Feature Selection for Nonlinear Regression and its Application to Cancer Research

Yijun Sun* Jin Yao† Steve Goodison‡

Abstract

Feature selection is a fundamental problem in machine learning. With the advent of high-throughput technologies, it becomes increasingly important in a wide range of scientific disciplines. In this paper, we consider the problem of feature selection for high-dimensional nonlinear regression. This problem has not yet been well addressed in the community, and existing methods suffer from issues such as local minima, simplified model assumptions, high computational complexity and selected features not directly related to learning accuracy. We propose a new wrapper method that addresses some of these issues. We start by developing a new approach to estimating sample responses and prediction errors, and then deploy a feature weighting strategy to find a feature subspace where a prediction error function is minimized. We formulate it as an optimization problem within the SVM framework and solve it using an iterative approach. In each iteration, a gradient descent based approach is derived to efficiently find a solution. A large-scale simulation study is performed on four synthetic and nine cancer microarray datasets that demonstrates the effectiveness of the proposed method.

Keywords: nonlinear regression, feature selection, bioinformatics

1 Introduction

High-throughput technologies now routinely produce large datasets characterized by unprecedented numbers of features. The performance of most learning algorithms suffers as the number of features becomes excessively large. This is typically due to the requirement that a training dataset used to estimate algorithm parameters needs to increase in size exponentially with the growing number of features - a phenomenon called *the curse of dimensionality.* One possible way to address the issue is to perform feature selection to extract the most relevant information about each observed datum from a potentially overwhelming quantity of its features [7]. An example where feature selection plays a critical role is the use of oligonucleotide microarray for the identification of cancer-associated gene expression profiles of prognostic value. Typically, the number of samples is around one hundred, while the number of genes associated with raw data is on the order of thousands or even tens of thousands. The identification of a small fraction of genes that drive cancerous tumor growth and/or spread can significantly improve the accuracy of cancer prognosis. In addition to defying the curse of dimensionality, eliminating irrelevant features can also reduce processing time of data analysis and the cost of collecting irrelevant features. In many cases, feature selection can also provide significant insights into the nature of the problem under investigation.

The problem of feature selection has been extensively studied in the machine learning community [11, 7, 23, 24, 25]. However, the majority of the work is for classification and linear regression exemplified by Lasso [25] and its variants [26], and only a limited work has been done for nonlinear regression. Recent years have witnessed significant progress on the development of feature selection algorithms for nonlinear regression. Representative methods include mRMR [19], FVM [13], HSIC Lasso [27], QPFS [21], sparse additive model (SpAM) [20], hierarchical multiple kernel learning (HMKL) [1] and RGS [16]. These algorithms can be categorized as wrapper or filter methods. Filter methods are independent of any learning algorithm and select informative features based on some statistical properties of data (e.g., correlation). Therefore, filter methods can be easily implemented and are computationally very efficient. A major drawback of filter methods is that the criteria used in selecting relevant features are not directly related to learning accuracy. It is generally believed that a wrapper method that selects features by wrapping a selection process around a learning algorithm usually outperforms filter methods [7]. However, due to the difficulty of modeling complex data structures (e.g., nonlinear manifolds with multiple branches

*Bioinformatics Laboratory, State University of New York at Buffalo, Buffalo, NY 14201, USA

†Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32610, USA

‡Department of Health Sciences Research, Mayo Clinic, Jacksonville, FL 32224, USA

such as the one shown in Fig. 1), there are only a few wrapper based algorithms reported in the literature, including SpAM [20], HMKL [1] and RGS [16]. RGS is probably one of the first feature selection algorithms for nonlinear regression. The basic idea is to use kernel regression to predict responses and find a feature subset to minimize prediction errors. However, RGS suffers from a local minimum problem. Moreover, it does not offer a principled way to achieve a sparse solution that is usually required for high-dimensional data analysis. One method that worth mentioning is the well-known SpAM algorithm. SpAM assumes that data is generated by an additive model, and performs extremely well if the model assumption is valid. However, it may not be able to identify correct features if there are interactions among features as noted by [27]. This is undesirable for biological applications since genes and gene products do interact with each other.

This paper presents a new wrapper method that addresses some limitations of existing methods. We first develop a new approach to estimating sample responses and prediction errors. We then use a feature weighting strategy to find a feature subspace where an error function is minimized. We formulate it as an optimization problem with a well-defined objective function within the SVM framework, and solve it by using an iterative approach. In each iteration, a gradient descent based approach is derived to efficiently find a solution. The algorithm can be easily implemented and is computationally very efficient. Moreover, our method does not explicitly impose any model assumption on data distribution, and works for both linear and nonlinear problems. We demonstrate the effectiveness of the algorithm by applying it to four synthetic and nine cancer gene expression datasets.

2 Algorithm

Suppose that we have a training dataset $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$, where $\mathbf{x}_n \in \mathcal{R}^J$ is the n-th sample, $y_n \in \mathcal{R}$ is its corresponding response and $J \gg N$. We seek to find a feature subset so that the responses of unseen test samples can be optimally predicted based on some criteria. Therefore, the essence is to design a criterion to quantify prediction accuracy that can be conveniently optimized by some optimization techniques. To this end, we digress slightly and consider the nonlinear regression problem for the moment. Given a sample $\mathbf{x}$, a straightforward approach is to find a sample $\mathbf{x}^*$ in $\mathcal{D}$ that is closest to $\mathbf{x}$ and assign the response of $\mathbf{x}^*$ to $\mathbf{x}$:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}_n \in \mathcal{D}} d(\mathbf{x}, \mathbf{x}_n), \quad \text{and} \quad \hat{y}(\mathbf{x}) = y(\mathbf{x}^*) \ , \tag{2.1}$$

where $d(\mathbf{x}, \mathbf{x}_n)$ is a distance function measuring the similarity between two samples. A general version is the Nadaraya-Watson method [9] that estimates the response of $\mathbf{x}$ as:

$$\hat{y}(\mathbf{x}) = \sum_{n=1}^{N} K(\mathbf{x}, \mathbf{x}_n) y_n \Big/ \sum_{n=1}^{N} K(\mathbf{x}, \mathbf{x}_n), \tag{2.2}$$

where $K(\cdot)$ is a kernel function. A natural idea then is to find a weighted subspace parameterized by a nonnegative weight vector $\mathbf{w}$ so that the objective function $\sum_{n=1}^N f(y_n, \hat{y}(\mathbf{x}_n|\mathbf{w}))$ is minimized, where $f(y_n, \hat{y})$ is a cost function, which can be $|y_n - \hat{y}|$ or $(y_n - \hat{y})^2$, and

$$\hat{y}(\mathbf{x}_n|\mathbf{w}) = \frac{\sum_{i=1, i\neq n}^N K(\mathbf{x}_n, \mathbf{x}_i|\mathbf{w}) y_i}{\sum_{i=1, i\neq n}^N K(\mathbf{x}_n, \mathbf{x}_i|\mathbf{w})} \ . \tag{2.3}$$

This is the basic idea of the RGS algorithm proposed in [16] and the objective function is optimized using a gradient descent method. A major issue with the above formulation is that there is no guarantee that an optimal solution can be found due to the presence of local minima.

We develop a new algorithm motivated by the RGS algorithm. The basic idea is to decompose a nonlinear regression problem into a set of classification problems and learn feature relevance within a classification framework. We start by developing a new approach to estimating sample responses and prediction errors. Without loss of generality, we assume that $y_i \geq y_j$ if $i > j$. For every y_n, $2 \leq n \leq N$, we compute $s_n = (y_{n-1} + y_n)/2$ and divide the dataset $\mathcal{D}$ into two subsets $\mathcal{D}_1 = \{\mathbf{x}_i | y_i < s_n, 1 \leq i \leq N\}$ and $\mathcal{D}_2 = \{\mathbf{x}_i | y_i > s_n, 1 \leq i \leq N\}$. Given a sample $\mathbf{x}$, we compute two distances:

$$d_1(y_n) = \min_{\mathbf{z} \in \mathcal{D}_1} d(\mathbf{x}, \mathbf{z}), \ \text{and} \ d_2(y_n) = \min_{\mathbf{z} \in \mathcal{D}_2} d(\mathbf{x}, \mathbf{z}) \ . \tag{2.4}$$

We determine that the response of $\mathbf{x}$ is larger than or equal to y_n if $d_1 > d_2$, and smaller than y_n otherwise. The above described test is repeated starting from y_2 until we find a y_n so that $d_1 \leq d_2$. Then, the response of $\mathbf{x}$ is estimated to be y_{n-1}. Let $\triangle d(y_n) = d_1(y_n) - d_2(y_n)$. It can be proved that $\triangle d(y_n)$ is a monotonically decreasing function of y_n. This means that once we find y_n there is no need to perform additional tests. Also, it is easy to prove that the response y_{n-1} estimated in the above test is equal to $\hat{y}$ estimated in (2.1). However, we will shortly see that the approach we use to estimate sample responses enables us to circumvent the local minimum problem.

Let y be the true response of $\mathbf{x}$, and define $\rho(\mathbf{x}|y_n) = \triangle d(y_n)\text{sign}(y - y_n)$. We define the prediction error as:

$$\epsilon(\mathbf{x}) = \sum_{n=2}^{N} \mathrm{I}(\rho(\mathbf{x}|y_n) < 0) \ , \tag{2.5}$$

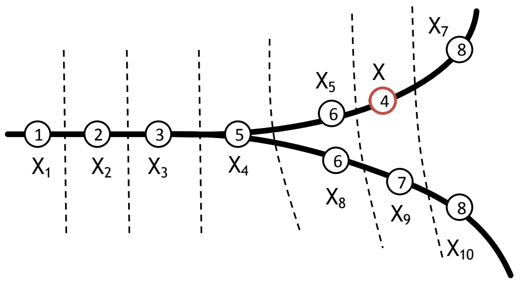

Figure 1: Toy example illustrating the basic idea. The number in a circle is the response of the corresponding sample. In this case, $\hat{y}(\mathbf{x})$ is estimated to be $y(\mathbf{x}_5)$ and $\epsilon(\mathbf{x}) = \sum_{n=2}^{N} \mathrm{I}(\rho(\mathbf{x}|y_n) < 0) = 2$.

where $\mathrm{I}(x < 0)$ is an indicator function that takes the value of 1 if $x < 0$ and 0 otherwise. The above definition can be interpreted within the classification framework: if we successively divide a dataset into two subsets and use the one-nearest-neighbor classifier to classify $\mathbf{x}$ into one of the two groups, $\epsilon(\mathbf{x})$ equals to the number of times when $\mathbf{x}$ is misclassified, and $\rho(\mathbf{x}|y_n)$ can be regarded as a margin of $\mathbf{x}$. This presents a close connection between regression and classification problems. Indeed, a classification problem can be viewed as a degenerate regression problem. Fig. 1 presents a toy example illustrating the basic idea.

Once we define a prediction error function, we proceed to find a weighted feature subspace where the overall prediction error is minimized:

$$\min_{\mathbf{w}\geq 0} \sum_{n=1}^{N}\sum_{i\in\mathcal{C}_n} \mathrm{I}(\rho(\mathbf{x}_n|y_i,\mathbf{w}) < 0)\ , \tag{2.6}$$

where $\mathcal{C}_n = \{i|3 \leq i \leq N\}$ if $n = 1$ and $\mathcal{C}_n = \{i|2 \leq i \leq N, i \neq n\}$ if $n \geq 2$. Since in the inner summation $\mathbf{x}_n$ is held out as a test sample, the above objective function can be interpreted as a leave-one-out cross-validation (LOOCV) error. For numerical convenience, we use the block distance to measure the similarity between two samples, which is also used in the RELIEF [11] and LOGO algorithms [24]. However, other distance functions (e.g., squared Euclidean distance) can also be used. Let $\mathrm{NN}(\mathcal{D}_1)$ and $\mathrm{NN}(\mathcal{D}_2)$ be the nearest neighbors of $\mathbf{x}_n$ in $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. Then, $\rho(\mathbf{x}_n|y_i,\mathbf{w})$ can be computed as a *linear* function of $\mathbf{w}$:

(2.7)
$$\begin{aligned} & \rho(\mathbf{x}_n|y_i,\mathbf{w}) \\ = \;& \mathbf{w}^T\Big(|\mathbf{x}_n - \mathrm{NN}(\mathcal{D}_1)| - |\mathbf{x}_n - \mathrm{NN}(\mathcal{D}_2)|\Big)\mathrm{sign}(y_n - y_i) \\ \triangleq \;& \mathbf{w}^T\mathbf{z}_n(i)\ , \end{aligned}$$

where $|\cdot|$ is an element-wise absolute operator. The problem (2.6) can now be simplified as:

$$\min_{\mathbf{w}\geq 0} \sum_{n=1}^{N}\sum_{i\in\mathcal{C}_n} \mathrm{I}(\mathbf{w}^T\mathbf{z}_n(i) < 0)\ . \tag{2.8}$$

Note that the indicator function is non-differentiable and non-convex. A commonly used practice to address the issue is to minimize the upper bound of a cost function [28]. We use the hinge loss, leading to a SVM formulation of feature selection for nonlinear regression:

(2.9)
$$\min_{\mathbf{w}} \sum_{n=1}^{N}\sum_{i\in\mathcal{C}_n} \max\left(0, 1 - \mathbf{w}^T\mathbf{z}_n(i)\right)\ ,\ \text{s.t. } \|\mathbf{w}\|_1 \leq \lambda, \mathbf{w} \geq 0,$$

where we impose an ℓ_1 penalty on $\mathbf{w}$ in order to obtain a sparse solution, and λ is a regularization parameter controlling the sparseness of a solution. Hence, the algorithm has two levels of regularization, i.e., the implicit LOOCV and explicit ℓ_1 regularization. We will shortly see that the performance of our algorithm is largely insensitive to a specific choice of λ due to the LOOCV regularization (Fig. 5(b)).

There are a number of algorithms that can be used to solve the ℓ_1-SVM problem (e.g., [15]). We demonstrate here that ℓ_1-SVM can be readily solved in its primal domain by using gradient descent techniques. Since the hinge loss is a non-differentiable function, we thus replace it by the Huber loss defined as

$$H(\rho) = \begin{cases} 0 & \rho > 1 + h\ , \\ \dfrac{(1+h-\rho)^2}{4h} & 1 - h \leq \rho \leq 1 + h\ , \\ 1 - \rho & \rho < 1 - h\ , \end{cases} \tag{2.10}$$

where h is a tunable parameter. If h is sufficiently small, SVM using the Huber loss yields the same solution as that obtained with the hinge loss [3].

The problem (2.9) is a constrained convex optimization problem. In order to use gradient descent techniques, traditional methods apply projection or barrier functions to prevent solutions from falling outside feasible regions. In this paper, we use a different approach where we convert the constrained problem into an unconstrained one by setting $w_j = v_j^2$ for $1 \leq j \leq J$. Then, the problem (2.9) with the hinge loss being replaced by the Huber loss can be re-written as

(2.11)
$$\min_{\mathbf{v}} L(\mathbf{v}) = \sum_{n=1}^{N}\sum_{i\in\mathcal{C}_n} H\left(\sum_{j=1}^{J} v_j^2 z_n^j(i)\right) + \alpha\sum_{j=1}^{J} v_j^2\ ,$$

where α is a Lagrange multiplier. Taking the derivatives of L with respect to $\mathbf{v}$ yields

$$(2.12) \qquad dL/d\mathbf{v} = 2\left(\sum_{n=1}^{N}\sum_{i\in\mathcal{C}_n}\frac{dH}{d\rho}\mathbf{z}_n(i) + \alpha\mathbf{1}\right)\odot\mathbf{v},$$

where $\mathbf{1}$ is an all-one vector and $\odot$ is the Hadamard operator. Thus, the problem (2.9) can be solved by using gradient descent with the following updating rule:

$$(2.13) \quad \mathbf{v}^{(k)} = \mathbf{v}^{(k-1)} - \eta\frac{dL}{d\mathbf{v}}|_{\mathbf{v}=\mathbf{v}^{(k-1)}}$$
$$= \left((1-2\eta\alpha)\,\mathbf{1} - 2\eta\sum_{n=1}^{N}\sum_{i\in\mathcal{C}_n}\frac{dH}{d\rho}\mathbf{z}_n(i)\right)\odot\mathbf{v}^{(k-1)},$$

where $\mathbf{v}^{(k)}$ is the solution obtained at the k-th iteration, and η is a learning rate that can be determined through a line search. Note that the objective function of (2.11) is no longer convex, and a gradient descent method may find a local minimizer or a saddle point. However, (2.11) is quasi-convex for $\mathbf{v} \geq 0$, and it can be proved that if the initial point $v_j^{(0)} \neq 0$ for $1 \leq j \leq J$, the solution obtained when the gradient vanishes is a global minimizer [2].

There are two issues associated with the above formulation. The first issue is that although local learning allows us to model complex local data structures, the nearest neighbor of a given sample is unknown before learning. In the presence of many thousands of irrelevant features, which is the case for many biological applications, the nearest neighbors defined in the original space can be completely different from those defined in a weighted space. In order to account for the uncertainty in defining local information, we develop a probabilistic model where the nearest neighbors of a given sample are treated as hidden variables. Following the principles of the expectation-maximization algorithm [5], we estimate $\rho(\mathbf{x}_n|y_i,\mathbf{w})$ by taking expectation via averaging out hidden variables:

$$(2.14) \quad \bar{\rho}(\mathbf{x}_n|y_i,\mathbf{w}) = \mathbb{E}[\rho(\mathbf{x}_n|y_i,\mathbf{w})]$$
$$= \mathbf{w}^T\Big(\mathbb{E}_{j\sim\mathcal{M}_1}[|\mathbf{x}_n-\mathbf{x}_j|]-\mathbb{E}_{j\sim\mathcal{M}_2}[|\mathbf{x}_n-\mathbf{x}_j|]\Big)\mathrm{sign}(y_n-y_i)$$
$$= \mathbf{w}^T\Bigg(\sum_{j\in\mathcal{M}_1}Q(j|n,\mathbf{w})|\mathbf{x}_n-\mathbf{x}_j|$$
$$-\sum_{j\in\mathcal{M}_2}P(j|n,\mathbf{w})|\mathbf{x}_n-\mathbf{x}_j|\Bigg)\mathrm{sign}(y_n-y_i) \triangleq \mathbf{w}^T\bar{\mathbf{z}}_n(i)\,,$$

where $\mathcal{M}_1 = \{j : \mathbf{x}_j \in \mathcal{D}_1\}$, $\mathcal{M}_2 = \{j : \mathbf{x}_j \in \mathcal{D}_2\}$, $\mathbb{E}_{j\sim\mathcal{M}_1}$ is expectation taken with respect to $\mathcal{M}_1$, and $Q(j|n,\mathbf{w})$ and $P(j|n,\mathbf{w})$ are the probabilities of $\mathbf{x}_j$ being the nearest neighbors of $\mathbf{x}_n$ in $\mathcal{D}_1$ and $\mathcal{D}_2$ with respect to $\mathbf{w}$, respectively. The probability $Q(j|n,\mathbf{w})$ can be estimated through a kernel method

$$Q(j|n,\mathbf{w}) = \frac{K(\mathbf{x}_j,\mathbf{x}_n|\mathbf{w})}{\sum_{m\in\mathcal{M}_1}K(\mathbf{x}_m,\mathbf{x}_n|\mathbf{w})}\,,$$

where $K(d)$ is a kernel function. $P(j|n,\mathbf{w})$ can be computed similarly. In this paper, we use the exponential kernel given by $K(d) = \exp(-d/\sigma)$, where kernel width σ determines the resolution at which data is analyzed.

The second issue is that $\bar{\mathbf{z}}_n$ implicitly depends on $\mathbf{w}$ through $P(j|n,\mathbf{w})$ and $Q(j|n,\mathbf{w})$ (see Eq. 2.14). We use a fixed-point recursive method to solve for $\mathbf{w}$. First, we make a guess on a weight vector $\mathbf{w}$ and compute the pairwise distances to estimate $P(j|n,\mathbf{w})$, $Q(j|n,\mathbf{w})$ and $\bar{\mathbf{z}}$, and then update the feature weight vector $\mathbf{w}$ by solving the problem (2.11). The iterations are carried out until convergence.

2.1 Convergence Analysis It can be proved that if the kernel width is properly selected, the algorithm converges to a *unique* solution for any nonnegative initial feature weights, which is stated formally in the following theorem.

THEOREM 2.1. *For the proposed algorithm, there exists σ^* such that $\lim_{t\to+\infty}\|\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}\| = 0$ whenever $\sigma > \sigma^*$, where $\mathbf{w}^{(t)}$ is the feature weight vector learned in the t-th iteration. Moreover, for a fixed kernel width $\sigma > \sigma^*$, the algorithm converges to a unique solution for any nonnegative initial feature weights $\mathbf{w}^{(0)}$.*

Proof. We use the fixed point theorem to prove that our algorithm converges to a unique fixed point. The gist is to identify a contraction operator for the algorithm, and make sure that the conditions of the fixed point theorem are satisfied. To this end, we define $\mathcal{P} = \{\mathbf{p} : \mathbf{p} = [P(j|n,\mathbf{w}), Q(j|n,\mathbf{w})]\}$ and $\mathcal{W} = \{\mathbf{w} : \mathbf{w} \in \mathcal{R}^J, \|\mathbf{w}\|_1 \leq \lambda, \mathbf{w} \geq 0\}$, and specify the first step of the algorithm in a functional form as $T1 : \mathcal{W} \to \mathcal{P}$, where $T1(\mathbf{w}) = \mathbf{p}$, and the second step as $T2 : \mathcal{P} \to \mathcal{W}$, where $T2(\mathbf{p}) = \mathbf{w}$. Then, the algorithm can be written as $\mathbf{w}^{(t)} = (T2\circ T1)(\mathbf{w}^{(t-1)}) \triangleq T(\mathbf{w}^{(t-1)})$, where $\circ$ denotes functional composition and $T : \mathcal{W} \to \mathcal{W}$. Since $\mathcal{W}$ is a closed subset of finite-dimensional normed space $\mathcal{R}^J$ (or a Banach space) and thus complete [12], T is an operator mapping complete subset $\mathcal{W}$ into itself. Next, note that for $\sigma \to +\infty$, $Q(j|n,\mathbf{w}) = 1/|\mathcal{M}_1|$ and $P(j|n,\mathbf{w}) = 1/|\mathcal{M}_2|$, where $|\mathcal{M}_1|$ and $|\mathcal{M}_2|$ are the cardinalities of sets $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. Therefore, $\bar{\mathbf{z}}_n$ is a constant vector independent of $\mathbf{w}$, and the algorithm converges with one iteration. We have $\lim_{\sigma\to+\infty}\|T(\mathbf{w}_1,\sigma) - T(\mathbf{w}_2,\sigma)\| = 0$, for any $\mathbf{w}_1,\mathbf{w}_2 \in \mathcal{W}$. Therefore, in the limit, T is a contraction operator with contraction constant $q = 0$, that is, $\lim_{\sigma \to +\infty} q(\sigma) = 0$. Therefore, for every $\varepsilon > 0$, there exists σ^* such that $q(\sigma) \leq \varepsilon$ whenever $\sigma > \sigma^*$. By setting $\varepsilon < 1$, the resulting operator T is a contraction operator. By the Banach fixed point theorem [12], the algorithm converges to a unique fixed point provided the kernel width is properly selected. The above arguments establish the convergence theorem of the algorithm.

The theorem ensures the convergence of the algorithm if the kernel width is properly selected. This is a very loose condition, as our empirical results show that the algorithm always converges for a sufficiently large kernel width (see Fig. 4(b)). An important implication is that even if the initial feature weights are randomly selected and the algorithm starts computing erroneous nearest neighbors for each sample, the algorithm will eventually converge to the *same* solution obtained as if one had perfect prior knowledge on which features are useful since it is a fixed-point method.

2.2 Computational Complexity The computational complexity of the algorithm is $\mathcal{O}(N^2J)$, where N is the number of samples and J is the data dimensionality. When N is sufficiently large, most CPU time is spent on computing pairwise distances. It is possible to use some recently developed nearest-neighbor-search algorithms to achieve linear or super-linear computational complexity with respect to N. A close look at the updating equation (2.13) allows us to further reduce complexity. If some elements of $\mathbf{v}$ are close to zero (say $< 10^{-5}$), the corresponding features can be eliminated from further consideration with a negligible impact on final solutions, thereby providing a built-in mechanism for automatically removing irrelevant features during the learning process.

3 Previous Work

We present a brief review of four state-of-the-art algorithms, namely HSIC Lasso, SpAM, RGS and HMKL, that we compare with in a numerical study.

The recently developed HSIC Lasso algorithm [27] is a filter method that solves a feature-wise Lasso problem and selects features based on the empirical Hilbert-Schmidt independence criterion (HSIC) by using the following formulation:

(3.15)
$$\min_{\mathbf{w}} \|\bar{\mathbf{Y}} - \sum_{j=1}^{J} w_j \bar{\mathbf{K}}_j\|_F^2 + \lambda \|\mathbf{w}\|_1, \text{ subject to } \mathbf{w} \geq \mathbf{0},$$

where $\|\cdot\|_F$ is the Frobenius norm, $\bar{\mathbf{K}}_j = \mathbf{L}\mathbf{K}_j\mathbf{L}$ and $\bar{\mathbf{Y}} = \mathbf{L}\mathbf{Y}\mathbf{L}$ are centered Gram matrices, $\mathbf{K}_j(n,m) = K(x_n(j), x_m(j))$ and $\mathbf{Y}(n,m) = K(y_n, y_m)$ are Gram matrices, $K(x,x')$ is a kernel function, $x_n(j)$ is the j-th element of $\mathbf{x}_n$, and $\mathbf{L} = \mathbf{I}_N - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T$ is a centering matrix. It is reported that HSIC Lasso compares favorably with existing methods including mRMR [19], FVM [13] and QPFS [21]. As showed in (3.15), a naive implementation of the algorithm requires extremely large memory usage since it needs to generate J matrices of size $N \times N$ before learning. To address this issue, the authors propose a lookup table-based method to deal with the scenario with a large sample size. However, since HSICLasso is a filter method, the selected features may not directly relate to learning accuracy.

The second algorithm we compare with is SpAM [20], which performs non-parametric regression and feature selection simultaneously by solving the following optimization problem:

(3.16)
$$\min_{\{\boldsymbol{\beta}_j\}} \|\mathbf{y} - \sum_{j=1}^{J} \boldsymbol{\Psi}_j \boldsymbol{\beta}_j\|_2^2 + \lambda \sum_{j=1}^{J} \sqrt{\frac{1}{N}\boldsymbol{\beta}_j \boldsymbol{\Psi}^T \boldsymbol{\Psi} \boldsymbol{\beta}_j},$$

where $\mathbf{y}(n) = y_n$, $\boldsymbol{\Psi}_j(n,l) = \Psi_l(x_n(j))$, and Ψ_l is the l-th basis function. One major drawback of SpAM is its additive model assumption. It may perform poorly when there exist interactions among features [27].

The third algorithm we compare with is HMKL [1], which embeds kernels in a direct acyclic graph (DAG) and selects kernels based on some heuristics by solving the following optimization problem:

$$(3.17) \quad \min_{\boldsymbol{\beta} \in \Pi_{v \in \mathcal{V}} \mathcal{F}_v} \frac{1}{N} \sum_{n=1}^{N} f\left(y_n, \sum_{v \in \mathcal{V}} \langle \boldsymbol{\beta}_v, \Phi_v(\mathbf{x}_n) \rangle\right) + \frac{\lambda}{2}\left(\sum_{v \in \mathcal{V}} d_v \|\boldsymbol{\beta}_{D(v)}\|\right)^2$$

where $\mathcal{V}$ is an index set of basis kernels k_v, $v \in \mathcal{V}$, and for each $v \in \mathcal{V}$, $\mathcal{F}_v$ and Φ_v are the feature space and feature map of k_v, respectively. $D(v)$ represents the descendent set of a given node v in the DAG and $(d_v)_{v \in \mathcal{V}}$ are positive weights. $\sum_{v \in \mathcal{V}} d_v \|\boldsymbol{\beta}_{D(v)}\| = \sum_{v \in \mathcal{V}} d_v \sqrt{\sum_{w \in D(v)} \|\boldsymbol{\beta}_w\|^2}$ is a structured block ℓ_1-norm to set some elements of vector $\boldsymbol{\beta}_v$ exactly zero in the solution. The main drawback of HMKL is its high computational complexity, which is also noted by the authors of the algorithm [1]. The time complexity of the algorithm is $O(N^3R + N^2RJ^2 + N^2R^2J)$, where N, J and R are the number of samples, features and selected kernels, respectively. The fourth algorithm we consider is RGS, which is discussed in Section 2.

4 Experimental Results

4.1 Synthetic Data Before applying the new algorithm to real data, we first perform a simulation study on four synthetic datasets. The first dataset is generated from an additive model given by

(4.18)
$$Y = -2\sin(2X_1) + X_2^2 + X_3 + \exp(-X_4) + \mathcal{N}(0,1),$$

where $\{X_j\}_{j=1}^4 \sim \mathcal{N}(0,1)$ are independently drawn from a Gaussian distribution with zero mean and unit variance. The second dataset is generated from a non-additive model:

$$Y = X_1 \exp(2X_2) + X_3^2 + \mathcal{N}(0,1), \quad (4.19)$$

where $\{X_j\}_{j=1}^3 \sim \mathcal{N}(0,1)$. The two datasets are also used in [27]. The third dataset is generated from a sine model representing the case where data has a weak linear dependency with responses:

$$Y = \sin(2\pi X) + \mathcal{N}(0, 0.1), \quad (4.20)$$

where $X \sim \mathcal{U}(0,4)$ is independently drawn from a uniform distribution $[0,4]$. The forth dataset is generated from a two-dimensional spiral model:

(4.21)
$$X_1 = Y\sin(Y) + \mathcal{N}(0,1), X_2 = Y\cos(Y) + \mathcal{N}(0,1),$$

where $Y \sim \mathcal{U}(0,20)$. For each dataset, the set of original features is augmented by 1000 irrelevant features independently sampled from $\mathcal{N}(0,1)$. Our goal is to detect relevant features to recover true signals that are completely buried in random noise.

The codes of RGS, SpAM, HSIC Lasso, HMKL are downloaded from the authors' websites, and the default parameters are used. For HSIC Lasso and HMKL, one needs to specify a regularization parameter. We run the two algorithms multiple times for each dataset using different parameters ($[1, 10, 20, \cdots, 100]$ for HSIC Lasso and $[10, 10^0, \cdots, 10^{-7}]$ for HMKL), and report the best result. The regularization parameter of SpAM is estimated by using the C_p statistics given in [20]. For our method, we simply set the kernel width $\sigma = 1$ and the regularization parameter $\lambda = 1$. Before learning, we scale the values of each feature into $[0,1]$ so that they are comparable, and no other preprocessing is performed. We apply the five methods to each dataset, rank the resulting feature weights in a descending order. If there are d useful features, the probability of correct recovery is defined as the fraction of the useful features detected in the top d features. This criterion is also used in [27, 20]. The experiment is repeated 50 times. HMKL is computationally very expensive so we run the algorithm only 10 times on a computer cluster.

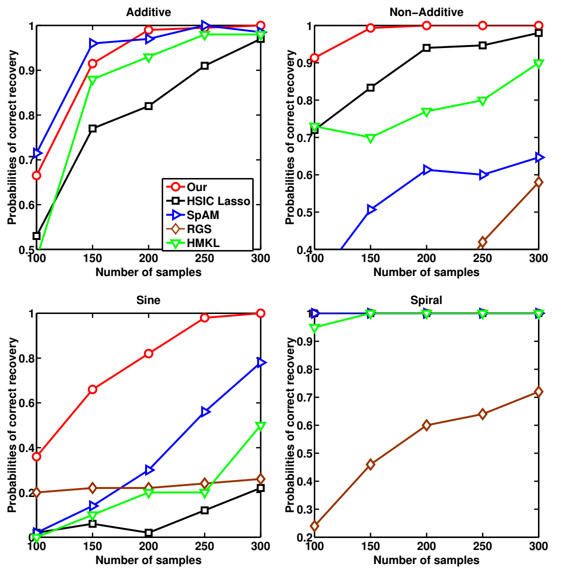

Figure 2: Probabilities of correct recovery of five algorithms applied to four datasets with 1000 irrelevant features. The probabilities of correct recovery of RGS are close to zero for the additive data and thus omitted.

Fig. 2 reports the probabilities of correct recovery of the five algorithms as a function of the number of samples ranging from 100 to 300, averaged over 50 runs (10 runs for HMKL). RGS performs poorly on all four datasets. One possible reason is that with the increase of data dimensionality, the chance of RGS being trapped by local minima is increased exponentially. To confirm this, we repeat the experiment by applying RGS to data with only 100 irrelevant features. RGS performs much better though still much worse than others, which suggests that RGS is not suitable for high-dimensional data analysis. The performance of HSIC Lasso on the additive and non-additive data is similar to that reported in [27]. HSIC Lasso yields a perfect result on the complex spiral data, but fails on the sine data. SpAM performs extremely well on the additive dataset. This is not surprising since it is designed specifically for handling data generated by an additive model. However, SpAM performs poorly on both non-additive and sine data. In contrast, our method does not make any model assumption. It performs perfectly on the spiral data and comparably with SpAM on the additive data, and outperforms the four competing methods by a large margin in the other two datasets. Fig. 3 plots the feature weights generated by our algorithm for the four datasets with 200 samples. It detects all useful features and removes nearly all irrelevant ones. Fig. 4(a) reports the CPU times of the five algorithms applied to the additive dataset with 100 samples contaminated by a varying number of irrelevant features ranging from 200

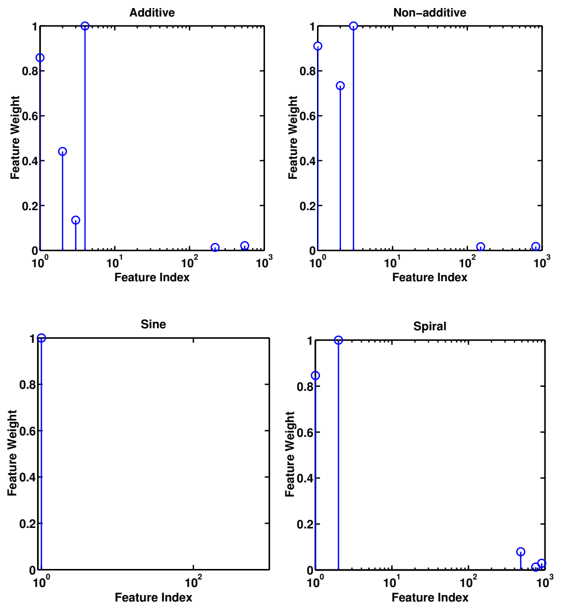

Figure 3: Feature weights generated by our algorithm applied to the four datasets with 200 samples each containing 1000 irrelevant features. Our algorithm removes nearly all irrelevant features.

to 1000. Our method is computationally slightly more expensive than HSIC Lasso and SpAM, but is much more efficient than HMKL.

We perform some additional simulation studies to demonstrate various properties of the algorithm. Fig. 4(b) reports the results of a convergence analysis of our algorithm applied to the non-additive data with 1000 irrelevant features, using $\lambda = 1$ and different kernel widths ranging from 0.1 to 5. The algorithm converges for a wide range of kernel width values, suggesting that our algorithm generally has no divergence issue. This result empirically verifies Theorem 2.1.

The kernel width and regularization parameter are two input parameters of the algorithm. In Fig. 5, we plot the feature weights learned on the non-additive data with 1000 irrelevant features by using different kernel widths and regularization parameters. The algorithm performs well over a wide range of parameter values, suggesting that the performance of our algorithm is largely insensitive to the choice of the parameters.

4.2 Cancer Gene Expression Data We next apply our method to nine cancer microarray gene expression datasets. The datasets are downloaded from European Genome-Phenome Archive (EGA) and Gene Expression Omnibus (GEO), including four breast cancer datasets (BRCA1-4), one lung cancer dataset (LUAD), one glioblastoma (GBM) dataset, one stomach cancer dataset (STAD), and two diffuse large B-cell lymphoma

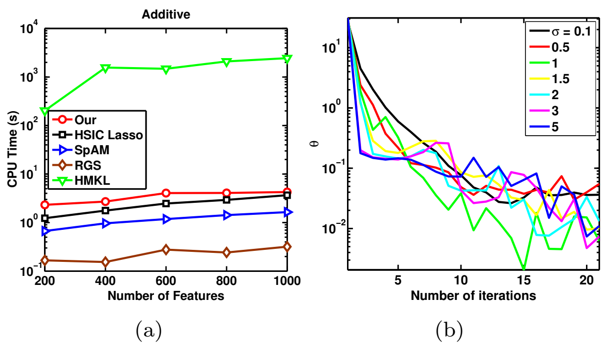

Figure 4: (a) CPU times of five algorithms applied to the additive dataset with a varying number of irrelevant features. (b) Convergence analysis of our algorithm performed on the non-additive dataset with 1000 irrelevant features, using $\lambda = 1$ and different kernel widths ranging from 0.1 to 5, where $\theta = \|\mathbf{w}^{(t)} - \mathbf{w}^{(t-1)}\|_2$.

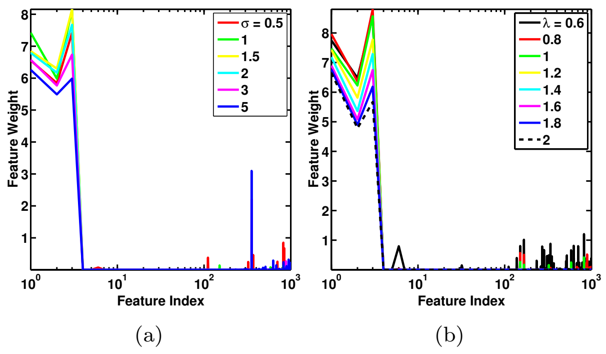

Figure 5: Feature weights learned on the non-additive data with 1000 irrelevant features. Only the first three features are relevant. (a) Using a fixed regularization parameter $\lambda = 1$ and different kernel widths. (b) Using a fixed kernel width $\sigma = 1$ and different regularization parameters. The performance of our algorithm is largely insensitive to the choice of the parameters.

(DLBCL1-2) datasets (Table 1).

We do not use RGS in the experiment as we have shown that RGS does not perform well for high-dimensional data. HMKL is also not suitable for this experiment due to its high computational complexity (see Fig. 4(a)). In order to justify the use of a nonlinear model, we also compare our method with Lasso [25]. Since the original implementation of HSIC Lasso requires a memory size that grows quadratically with the number of samples, we use the lookup table-based implementation downloaded from the authors' website. The kernel width of our method is set to 2 for all datasets, and the regularization parameters of all methods are estimated through ten-fold cross validation. In order to make all features comparable and remove outlier data, we apply robust linear scaling to each gene so that the expression quantiles 2% and 98%

Table 1: Microarray data used in the experiment.

Dataset	# of Samples	# of Features	Response[†]	Accession #
BRCA1	1147	41344	DFS	EGAS00000000083
BRCA2	171	22283	DMFS	GSE2034/5327
BRCA3	185	44928	RFS	GSE3494
BRCA4	185	54675	DMFS	GSE12276
LUAD	218	54675	OS	GSE30219
GBM	176	22283	OS	GSE13041
STAD	207	17418	RFS	GSE26253
DLBCL1	137	24526	DFS	GSE32918
DLBCL2	183	54675	OS	GSE10846

[†] Used response data: disease-free survival (DFS), distant-metastasis-free survival (DMFS), relapse-free survival (RFS) and overall survival (OS)

are set to 0 and 1, respectively. No other preprocessing is performed.

Two different criteria are used to evaluate the performance of the four methods. First, we compare the prediction accuracy of regression analysis performed on the features selected by the four methods. The goal is to identify a cancer-associated gene profile to build a computational model to predict patient clinical outcomes for disease prognosis. To this end, we first randomly partition a dataset into two sub-datasets, one with 80% samples for training and one with the remaining 20% samples for testing. We then apply each method to the training dataset to identify a list of relevant features and construct a prediction model which is then tested blindly on the test dataset. Both SpAM and Lasso can perform feature selection and prediction simultaneously. In order to use the features selected by our method and HSIC Lasso to predict clinical outcomes, the Nadaraya-Watson method is used. The experiment is repeated ten times for each dataset. Table 2 presents the averaged prediction errors of the four methods. The best result and the results that are not significantly worse than the best one are highlighted in bold (p-value<0.05, Wilcoxon rank-sum test). From the table, we can see that our method yields the smallest prediction errors and performs significantly better than all the other methods for most of the datasets. Lasso performs the worst among the four methods. This may be due to its linear assumption that fails to capture the nonlinear dependence of the survival time on patients' gene expression profiles. SpAM performs slightly better than Lasso. One possible reason is that SpAM assumes an additive model and thus ignores the possible interactions among genes. HSIC Lasso performs much better than Lasso and SpAM and exhibits no significant difference from our method in three datasets (BRAC4, GBM, and STAD). However, possibly due to the fact that HSIC Lasso does not take prediction errors into account in feature selection, it does not perform as well as our method in the other six datasets.

In addition to using gene expression profiles to predict patient clinical outcomes, molecular subtyping is another major line of cancer research [4, 22, 14]. The majority of the work is performed on breast cancer, and the pioneering studies by [22] have showed that breast cancer is not a single disease but consists of at least five genetically heterogenous diseases. Thus, we next examine whether the features selected by the four methods enable us to identify cluster structures consistent with those reported in the literature. Specifically, we perform spectral clustering to detect genetically homogenous groups based on the profiles of the selected genes, and then compare the clustering results with breast cancer molecular subtypes. The standard normalized spectral clustering method [17] is used and the number of clusters is estimated by using the method proposed by [30]. We consider only the BRCA1 dataset as it contains a much larger number of samples than other three breast cancer datasets. Note that there are currently no widely accepted molecular subtyping methods [14]. We thus compare with seven major molecular subtyping methods developed in the last decade, including SSP2003 [22], SSP2006 [10], PAM50 [18], SCMOD1 [6], SCMOD2 [29], SCMGENE [8] and IntClust [4]. We use normalized mutual information (NMI) and adjusted rand index (ARI), the two most commonly used evaluation metrics in the machine learning community, to measure the concordance of the clustering results generated by two different algorithms. Fig. 6 reports the NMI and ARI scores of the four methods. We can see that our algorithm performs significantly better than all the other three algorithms (p-value<0.0, one-sided Wilcoxon signed rank test). This result suggests that the genes selected by our method more accurately reflect the underlying biological processes of cancer development than the other methods.

Table 2: Prediction performance of four methods measured in absolute errors (years). The best result and the results that are not significantly worse than the best one are highlighted in bold (p-value<0.05, Wilcoxon rank-sum test). Standard errors are listed in parentheses.

Dataset	Lasso	SpAM	HSIC Lasso	Ours
BRCA1	3.93 (0.29)	3.93 (0.25)	3.82 (0.22)	**3.28 (0.21)**
BRCA2	4.03 (0.41)	4.12 (0.39)	3.74 (0.34)	**2.81 (0.58)**
BRCA3	2.27 (0.31)	2.15 (0.20)	1.76 (0.48)	**0.97 (0.32)**
BRCA4	1.01 (0.13)	1.00 (0.17)	**0.88 (0.15)**	**0.80 (0.12)**
LUAD	3.07 (0.47)	3.48 (0.33)	2.84 (0.27)	**2.41 (0.32)**
GBM	**0.62 (0.11)**	**0.65 (0.14)**	**0.61 (0.08)**	**0.59 (0.06)**
STAD	2.03 (0.23)	1.90 (0.35)	**1.52 (0.16)**	**1.36 (0.20)**
DLBCL1	0.96 (0.23)	0.96 (0.14)	0.60 (0.11)	**0.31 (0.16)**
DLBCL2	2.00 (0.19)	1.88 (0.29)	1.72 (0.33)	**0.96 (0.14)**

5 Conclusion

In this paper, we developed a new feature-selection method for nonlinear regression. The proposed method does not explicitly impose any model assumption on

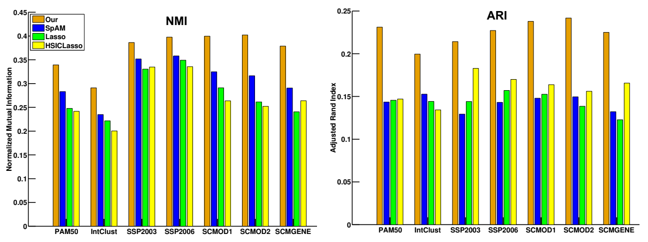

Figure 6: NMI and ARI scores of four methods obtained by comparing with seven existing breast cancer molecular subtyping methods.

data distribution, and is able to select relevant features supporting complex data structure hidden in a high-dimensional space. We demonstrated the effectiveness and utilities of the new method by applying it to a set of simulation and cancer transcriptomic datasets. As currently there are no reliable methods for cancer prognosis and molecular subtyping, the developed methods could be used to identify cancer-associated gene expression profiles to build improved prediction models.

Acknowledgements

This work was supported in part by the National Science Foundation under grant number 1322212 and the SUNY Research Foundation.

References

[1] F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *NIPS*, 2008.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, 2004.

[3] O. Chapelle. Training a support vector machine in the primal. *Neural Comput*, 19(5):1155–1178, 2007.

[4] C. Curtis, S. P. Shah, S.-F. Chin, et al. The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature*, 486(7403):346–352, 2012.

[5] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc Series B*, 39(1):1–38, 1977.

[6] C. Desmedt, B. Haibe-Kains, P. Wirapati, et al. Biological processes associated with breast cancer clinical outcome depend on the molecular subtypes. *Clin Cancer Res*, 14(16):5158–5165, 2008.

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

[8] B. Haibe-Kains, C. Desmedt, S. Loi, et al. A three-gene model to robustly identify breast cancer molecular subtypes. *J Natl Cancer Inst*, 104(4):311–325, 2012.

[9] T. Hastie, R. Tibshirani, J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2009.

[10] Z. Hu, C. Fan, D. Oh, et al. The molecular portraits of breast tumors are conserved across microarray platforms. *BMC Genomics*, 7(1):96, 2006.

[11] K. Kira and L. Rendell. A practical approach to feature selection. In *ICML*, 1992.

[12] R. Kress. *Numerical Analysis*. Springer, New York, 1998.

[13] F. Li, Y. Yang, and E. P. Xing. From lasso regression to feature vector machine. In *NIPS*, 2005.

[14] A. Mackay, B. Weigelt, A. Grigoriadis, et al. Microarray-based class discovery for molecular classification of breast cancer: Analysis of interobserver agreement. *J Natl Cancer Inst*, 103(8):662–673, 2011.

[15] O. Mangasarian. Exact 1-norm support vector machines via unconstrained convex differentiable minimization. *J Mach Learn*, 7(2):1517–1530, 2006.

[16] A. Navot, L. Shpigelman, N. Tishby, and E. Vaadia. Nearest neighbor based feature selection for regression and its application to neural activity. In *NIPS*, 2005.

[17] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *NIPS*, 2001.

[18] J. S. Parker, M. Mullins, M. C. Cheang, et al. Supervised risk predictor of breast cancer based on intrinsic subtypes. *J Clin Oncol*, 27(8):1160–1167, 2009.

[19] H. Peng, F. Long, and C. Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans Pattern Anal Mach Intell*, 27(8):1226–1238, 2005.

[20] P. Ravikumar, H. Liu, J. Lafferty, and L. Wasserman. SpAM: Sparse additive models. In *NIPS*, 2007.

[21] I. Rodriguez-Lujan, R. Huerta, C. Elkan, and C. S. Cruz. Quadratic programming feature selection. *J Mach Learn*, 11:1491–1516, 2010.

[22] T. Sørlie, R. Tibshirani, J. Parker, et al. Repeated observation of breast tumor subtypes in independent gene expression data sets. *Proc Natl Acad Sci*, 100(14):8418–8423, 2003.

[23] Y. Sun. Iterative RELIEF for feature weighting: algorithms, theories, and applications. *IEEE Trans Pattern Anal Mach Intell*, 29(6):1035–1051, 2007.

[24] Y. Sun, S. Todorovic, and S. Goodison. Local-learning-based feature selection for high-dimensional data analysis. *IEEE Trans Pattern Anal Mach Intell*, 32(9):1610–1626, 2010.

[25] R. Tibshirani. Regression shrinkage and selection via the lasso. *J R Stat Soc Series B*, 58(1):267–288, 1996.

[26] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight. Sparsity and smoothness via the fused lasso. *J R Stat Soc Series B*, 67(1):91–108, 2005.

[27] M. Yamada, W. Jitkrittum, L. Sigal, et al. High-dimensional feature selection by feature-wise kernelized lasso. *Neural Comput*, 26(1):185–207, 2014.

[28] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 2000.

[29] P. Wirapati, C. Sotiriou, S. Kunkel, et al. Meta-analysis of gene expression profiles in breast cancer: toward a unified understanding of breast cancer subtyping and prognosis signatures. *Breast Cancer Res*, 10(4):R65, 2008.

[30] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, 2004.