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 [23], 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 closest to a straightforward approach is to find a sample regression problem for the moment. Given a sample

\[
\begin{align*}
\sum_{n=1}^{N} K(x, x_n) y_n / \sum_{n=1}^{N} K(x, x_n),
\end{align*}
\]

where \( K(\cdot) \) is a kernel function. A natural idea then is to find a weighted subspace parameterized by a non-negative weight vector \( w \) so that the objective function

\[
\sum_{n=1}^{N} f(y_n, \hat{y}(x_n|w))
\]

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

\[
\hat{y}(x_n|w) = \sum_{i=1,i\neq n}^{N} K(x_n, x_i|w)y_i / \sum_{i=1,i\neq n}^{N} K(x_n, x_i|w).
\]

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 \( D \) into two subsets \( D_1 = \{x_i|y_i < s_n, 1 \leq i \leq N\} \) and \( D_2 = \{x_i|y_i > s_n, 1 \leq i \leq N\} \). Given a sample \( x \), we compute two distances:

\[
d_1(y_n) = \min_{z \in D_1} d(x, z), \quad \text{and} \quad d_2(y_n) = \min_{z \in D_2} d(x, z).
\]

We determine that the response of \( x \) 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 \( x \) is estimated to be \( y_{n-1} \). Let \( \Delta d(y_n) = d_1(y_n) - d_2(y_n) \). It can be proved that \( \Delta d(y_n) \) 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 \( x \), and define \( \rho(x|y_n) = \Delta d(y_n) \text{sign}(y - y_n) \). We define the prediction error as:

\[
\epsilon(x) = \sum_{n=2}^{N} I(\rho(x|y_n) < 0),
\]
where $I(x < 0)$ 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 the data into one of the two groups, $\epsilon(x)$ equals to the number of times when $x$ is misclassified, and $\rho(x|y_n)$ can be regarded as a margin of $x$. 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:

\begin{equation}
(2.6) \quad \min_{w \geq 0} \sum_{n=1}^{N} \sum_{i \in C_n} I(\rho(x_n|y_i, w) < 0),
\end{equation}

where $C_n = \{i|3 \leq i \leq N\}$ if $n = 1$ and $C_n = \{i|2 \leq i \leq N, i \neq n\}$ if $n \geq 2$. Since in the inner summation $x_n$ 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 NN($D_1$) and NN($D_2$) be the nearest neighbors of $x_n$ in $D_1$ and $D_2$, respectively. Then, $\rho(x_n|y_i, w)$ can be computed as a linear function of $w$:

\begin{equation}
(2.7) \quad \rho(x_n|y_i, w) = w^T \left( |x_n - \text{NN}(D_1)| - |x_n - \text{NN}(D_2)| \right) \text{sign}(y_n - y_i) \triangleq w^T z_n(i),
\end{equation}

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

\begin{equation}
(2.8) \quad \min_{w \geq 0} \sum_{n=1}^{N} \sum_{i \in C_n} I(w^T z_n(i) < 0).
\end{equation}

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:

\begin{equation}
(2.9) \quad \min_{w} \sum_{n=1}^{N} \sum_{i \in C_n} \max(0, 1 - w^T z_n(i)), \; \text{s.t.} \; \|w\|_1 \leq \lambda, w \geq 0,
\end{equation}

where we impose an $\ell_1$ penalty on $w$ in order to obtain a sparse solution, and $\lambda$ 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 $\ell_1$ regularization. We will shortly see that the performance of our algorithm is largely insensitive to a specific choice of $\lambda$ due to the LOOCV regularization (Fig. 5(b)).

There are a number of algorithms that can be used to solve the $\ell_1$-SVM problem (e.g., [15]). We demonstrate here that $\ell_1$-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:

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

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

\begin{equation}
(2.11) \quad \min_{v} L(v) = \sum_{n=1}^{N} \sum_{i \in C_n} H \left( \sum_{j=1}^{J} v_j^2 z_n^2(i) \right) + \alpha \sum_{j=1}^{J} v_j^2,
\end{equation}

where $\alpha$ is a Lagrange multiplier. Taking the derivatives
of \( L \) with respect to \( v \) yields
\[
(2.12) \quad \frac{dL}{dv} = 2 \left( \sum_{n=1}^{N} \sum_{i \in C_n} \frac{dH}{dp} z_n(i) + \alpha 1 \right) \odot v,
\]
where \( 1 \) 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:
\[
(2.13) \quad v^{(k)} = v^{(k-1)} - \eta \frac{dL}{dv} \bigg|_{v=v^{(k-1)}} = \left( (1 - 2\eta \alpha) 1 - 2\eta \sum_{n=1}^{N} \sum_{i \in C_n} \frac{dH}{dp} z_n(i) \right) \odot v^{(k-1)},
\]
where \( v^{(k)} \) is the solution obtained at the \( k \)-th iteration, and \( \eta \) 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 \( v \geq 0 \), 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 \([3]\), we estimate \( \rho(x_n|y_i, w) \) by taking expectation over averaging out hidden variables:
\[
(2.14) \quad \hat{\rho}(x_n|y_i, w) = \mathbb{E}[\rho(x_n|y_i, w)] = w^T \left( \mathbb{E}_{j \sim \mathcal{M}_1} ||x_n - x_j|| - \mathbb{E}_{j \sim \mathcal{M}_2} ||x_n - x_j|| \right) \text{sign}(y_n - y_i) = w^T \sum_{j \in \mathcal{M}_1} Q(j|n,w)|x_n - x_j| - \sum_{j \in \mathcal{M}_2} P(j|n,w)|x_n - x_j| \text{sign}(y_n - y_i) \triangleq w^T \tilde{z}_n(i),
\]
where \( \mathcal{M}_1 = \{ j : x_j \in \mathcal{D}_1 \}, \mathcal{M}_2 = \{ j : x_j \in \mathcal{D}_2 \}, \mathbb{E}_{j \sim \mathcal{M}_1} \) is expectation taken with respect to \( \mathcal{M}_1 \), and \( Q(j|n,w) \) and \( P(j|n,w) \) are the probabilities of \( x_j \) being the nearest neighbors of \( x_n \) in \( \mathcal{D}_1 \) and \( \mathcal{D}_2 \) with respect to \( w \), respectively. The probability \( Q(j|n,w) \) can be estimated through a kernel method
\[
Q(j|n,w) = \frac{K(x_j,x_n|w)}{\sum_{m \in \mathcal{M}_1} K(x_m,x_n|w)},
\]
where \( K(d) \) is a kernel function. \( P(j|n,w) \) can be computed similarly. In this paper, we use the exponential kernel given by \( K(d) = \exp(-d/\sigma) \), where kernel width \( \sigma \) determines the resolution at which data is analyzed.

The second issue is that \( \tilde{z}_n \) implicitly depends on \( w \) through \( P(j|n,w) \) and \( Q(j|n,w) \) (see Eq. \(2.14\)). We use a fixed-point recursive method to solve for \( w \). First, we make a guess on a weight vector \( w \) and compute the pairwise distances to estimate \( P(j|n,w) \), \( Q(j|n,w) \) and \( \tilde{z} \), and then update the feature weight vector \( w \) by solving the problem \((2.9)\). 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 \( \sigma^* \) such that \( \lim_{t \to +\infty} ||w^{(t)} - w^{(t-1)}|| = 0 \) whenever \( \sigma > \sigma^* \), where \( w^{(t)} \) 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 \( w^{(0)} \).

**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 \( \mathcal{P} = \{ p : p = [P(j|n,w),Q(j|n,w)] \} \) and \( \mathcal{W} = \{ w : w \in \mathcal{R}^J, ||w||_1 \leq \lambda,w \geq 0 \} \), and specify the first step of the algorithm in a functional form as \( T1 : \mathcal{W} \to \mathcal{P} \), where \( T1(w) = p \), and the second step as \( T2 : \mathcal{P} \to \mathcal{W} \), where \( T2(p) = w \). Then, the algorithm can be written as \( w^{(t)} = \tilde{T}(T2 \circ T1)(w^{(t-1)}) \triangleq T(w^{(t-1)}) \), 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 \( \sigma \to +\infty \), \( Q(j|n,w) = 1/|\mathcal{M}_1| \) and \( P(j|n,w) = 1/|\mathcal{M}_2| \), where \( |\mathcal{M}_1| \) and \( |\mathcal{M}_2| \) are the cardinalities of sets \( \mathcal{M}_1 \) and \( \mathcal{M}_2 \), respectively. Therefore, \( \tilde{z}_n \) is a constant vector independent of \( w \), and the algorithm converges with one iteration. We have \( \lim_{\sigma \to +\infty} ||T(w_1,\sigma) - T(w_2,\sigma)|| = 0 \), for any \( w_1, w_2 \in \mathcal{W} \). 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 \( \sigma^* \) 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. [1](b)). An important implication is that even if the initial feature weights are randomly sampled, 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 \( \mathcal{O}(N^2 J) \), where \( N \) is the number of samples and \( J \) is the number of features 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 \( v \) are close to zero (say \( \varepsilon < 10^{-5} \)), 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:

\[
\min_{w} \| \mathbf{Y} - \sum_{j=1}^{J} w_{j} \mathbf{K}_{j} \|_{F}^{2} + \lambda \| w \|_{1}, \text{ subject to } w \geq 0,
\]

where \( \| \cdot \|_{F} \) is the Frobenius norm, \( \mathbf{K}_{j} = \mathbf{L} \mathbf{K}_{j} \mathbf{L} \) and \( \mathbf{Y} = \mathbf{L} \mathbf{Y} \mathbf{L} \) 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 \( x_{n} \), 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:

\[
\min_{\beta} \| \mathbf{y} - \sum_{j=1}^{J} \mathbf{Y}_{j} \mathbf{v}_{j} \|_{2}^{2} + \lambda \sum_{j=1}^{J} \sqrt{\frac{1}{N} \mathbf{Y}_{j}^{T} \mathbf{Y}_{j}} \mathbf{v}_{j},
\]

where \( \mathbf{Y}(n) = y_{n}, \mathbf{Y}_{j}(n, l) = \Psi_{j}(x_{n}(j)), \) and \( \Psi_{j} \) 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:

\[
\min_{\beta} \sigma \in \Pi_{v \in V} \mathbf{F}_{v} \frac{1}{N} \sum_{n=1}^{N} \sum_{v \in V} \left( y_{n} \sum_{v \in V} \langle \beta_{v}, \Phi_{v}(x_{n}) \rangle \right) + \frac{\lambda}{2} \left( \sum_{v \in V} d_{v} \| \beta_{D(v)} \|_{1} \right)^{2}
\]

where \( V \) is an index set of basis kernels \( k_{v}, v \in V \), and for each \( v \in V, \mathbf{F}_{v}, \) and \( \Phi_{v} \) are the feature space and feature map of \( k_{v} \), respectively. \( D(v) \) represents the descendnet set of a given node \( v \) in the DAG and \( (d_{v})_{v \in V} \) are positive weights. \( \sum_{v \in V} d_{v} \| \beta_{D(v)} \|_{1} = \sum_{v \in V} d_{v} \sqrt{\sum_{w \in D(v)} \| \beta_{w} \|_{2}^{2}} \) is a structured block \( \ell_{1} \)-norm to set some elements of vector \( \beta \) 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 \( \mathcal{O}(N^{3} R + N^{4} R^{2} J + N^{4} R^{2} J) \), 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

\[ Y = -2\sin(2X_1) + X_2^2 + X_3 + \exp(-X_4) + N(0,1), \]

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

\[ Y = X_1 \exp(2X_2) + X_3^2 + N(0,1), \]

where \( \{X_j\}_{j=1}^3 \sim N(0,1) \). 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:

\[ Y = \sin(2\pi X) + N(0,0.1), \]

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

\[ X_1 = Y \sin(Y) + N(0,1), \quad X_2 = Y \cos(Y) + N(0,1), \]

where \( Y \sim U(0,20) \). For each dataset, the set of original features is augmented by 1000 irrelevant features independently sampled from \( N(0,1) \). 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 \([10, 10^3, \cdots , 10^7]\) 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.

Fig. 2 reports the probabilities of correct recovery of the 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.

---

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.
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 (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%
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). Standard errors are listed in parentheses.

Table 1: Microarray data used in the experiment.

<table>
<thead>
<tr>
<th>Dataset</th>
<th># of Samples</th>
<th># of Features</th>
<th>Response</th>
<th>Accession #</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRCA1</td>
<td>1147</td>
<td>4134</td>
<td>DFS</td>
<td>EGAS00000000083</td>
</tr>
<tr>
<td>BRCA2</td>
<td>171</td>
<td>22283</td>
<td>DMFS</td>
<td>GSE2034/5327</td>
</tr>
<tr>
<td>BRCA3</td>
<td>185</td>
<td>44928</td>
<td>RFS</td>
<td>GSE3491</td>
</tr>
<tr>
<td>BRCA4</td>
<td>185</td>
<td>54675</td>
<td>DMFS</td>
<td>GSE12276</td>
</tr>
<tr>
<td>LUAD</td>
<td>218</td>
<td>54675</td>
<td>OS</td>
<td>GSE0219</td>
</tr>
<tr>
<td>GBM</td>
<td>176</td>
<td>22283</td>
<td>OS</td>
<td>GSE13041</td>
</tr>
<tr>
<td>STAD</td>
<td>207</td>
<td>17418</td>
<td>RFS</td>
<td>GSE26253</td>
</tr>
<tr>
<td>DLBCL1</td>
<td>137</td>
<td>24526</td>
<td>DFS</td>
<td>GSE32918</td>
</tr>
<tr>
<td>DLBCL2</td>
<td>183</td>
<td>54675</td>
<td>OS</td>
<td>GSE10846</td>
</tr>
</tbody>
</table>

1Used response data: disease-free survival (DFS), distant-metastasis-free survival (DMFS), relapse-free survival (RFS) and overall survival (OS).

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.

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

Another major line of cancer research [3, 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 homogeneous 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 used 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.

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
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