Engineering from Newcastle Upon Tyne Polytechnic in 1989 and a PhD degree in Electronic Engineering from the University of Northumbria at Newcastle in 1993. He joined the Department of Electronic and Information Engineering at The Hong Kong Polytechnic University in 1993. He has authored more than 100 technical articles in speaker recognition, machine learning, and bioinformatics, and served as a guest editor of international journals. Dr. Mak is also a co-author of the postgraduate textbook “Biometric Authentication: A Machine Learning Approach, Prentice Hall, 2005.” Dr. Mak received seven research grants from the RGC of Hong Kong over the last eight years, and received a Faculty of Engineering Research Grant Achievement Award in 2003. Since 1995, Dr. Mak has been an Executive Committee member of the IEEE Hong Kong Section Computer Chapter. He was the Chairman of the IEEE Hong Kong Section Computer Chapter in 2003–2005. He also served as a member of the IEEE Machine Learning for Signal Processing Technical Committee in 2005–2007 and a member of Technical Committee Member, IEEE Computation Intelligence Society, Intelligent Systems Applications in 2008. Dr. Mak was a Program Committee member of a number of international conferences, including MLSP'06-08, ISCSLP'07-08, BIBE'07, PCM'07 (Track Co-Chair), ICMLC'07-08, ISCSLP'07-08, etc. Dr. Mak's research interests include speaker recognition, machine learning, and bioinformatics.