Variable Selection via Penalized Neural Network: a Drop-Out-One Loss Approach

Mao Ye [*1] **Yan Sun** [*1]

Abstract

We propose a variable selection method for high dimensional regression models, which allows for complex, nonlinear, and high-order interactions among variables. The proposed method approximates this complex system using a penalized neural network and selects explanatory variables by measuring their utility in explaining the variance of the response variable. This measurement is based on a novel statistic called *Drop-Out-One Loss*. The proposed method also allows (overlapping) group variable selection. We prove that the proposed method can select relevant variables and exclude irrelevant variables with probability one as the sample size goes to infinity, which is referred to as the Oracle Property. Experimental results on simulated and real world datasets show the efficiency of our method in terms of variable selection and prediction accuracy.

1. Introduction

Variable selection is an important task in high dimensional statistics and plays a critical role in many areas such as genomics, genetics and machine learning. Most previous topics on variable selection in high dimensional regression assume that the regression function has some restricted structures such as linearity (Tibshirani, 1996; Fan & Li, 2001; Yuan & Lin, 2006; Fan & Lv, 2008), additivity (Ravikumar et al., 2007; Huang et al., 2010; Fan et al., 2011; Chouldechova & Hastie, 2015), or limited order of interaction among explanatory variables (Lin et al., 2006; Bien et al., 2013). However, recent literature (Jönsson et al., 2010; Curtis et al., 2012) suggests that in real world data, unknown, complex, and nonlinear relationships may exist among response and explanatory variables. Thus, the assumptions of these methods may be too restricted. It is therefore of interest to develop a variable selection technique that can accommodate more flexible regression models.

Our method is motivated by a fundamental idea in a variable selection theory called feature screening (Fan et al., 2011; He et al., 2013; Chang et al., 2013; Cui et al., 2015). It selects variables according to the utility of a single explanatory variable in explaining the response variable. For example, (Fan et al., 2011) selects explanatory variable x_j if $E(f_j^2(x_j))$ is relatively large, where $f_j(x_j) = E(y \mid x_j)$, the projection of y onto x_j. However, the success of this method relies on whether the utility of an explanatory variable can be correctly measured using a marginal regression (which is why additional assumptions such as additivity are often required). In the situation where the regression function is complex, and high-order interactions among variables exist, marginal regression becomes inappropriate to measure the utility. To address the aforementioned problem, we propose a novel approach to measure the utility of an explanatory variable based on the "drop-out-one difference of empirical loss" or *Drop-Out-One Loss* for simplification. Our method first fits a lower bound model using all explanatory variables and then drops out one explanatory variable without refitting. This variable is eliminated if the change in empirical loss is small. We choose neural network as a tool to approximate the system for its universal approximation property (Raghu et al., 2017). The neural network is penalized to avoid overfitting. Additionally, our method allows us to incorporate grouping information of variables into the selection procedure.

2. Problem Formulation

Consider the following nonparametric regression model $y = f^*(\boldsymbol{x}) + \varepsilon$, where $y \in R$, $\boldsymbol{x} \in R^p$, ε is random noise independent of $\boldsymbol{x}$ and has mean zero. The aim is to recover the mapping f^*, given n i.i.d. training samples $(\boldsymbol{x}^{(i)}, y^{(i)})_{i=1,\ldots,n}$. In this paper, we allow $p = O(\exp(n^l))$, where $l \in (0,1)$. Let x_j be the j-th feature or variable of $\boldsymbol{x}$. For this small n large p problem, a sparsity assumption is usually required. That is, $f^*(\boldsymbol{x})$ depends on $\boldsymbol{x}$ only through $\{x_j : j \in S\}$, where $S \subset \{1, 2, ..., p\}$ is an index set of variables and $|S|$, the

[*]Equal contribution [1]Department of Statistics, Purdue University, West Lafayette, IN, USA. Correspondence to: Mao Ye <ye207@purdue.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

cardinality of S, is smaller than n. Our target is to identify S and to train a good model to make prediction. In many real world problems such as gene expression analysis, a natural grouping structure exists within the explanatory variables. In this article, we also allow incorporating this prior information that $\boldsymbol{x}$ can be divided into d different (overlapping) groups. Let $g_1, ..., g_d$ denote the index sets of variables in group $1, 2, \ldots, d$, respectively. For example, if group 1 consists of x_1, x_2 and x_4, then $g_1 = \{1, 2, 4\}$. We denote the variables in group g_j as $\boldsymbol{x}_{g_j}$. Let $S_g = \{j,\ g_j \cap S \neq \emptyset\}$ be the index set of groups that contain relevant variables. Let $S^c = \{1, 2, ..., p\} - S$ and $S_g^c = \{1, 2, ..., d\} - S_g$. In this group variable selection case, our target is to identify S_g and to train a good model. We call variables with indexes in S relevant variables, variables with indexes in S^c irrelevant variables, groups with indexes in S_g relevant groups and groups with indexes in S_g^c irrelevant groups.

3. The Proposed Method

Our method first uses neural network to fit the data using all variables. Then, for each variable or variable group, our method drops out the weights tied to this variable or variable group and records how the loss function changes. The variable or variable group is eliminated if the change in the loss is small. Then we refit the data using the rest of the variables and repeat the above procedure until no variable or variable group can be eliminated. In this way, our method iteratively and greedily eliminates irrelevant variables or variable groups. The neural network is penalized to avoid overfitting. We call our method Greedy Elimination Penalized Neural Network (GEPNN).

3.1. Penalized Neural Network

We consider a neural network with a single hidden layer with p input units, m hidden units and 1 output unit. As is suggested in (Liang et al., 2017; Feng & Simon, 2017), a network with one hidden layer is usually large enough to approximate the system. The mapping is

$$f_{\boldsymbol{\eta}}(\boldsymbol{x}) = \boldsymbol{\beta}^T \psi(\boldsymbol{w}^T \boldsymbol{x} + \boldsymbol{t}) + b,$$

where $\boldsymbol{w} \in R^{p\times m}$, $\boldsymbol{t} \in R^m$, $\boldsymbol{\beta} \in R^{m\times 1}$ and $b \in R^1$. $\psi : R^m \to R^m$ is the activation function of the output of the hidden layer. In this article, we use $tanh$ as the activation function. Let $\boldsymbol{\eta} = (\boldsymbol{w}, \boldsymbol{\theta})$ where $\boldsymbol{\theta} = (\boldsymbol{t}^T, \boldsymbol{\beta}^T, b)^T$. The network is trained by minimizing $\frac{1}{n}\sum_{i=1}^n \ell\left(y^{(i)} - f_{\boldsymbol{\eta}}(\boldsymbol{x}^{(i)})\right) + pen(\boldsymbol{\eta})$,where $\ell(u)$ is the loss function and $pen(\boldsymbol{\eta})$ is penalty for parameters. In this article, we consider the squared error loss $\ell(u) = u^2$ and sparse group lasso penalty defined as $pen(\boldsymbol{\eta}) = \lambda_0 \|\boldsymbol{\theta}\|_2^2 + \lambda_1 \sum_{j=1}^p \Omega_\alpha(\boldsymbol{w}_{j,*})$, where $\Omega_\alpha(\boldsymbol{w}_{j,*}) = (1-\alpha)\|\boldsymbol{w}_{j,*}\|_1 + \alpha\|\boldsymbol{w}_{j,*}\|_2$ and $\boldsymbol{w}_{j,*}$ is the j-th row of $\boldsymbol{w}$. We use the same network and penalty settings as that in (Feng & Simon, 2017), however, other settings can also be considered. We call the weights tied to the irrelevant variables, i.e. $\boldsymbol{w}_{j,*}, j \in S^c$, irrelevant weights and the weights tied to the relevant variables, i.e. $\boldsymbol{w}_{j,*}, j \in S$, relevant weights.

3.2. Greedy Elimination Algorithm

Without loss of generality, we propose the greedy elimination algorithm for (overlapping) group variable selection. Note that individual variable selection is a special case of group variable selection. Given $\boldsymbol{\eta}$, let $supp(\boldsymbol{\eta}) = \{j : \|\boldsymbol{w}_{j,*}\|_2 \neq 0\}$. Here we abuse the conventional notation of support. Let $\gamma \subset \{1, 2, ..., p\}$ be an index set of variables and let $\hat{\boldsymbol{\eta}}(\gamma)$ be the estimated parameters under the constraint that $supp(\boldsymbol{\eta}) \subset \gamma$, i.e. $\hat{\boldsymbol{\eta}}(\gamma) = \arg\min \frac{1}{n}\sum_{i=1}^n \ell\left(y^{(i)} - f_{\boldsymbol{\eta}}(\boldsymbol{x}^{(i)})\right) + pen(\boldsymbol{\eta})$ subject to $supp(\boldsymbol{\eta}) \subset \gamma$. Given $\boldsymbol{\eta} = (\boldsymbol{w}, \boldsymbol{\theta})$, define $\boldsymbol{\eta}^{-g_j} = (\boldsymbol{w}^{-g_j}, \boldsymbol{\theta})$, where $\boldsymbol{w}^{-g_j}$ is $\boldsymbol{w}$ with rows corresponding to group g_j set to be $\boldsymbol{0}$, i.e. $\boldsymbol{w}_{i,*}^{-g_j} = \boldsymbol{w}_{i,*}$ for $i \notin g_j$ and $\boldsymbol{w}_{i,*}^{-g_j} = \boldsymbol{0} \in R^{1\times m}$ for $i \in g_j$. We also define $\boldsymbol{\eta}^{-j} = (\boldsymbol{w}^{-j}, \boldsymbol{\theta})$, where $\boldsymbol{w}^{-j}$ is $\boldsymbol{w}$ with j-th row set to be $\boldsymbol{0}$, i.e. $\boldsymbol{w}_{i,*}^{-j} = \boldsymbol{w}_{i,*}$ for $i \neq j$ and $\boldsymbol{w}_{j,*}^{-j} = \boldsymbol{0} \in R^{1\times m}$. Suppose we have $\tilde{n}$ validation samples $(\tilde{y}^{(i)}, \tilde{\boldsymbol{x}}^{(i)})_{i=1,...,\tilde{n}}$, given $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$, we define the difference of empirical loss based on validation set as follows:

$$\begin{aligned} & \Delta_{\tilde{n}}\mathcal{L}(\boldsymbol{\eta}_1, \boldsymbol{\eta}_2) \\ = & \frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} \left\{ \ell\left(\tilde{y}^{(i)} - f_{\boldsymbol{\eta}_1}(\tilde{\boldsymbol{x}}^{(i)})\right) - \ell\left(\tilde{y}^{(i)} - f_{\boldsymbol{\eta}_2}(\tilde{\boldsymbol{x}}^{(i)})\right) \right\}. \end{aligned}$$

Group j is eliminated by the algorithm if $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-g_j}, \hat{\boldsymbol{\eta}})$ is small. We call $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-g_j}, \hat{\boldsymbol{\eta}})$ the drop-out-one difference of empirical loss of group g_j or drop-out-one loss of group g_j for simplification. The Greedy Elimination algorithm is summarized in Algorithm [2]. Note that, if eliminating one group lowers the loss, that group will be eliminated. $\gamma^{(t-1)}$ and $\mathcal{G}^{(t-1)}$ returned by the Greedy Elimination algorithm are the index sets of selected variables and selected variable groups respectively. The training algorithm for penalized neural network is summarized in Algorithm [1], in which lr is the learning rate. The line search criterion in Algorithm [1] is

$$\begin{aligned} & \frac{1}{n}\sum_{i=1}^n \left(y^{(i)} - f_{\boldsymbol{\eta}^{(t)}}(\boldsymbol{x}^{(i)})\right)^2 + pen(\boldsymbol{\eta}^{(t)}) \\ \leq & \max_{r=\max(0,t-5),...,t-1} \left\{ \frac{1}{n}\sum_{i=1}^n \left(y^{(i)} - f_{\boldsymbol{\eta}^{(r)}}(\boldsymbol{x}^{(i)})\right)^2 \right\} \\ + & pen(\boldsymbol{\eta}^{(t-1)}) - \frac{lr}{2}\left\|\boldsymbol{\eta}^{(t)} - \boldsymbol{\eta}^{(t-1)}\right\|_2^2. \end{aligned}$$

Algorithm [1] is a combination of the GIST algorithm (Gong et al., 2013) and block-wise descent algorithm. Note that in Algorithm [1], we only update $\boldsymbol{w}_{j,*}, j \in \gamma$ and $\boldsymbol{\theta}$. $\boldsymbol{w}_{j,*}, j \in \{1, 2, ..., p\} - \gamma$ are set to be $\boldsymbol{0}$.

Algorithm 1 Training Penalized Neural Network

Initialization: Obtain the initial value $\boldsymbol{\eta}^{(0)}$ by Xavier's method (Glorot & Bengio, 2010).
for $t = 1, 2, ...$ **do**
  $\boldsymbol{\eta}^{(t,1)} = \boldsymbol{\eta}^{(t-1)} - lr\nabla_{\boldsymbol{\eta}}[\frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - f_{\boldsymbol{\eta}}(\boldsymbol{x}^{(i)}))^2 + \lambda_0 \|\boldsymbol{\theta}\|_2^2]$
  $\boldsymbol{w}^{(t,2)} = sign(\boldsymbol{w}^{(t,1)}) \left(\left|\boldsymbol{w}^{(t,1)}\right| - lr(1-\alpha)\lambda_1\right)_+$
  for $j \in \gamma$ **do**
    $\boldsymbol{w}_{j,*}^{(t,3)} = \left(1 - \frac{lr\lambda_1\alpha}{\left\|\boldsymbol{w}_{j,*}^{(t,2)}\right\|_2}\right)_+ \boldsymbol{w}_{j,*}^{(t,2)}$
  end for
  if line search criterion is satisfied **then**
    $\boldsymbol{\eta}^{(t)} = \boldsymbol{\eta}^{(t,3)}$
  else
    $\boldsymbol{\eta}^{(t)} = \boldsymbol{\eta}^{(t-1)}$
    $lr = \frac{9}{10}lr$
  end if
  if iteration converges **then**
    Return $\boldsymbol{\eta}^{(t)}$
  end if
end for

We need to tune λ_0, α, λ_1 and $thre^{(t)}$ for the algorithm. In practice, similar to (Feng & Simon, 2017), we can simply fix λ_0 and α and use the validation set to tune λ_1 and $thre^{(t)}$. Tuning $thre^{(t)}$ does not require choosing a good value for each iteration. One practical method is to set $thre^{(t)}, t \in \{1, ..., \bar{t}\}$ be the δ-th percentile of $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-g_j}, \hat{\boldsymbol{\eta}})$ and set $thre^{(t)}(\mathrm{t} > \bar{t})$ be $thre^{(t)} = \frac{\vartheta}{\sum_{j=1}^{d}\left(\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-g_j}, \hat{\boldsymbol{\eta}})\right)_+}$, where $\hat{\boldsymbol{\eta}}$ is the estimated parameter in t-th iteration. We can fix δ to be a certain number and then tune ϑ. Our method is not very sensitive to the choice of λ_1 and $thre^{(t)}$. Tuning settings for the experiments in this paper are given in supplementary material.

4. Theory

In this section, we study the asymptotic property of the proposed method. All the proofs are in the supplementary material. Suppose λ_0 is fixed, then we represent the penalized optimization problem using its dual form, that is

$$\hat{\boldsymbol{\eta}} = \arg\min \frac{1}{n}\sum_{i=1}^{n} l(y^{(i)}, f_{\boldsymbol{\eta}}(\boldsymbol{x}^{(i)})) + \lambda_1 \sum_{j=1}^{p} \Omega_\alpha(\boldsymbol{w}_{j,*}) \quad (1)$$
$$s.t.\ \boldsymbol{\eta} \in \boldsymbol{\Theta} = \{\boldsymbol{\eta} \in \mathbb{R}^P : ||\boldsymbol{t}||_2^2 + ||\boldsymbol{\beta}||_2^2 + b^2 \le K_{\lambda_0}\},$$

where $K_{\lambda_0} > 0$ is a constant that depends on λ_0 and $P = mp + 2m + 1$. Similar to (Feng & Simon, 2017), we assume the estimator is a global minimizer of this non-convex objective function. We first study the theoretic performance of the proposed method when variables are not grouped and then give the result for group variable selection.

Algorithm 2 Greedy Elimination Method

Initialization: Let $\gamma^{(0)} = \{1, 2, 3, \ldots, p\}, \mathcal{G}^{(0)} = \{1, \ldots, d\}$
for $t = 1, 2, ...$ **do**
  Re-estimate: Obtain $\hat{\boldsymbol{\eta}}(\gamma^{(t-1)})$ by Algorithm [1]
  for $j\ in\ \mathcal{G}^{(t-1)}$ **do**
    if $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-g_j}(\gamma^{(t-1)}), \hat{\boldsymbol{\eta}}(\gamma^{(t-1)})) < thre^{(t)}$ **then**
      $\mathcal{G}^{(t-1)} = \mathcal{G}^{(t-1)} - \{j\}$
    end if
  end for
  $\gamma^{(t-1)} = \cup_{i \in \mathcal{G}^{(t-1)}} g_i$
  if $t > 1$ and $\mathcal{G}^{(t-1)} == \mathcal{G}^{(t-2)}$ **then**
    Return $\gamma^{(t-1)}$ and $\mathcal{G}^{(t-1)}$
  else
    $\gamma^{(t)} = \gamma^{(t-1)}$
    $\mathcal{G}^{(t)} = \mathcal{G}^{(t-1)}$
  end if
end for

4.1. Result when variables are not grouped

Define $\mathbb{P}$ as the underlining joint-distribution of $\boldsymbol{X}$ and Y, so that $\mathbb{P}g(\boldsymbol{x}, y) = \int_{\mathcal{X}\times\mathcal{Y}} g(\boldsymbol{x}, y)dF(\boldsymbol{x}, y)$, where $g : \mathcal{X} \times \mathcal{Y} \to R$ and $\mathcal{X}$, $\mathcal{Y}$ is the support of $\boldsymbol{X}$ and Y. Let $\mathbb{P}_n$ be the empirical joint-distribution of $\boldsymbol{X}$ and Y based on training set and then $\mathbb{P}_n g(x, y) = \frac{1}{n}\sum_{i=1}^{n} g(\boldsymbol{x}^{(i)}, y^{(i)})$. Similarly, let $\mathbb{P}_{\tilde{n}}$ be the empirical joint-distribution based on validation set and then $\mathbb{P}_{\tilde{n}} g(x, y) = \frac{1}{\tilde{n}}\sum_{i=1}^{\tilde{n}} g(\tilde{\boldsymbol{x}}^{(i)}, \tilde{y}^{(i)})$. We denote $\ell\left(y - f_{\boldsymbol{\eta}}(\boldsymbol{x})\right)$ as $\ell_{\boldsymbol{\eta}}(y, \boldsymbol{x})$ for simplification. Now we introduce equivalent class to classify the networks that have different parameterization but form same mapping. Given parameter $\boldsymbol{\eta}$, the set of its equivalent parameterization is defined as: $Eq(\boldsymbol{\eta}) = \left\{\boldsymbol{\eta}^{'} \in \Theta :\ f_{\boldsymbol{\eta}^{'}}(\boldsymbol{x}) = f_{\boldsymbol{\eta}}(\boldsymbol{x}),\ a.e.\ \boldsymbol{x} \in \mathcal{X}\right\}$. Given the loss function, the set of optimal networks is $Eq_0 = \{\boldsymbol{\eta} : \boldsymbol{\eta} = \arg\min_{\boldsymbol{\eta}\in\boldsymbol{\Theta}} \mathbb{P}\ell_{\boldsymbol{\eta}}(y, \boldsymbol{x})\}$. We assume Eq_0 is composed of Q equivalent classes, where Q is a finite number. Similar to the definition of $\Delta_{\tilde{n}}\mathcal{L}(\boldsymbol{\eta}_1, \boldsymbol{\eta}_2)$, we also define $\Delta_n\mathcal{L}(\boldsymbol{\eta}_1, \boldsymbol{\eta}_2) = \mathbb{P}_n\left\{\ell_{\boldsymbol{\eta}_1}(x, y) - \ell_{\boldsymbol{\eta}_2}(x, y)\right\}$ and $\Delta\mathcal{L}(\boldsymbol{\eta}_1, \boldsymbol{\eta}_2) = \mathbb{P}\left\{\ell_{\boldsymbol{\eta}_1}(x, y) - \ell_{\boldsymbol{\eta}_2}(x, y)\right\}$. Given $\boldsymbol{\eta}$, let $\boldsymbol{\eta}_0^{(\boldsymbol{\eta})} = \arg\min_{\boldsymbol{\eta}_0 \in Eq_0} \|\boldsymbol{\eta} - \boldsymbol{\eta}_0\|_2$. We introduce the following assumptions:

Assumption 1: $y = f^*(\boldsymbol{x}) + \varepsilon$, ε is a sub-gaussian noise, i.e., $P(|\epsilon| > t) \le 2\exp\left(-\frac{t^2}{2\sigma^2}\right)$ independent of $\boldsymbol{x}$ and has mean zero. $\mathcal{X} \subseteq [X_{min}, X_{max}]^p$ and $|X_{min}| \vee |X_{max}| \le M_1 < \infty$ for some constant M_1, where $a \vee b = \max(a, b)$. $|f^*(\boldsymbol{x})| < M_0 \le \infty,\ a.e.\ \boldsymbol{x} \in \mathcal{X}$ for some constant M_0.

Assumption 2: The number of hidden units is a constant $m < \infty$. The activation function $\psi :\ R^{K_1} \to R^{K_2}$ in the network satisfies $\|\psi(\boldsymbol{z})\|_\infty \le M_2 < \infty$, $\|\nabla\psi(\boldsymbol{z})\|_\infty \le$

$M_3 < \infty$ and $\left\|\nabla^2\psi(\boldsymbol{z})\right\|_\infty \leq M_4$, $\forall \boldsymbol{z} \in R^{K_1}$ for some constant M_2, M_3 and M_4. Note that both $sigmoid$ and $tanh$ satisfy our assumption.

Assumption 3: $\boldsymbol{\Theta}$ is large enough such that $\forall \boldsymbol{\eta}_0 \in Eq_0$, $\nabla_{\boldsymbol{\eta}}\mathbb{P}\ell_{\boldsymbol{\eta}}(y,\boldsymbol{x}) \mid_{\boldsymbol{\eta}=\boldsymbol{\eta}_0} = 0$ and $\sup_{\boldsymbol{\eta}\in\boldsymbol{\Theta}} \max_{j_1,j_2,j_3} \left|\frac{\partial^3 \mathbb{P}\ell_{\boldsymbol{\eta}}(y,\boldsymbol{x})}{\partial\boldsymbol{\eta}_{j_1}\partial\boldsymbol{\eta}_{j_2}\partial\boldsymbol{\eta}_{j_3}}\right| \leq M_5 < \infty$ for some constant M_5.

Assumption 4: $\forall \boldsymbol{\eta}_0, \boldsymbol{\eta}_0^{'} \in Eq_0$, $supp(\boldsymbol{\eta}_0) = supp(\boldsymbol{\eta}_0^{'}) = S$.

Assumption 5: Suppose $\boldsymbol{\eta}$ can be reordered as $\boldsymbol{\eta} = (\boldsymbol{w}_S, \boldsymbol{\theta}, \boldsymbol{w}_{S^c})$, where $\boldsymbol{w}_S$ is the weights tied to the input nodes with indexes in S and $\boldsymbol{w}_{S^c}$ is the weights tied to the input nodes with indexes in S^c. Denote $A \succeq B$ when $A - B$ is semi-positive definite. We have $\nabla^2_{\boldsymbol{\eta}}\mathbb{P}\ell_{\boldsymbol{\eta}}(y,\boldsymbol{x}) \mid_{\boldsymbol{\eta}=\boldsymbol{\eta}_0} \succeq H \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}$, where $\boldsymbol{I}$ is an identity matrix in $R^{\{(|S|+2)m+1\}\times\{(|S|+2)m+1\}}$, $\boldsymbol{0}$ are zero matrices with appropriate dimension and H is a positive constant that is independent of p.

Assumption 6: (Identifiability condition) $\forall \epsilon > 0$, $\exists \alpha_\epsilon > 0$ independent of p such that

$$\alpha_\epsilon < \inf_{\boldsymbol{\eta}\in\boldsymbol{\Theta}} \Big\{ \mathbb{P}\left\{\ell_{\boldsymbol{\eta}}(y,\boldsymbol{x}) - \ell_{\boldsymbol{\eta}_0}(y,\boldsymbol{x})\right\} : \left\|\boldsymbol{\eta} - \boldsymbol{\eta}_0^{(\boldsymbol{\eta})}\right\| \geq \epsilon, \\ \|\boldsymbol{w}_{S^c}\|_1 \leq 3\sum_{j\in S} \Omega_\alpha(\boldsymbol{w}_{j,*} - \boldsymbol{w}_{0j,*}^{(\boldsymbol{\eta})}) + \left\|\boldsymbol{\theta} - \boldsymbol{\theta}_0^{(\boldsymbol{\eta})}\right\|_2 \Big\}.$$

Remark: Assumption 1, 2, 3 and 5 are just some mild conditions that regularize the property of distribution, network structure and loss function. Assumption 4 implies that relevant variables can't be represented by irrelevant variables. This assumption mimics the restricted isometry condition in (Zhao & Yu, 2006; Zhang, 2009). Assumption 6 sets a lower bound that distinguishes the optimal network from the network that is not optimal.

Let $\hat{S}$ be the set of selected variables when the algorithm converges and let $\hat{S}^{(t)}$ be the set of selected variables at the t-th iteration of the proposed method. In the first two theorems, we show that the set of selected variables in the first iteration, i.e. $\hat{S}^{(1)}$, is equal to S asymptotically.

Theorem 1: Denote $\hat{\boldsymbol{\eta}}$ be the solution of (1) in the first iteration of the proposed method. Suppose assumption 1 to assumption 6 hold and $\lambda_1 = \left[\sqrt{\frac{m\log n}{n}}\left\{\sqrt{\log Q} + \frac{\sqrt{m\log p}\log(nm)}{1-\alpha+\frac{\alpha}{\sqrt{m}}}\right\}\right](c_0 + o(1))$, for a constant c_0, we have

$$\left|\Delta_n\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})\right| = O_p(\frac{\log^3 n\log p}{n}),\ \forall j \in S^c,$$

$$\Delta_n\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}}) \geq C + O_p(\frac{1}{\sqrt{n}} \vee \frac{\log^3 n\log p}{n}),\ \forall j \in S,$$

where $C > 0$ is a constant.

Theorem 2: (Oracle Property) Denote $\hat{\boldsymbol{\eta}}$ be the solution of (1) in the first iteration of the proposed method. Suppose assumption 1 to assumption 6 hold, $\lambda_1 = \left[\sqrt{\frac{m\log n}{n}}\left\{\sqrt{\log Q} + \frac{\sqrt{m\log p}\log(nm)}{1-\alpha+\frac{\alpha}{\sqrt{m}}}\right\}\right](c_0 + o(1))$, for a constant c_0, and $\tilde{n} = O(n)$, we have

$$\left|\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})\right| = O_p(\frac{\log^3 n\log p}{n}),\ \forall j \in S^c,$$

$$\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}}) \geq C + O_p(\frac{1}{\sqrt{n}} \vee \frac{\log^3 n\log p}{n}),\ \forall j \in S$$

where $C > 0$ is a constant.

Thus, when $thre^{(1)}$ is properly tuned, we have the oracle property that $\hat{S}^{(1)} = S$ with probability 1 as $n \to \infty$. Notice that theorem 1 implies that we can also use the empirical loss based on training set to select variables but in practice, our experiment implies that it is more efficient to use empirical loss based on validation set. Since our result shows that $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}_{-j}, \hat{\boldsymbol{\eta}}) \geq C + O_p(\frac{1}{\sqrt{n}} \vee \frac{\log^3 n\log p}{n})$, $\forall j \in S$, we have the following corollary.

Corollary 3: Suppose the assumptions in Theorem 2 hold, when $thre^{(t)}$ is properly tuned, we have $\hat{S} = S$ with probability 1 as $n \to \infty$.

We successfully show that our method has oracle property in selecting variables, which is not established in (Feng & Simon, 2017). To the best of our knowledge, we are the first to give a frequentist variable selection method that has oracle property for high dimensional nonparametric regression problem that does not assume rigorous regression structure, such as, additivity and limited order of interaction.

4.2. Result for group variable selection

When variables are grouped, define $supp_g(\boldsymbol{\eta}) = \{j : \sum_{i\in g_j} \|\boldsymbol{w}_{i,*}\|_2 \neq 0\}$. Recall that $S_g = \{i,\ g_i \cap S \neq \emptyset\}$ is the index set of groups that contain relevant variables. We can change Assumption 4 to the following assumption:

Assumption 4* $\forall \boldsymbol{\eta}_0, \boldsymbol{\eta}_0^{'} \in Eq_0$, $supp_g(\boldsymbol{\eta}_0) = supp_g(\boldsymbol{\eta}_0^{'}) = S_g$.

Note that Assumption 4* is actually weaker than Assumption 4 since $supp(\boldsymbol{\eta}) = S$ implies $supp(\boldsymbol{\eta})_g = S_g$. Let $\hat{S}_g$ be the set of selected groups of variables when the algorithm ends. We have the following Corollary:

Corollary 4: Suppose the assumptions in Theorem 2 hold (replace Assumption 4 with Assumption 4*), when $thre^{(t)}$ is properly tuned, we have $\hat{S}_g = S_g$ with probability 1 as $n \to \infty$.

5. Experiment

5.1. Simulation study

5.1.1. Case 1: individual variable selection

In this example, we generate 10 datasets from the following model,

$$\begin{aligned} y &= \frac{10\sin(x_1 \vee x_2) + (x_3 \vee x_4 \vee x_5)^3}{1+(x_1+x_5)^2} \\ &+ \sin(0.5x_3)(1+\exp^{x_4-0.5x_3}) \\ &+ x_3^2 + 2\sin(x_4) + 2x_5 + \epsilon, \end{aligned}$$

where $\epsilon \sim N(0,1)$. Each dataset consists of 600 observations, with 200 for training, 100 for validation and 300 for testing. Variables $x_1, ..., x_5$ with other 495 additional variables are generated by $x_j^{(i)} = \frac{e^{(i)}+z_j^{(i)}}{2}$, $j = 1, ..., 500$, $i = 1, ..., 600$, where $e^{(i)}$ and $z_j^{(i)}$ are independently generated from $N(0,1)$. In this way, all variables are mutually correlated with a correlation of 0.5. We use the false selection rates (FSR) and negative selection rates (NSR) to measure the performance of variable selection. Let S denote the set of true relevant variables and let $\widehat{S}_i$ denote the set of selected variables of dataset i. Define $FSR = \frac{\sum_{i=1}^{10}|\widehat{S}_i - S|}{\sum_{i=1}^{10}|\widehat{S}_i|}$, $NSR = \frac{\sum_{i=1}^{10}|S-\widehat{S}_i|}{\sum_{i=1}^{10}|S|}$. The method performs well when both FSR and NSR are small. We calculate the mean square prediction error (MSPE) and mean square fitting error (MSFE) to measure the predictive and fitting performance. For comparison, we apply Sparse input neural network (Spinn) and l_1 penalized neural network (l_1-NN) (Feng & Simon, 2017), generalized additive model selection (GAM) (Chouldechova & Hastie, 2015), random forest (RF) (Breiman, 2001), Bayesian adaptive regression trees (BART) (Bleich et al., 2014), Model-X knockoffs (Knockoffs) (Candes et al., 2018) and Sure Independence Screening using SCAD penalty (SIS-SCAD) (Fan & Lv, 2008; Fan & Li, 2001). Spinn and l_1-NN are also penalized neural network based variable selection methods. GAM is a penalized likelihood approach for fitting sparse high dimensional generalized additive model. It can be viewed as a generalized version of sparse additive model (Spam) (Ravikumar et al., 2007). Both RF and BART are regression tree-based methods, which are flexible in capturing nonlinearities and interaction effects in the unknown regression function. For RF and BART, similar setting as (Liang et al., 2017; Bleich et al., 2014) is applied. We select the variables with variable importance percentage greater than 1% and use a 500-tree RF and try BART with 20, 35 and 50 trees in all the numerical experiments. Knockoffs is a nonparametric controlled variable selection method but it assumes the distributions of variables are known. In all experiment, we try Knockoffs with FDR $= 0.1, 0.2$ and 0.3. SIS-SCAD is a screening based method for linear model. In all the experiments, we use the same network structure for Spinn and GEPNN. We also try different structures and the results are similar. In this case, the network structure of Spinn and GEPNN are set to have 6 hidden units. The result is summarized in Table [1]. Details on implementation for all experiments are in the supplementary material.

5.1.2. Case 2: group variable selection

In this example, we generate 10 datasets from the following model,

$$y = \frac{4(x_1 \vee x_2)^3}{1+2x_3^2} sin(x_4) + \exp^{x_3-x_4}(1+x_5+x_6) + \epsilon,$$

where $\epsilon \sim N(0,1)$. Each dataset consists of 600 observations, with 200 for training, 100 for validation and 300 for testing. Variables $x_1, ..., x_5$ with other 497 additional variables are generated using the same method in case 1. Overlapped groups are predefined such that $\{x_1, ..., x_4\}$ forms the first group, $\{x_3, ..., x_6\}$ forms the second group, $\{x_5, ..., x_8\}$ forms the third group,.... Each group overlaps half of the previous group. We use the group false selection rates (gFSR) and group negative selection rates (gNSR) to measure the performance in group variable selection. Let S_g denote the set of true relevant groups and let $\widehat{S}_{gi}$ denote the set of selected groups of dataset i. Define $gFSR = \frac{\sum_{i=1}^{10}|\widehat{S}_{gi} \backslash S_g|}{\sum_{i=1}^{10}|\widehat{S}_{gi}|}$ and $gNSR = \frac{\sum_{i=1}^{10}|S_g \backslash \widehat{S}_{gi}|}{\sum_{i=1}^{10}|S_g|}$. The network structures of Spinn and GEPNN are set to have 6 hidden units. We use overlapping group lasso linear model (LR-OGL) instead of SIS-SCAD since the variables are grouped. The result is summarized in Table [1].

5.2. Real Data

We use 4 real datasets, CCLE, CCPP (Combined Cycle Power Plant), Airfoil and Boston Housing to evaluate the performance. For each dataset, we run 10 runs, where for each run, the dataset is randomly split into a training set, a validation set and a test set. CCLE is taken from (Liang et al., 2017) and the other 3 datasets are from UCI machine learning repository. CCLE consists of 490 instances with 81 attributes. Training set, validation set and test set consist of 245, 122 and 123 instances respectively. The Airfoil consists of 1503 instances with 5 variables and the CCPP has 9568 instances with 4 variables. The variables in these two datasets are normalized to have means 0 and variance 1. We add 500 additional irrelevant variables which are independently drawn from $N(0,1)$ to Airfoil and CCPP. For these two datasets, we use 200 instances for training, 100 instances for validating and the rest for testing. CCLE, CCPP and Airfoil are datasets for individual variable selection. The Boston Housing contains 506 instances with 2 discrete variables and 11 continuous variables. Similar as (Swirszcz et al., 2009), we consider third-polynomial expan-

Table 1. Result of simulation case1 and case2. $\left|\widehat{S}_i\right|$ denotes the number of selected variables in each run. $\left|\widehat{S}_{gi}\right|$ denotes the number of selected variable groups in each run. FSR and NSR denote the False Selection Rate and Negative Selection Rate respectively. gFSR and gNSR denote the group False Selection Rate and group Negative Selection Rate respectively. MSFE denotes the mean square of fitting error and MSPE denotes the mean square of prediction error. BART-20, BART-30 and BART-50 denote BART with 20, 30 and 50 trees respectively. Knockoffs-0.1, Knockoffs-0.2 and Knockoffs-0.3 denote Knockoffs using FDR $= 0.1, 0.2$ and 0.3, respectively. The numbers outside the parentheses denote the mean of the corresponding values. The numbers in parentheses denote the standard deviations of the corresponding values. *In SIS-SCAD, the algorithm fails to detect the variables and in each run, the number of selected variables exceeds its default threshold (37 in this case), so it returns 37 variables each time.

	METHODS	$\left\|\widehat{S}_i\right\|$	FSR	NSR	MSFE	MSPE
CASE 1	OURS	5.2(0.4)	0.038	0	1.57(0.15)	2.34(0.25)
	SPINN	8.2(3.66)	0.463	0.12	1.49(0.29)	6.68(0.76)
	l_1-NN	3.6(2.12)	0.167	0.4	2.87(0.83)	6.08(0.77)
	GAM	29.2(9.92)	0.829	0	3.14(0.59)	26.71(67.93)
	RF	10.5(2.32)	0.610	0.18	0.97(0.10)	6.63(1.65)
	BART-20	15.9(3.25)	0.755	0.22	2.96(0.46)	7.19(1.20)
	BART-35	12.2(3.29)	0.672	0.2	2.18(0.49)	6.90(1.70)
	BART-50	9.6(3.06)	0.562	0.16	1.45(0.27)	6.93(1.62)
	KNOCKOFFS-0.1	0	NA	NA	-	-
	KNOCKOFFS-0.2	2.3(2.98)	0.22	0.64	-	-
	KNOCKOFFS-0.3	4(3.46)	0.25	0.4	-	-
	SIS-SCAD	37(0.0)*	0.881	0.12	1.92(0.30)	9.87(1.80)
	METHODS	$\left\|\widehat{S}_{gi}\right\|$	GFSR	GNSR	MSFE	MSPE
CASE 2	OURS	3.2(0.42)	0.0625	0	2.16(0.63)	4.20(0.80)
	SPINN	9.83(6.81)	0.533	0.57	2.43(0.50)	6.62(1.49)
	l_1-NN	6.44(8.32)	0.61	0.81	4.22(2.06)	6.78(1.72)
	GAM	33.6(17.26)	0.911	0	4.68(2.33)	5.97(3.81)
	RF	11.4(3.47)	0.737	0	1.14(0.42)	7.20(4.08)
	BART-20	19.1(4.07)	0.895	0.33	2.31(0.67)	9.64(4.02)
	BART-35	13.0(5.50)	0.808	0.17	1.26(0.28)	9.08(4.00)
	BART-50	7.1(2.28)	0.732	0.37	0.81(0.19)	9.77(4.54)
	KNOCKOFFS-0.1	0	NA	NA	-	-
	KNOCKOFFS-0.2	0	NA	NA	-	-
	KNOCKOFFS-0.3	6.2(8.98)	0.71	0.4	-	-
	LR-OGL	190.8(15.19)	0.984	0	3.51(1.06)	8.16(3.73)

sion for the 11 continuous variables, i.e. x_i, x_i^2 and x_i^3 and consider them as a group. We let 2 discrete variables form 2 groups and add 154 additional groups each consisting of 3 irrelevant variables independently drawn from $N(0,1)$. We use 200, 100 and 206 instances for training, validating and testing. We set the number of hidden units of Spinn and GEPNN to be 3 for CCLE, CCPP and Airfoil. Since we add nonlinear features for Boston Housing dataset, we reduce the number of hidden units to 2 for Spinn and GEPNN. For Boston Housing dataset, we use sparse group lasso instead of SIS-SCAD since it is a group variable selection problem. To measure the sparsity, we calculate the number of selected variables or variable groups, denoted as $\left|\widehat{S}_i\right|$ and $\left|\widehat{S}_{gi}\right|$ respectively. For datasets with artificial noise variables, we also calculate the number of selected variables or variable groups in original dataset denoted as $\left|\widehat{S}_i\right|_q$ and $\left|\widehat{S}_{gi}\right|_q$ respectively, where q is the number of original variables or original groups of variables. Result for CCLE is summarized in Table [2] and result for the other 3 datasets is summarized in Table [3]. We don not compare Knockoffs in CCLE data since it can not predict and the possible relevant variables are unknown. In the other 3 datasets, the performance of Knockoffs is not stable. In CCPP and Airfoil, it tends to select no variables or too many variables. Thus, the variance of the number of selected variables is large. In Boston, it selects too many variables[1]. In all the 4 datasets, GEPNN has the highest averaged prediction accuracy and also tends to selects less variables. We use t-test to show the statistical significance of GEPNN in terms of having higher prediction accuracy and obtaining a sparser model. For each dataset, we compare GEPNN with other two methods that have the smallest averaged MSPE. The result is summarized in Table [4]. For CCLE dataset, although the MSPE of GEPNN is not significantly smaller, the number of variables selected by GEPNN is significantly smaller. For the other 3 datasets, MSPE of GEPNN is significantly smaller than that of all the compared methods. Note that although the p-values for MSPE of CCPP/GAM and Boston/RF are relatively not very small, GEPNN obtains a much sparser model.

6. Related Work

Spinn by (Feng & Simon, 2017) is closely related to our method, which also uses a penalized neural network to select variables. It selects variables by shrinking the weights of irrelevant variables to zero. Here we point out several major differences.

[1] Some variables in original dataset of Boston are irrelevant.

Table 2. Result for CCLE data. The setting of this table is same as that of Table [1].

METHODS	$\lvert\widehat{S}_i\rvert$	MSFE	MSPE
OURS	7.3(2.26)	1.15(0.16)	1.19(0.061)
SPINN	28.9(7.71)	0.63(0.068)	1.24(0.17)
l_1-NN	16.9(5.07)	0.76(0.084)	1.20(0.13)
GAM	22.3(4.14)	0.89(0.11)	32.37(57.08)
RF	36.1(3.18)	0.18(0.014)	1.22(0.088)
BART-20	32.4(3.27)	0.68(0.081)	1.29(0.11)
BART-35	44.8(3.91)	0.62(0.087)	1.23(0.12)
BART-50	51.3(2.50)	0.58(0.066)	1.23(0.10)
SIS-SCAD	14.6(2.91)	0.95(0.14)	1.27(0.10)

Criterion in selecting variables: Spinn selects variables by shrinking the weights of irrelevant variables to zero, while our method eliminates irrelevant variables by measuring their utility in explaining the response variable. To study the difference between these two criteria in identifying irrelevant variables, we conduct the following experiment: We use the method in case 1 of simulation study with $p = 500, 200, 100$ to generate the data. Figure [1] shows the $\|\hat{\boldsymbol{w}}_{j,*}\|_2^2$ of Spinn and $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})$ in the first iteration of our method, $j = 1, ..., p$ (for the cases $p = 500, 200$, we sort values and plot the 100 largest values). Here we use the squared l_2 norm for the weights to make the convergence rate of irrelevant weights the same order as the convergence rate of $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}}), j \in S^c$. Both Spinn and our method are properly tuned. Different from what we expect from a linear model, $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})$ is not necessarily proportional to $\|\hat{\boldsymbol{w}}_{j,*}\|_2^2$. In Spinn, there is no significant gap between the squared l_2 norm of relevant weights and that of irrelevant weights even when p is small enough. For example, in Figure [1], when $p = 500$, the squared l_2 norm of 4 irrelevant weights are higher than that of 2 relevant weights. Even when $p = 100$, there is no significant difference between $\max_{j\in S^c} \|\hat{\boldsymbol{w}}_{j,*}\|_2^2$ and $\min_{j\in S} \|\hat{\boldsymbol{w}}_{j,*}\|_2^2$. However, we can observe a significant gap in our method since $\min_{j\in S} \Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}_{-j}, \hat{\boldsymbol{\eta}}) \gg \max_{j\in S^c} \Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}_{-j}, \hat{\boldsymbol{\eta}})$. This agrees with our result in Theorem 2. Note that, similar to the inclusion probability in (Meinshausen & Buhlmann, 2010), *Drop-Out-One Loss* can alternatively be applied to rank the importance of variables.

The role of λ_1: In practice, (Feng & Simon, 2017) needs to choose λ_1 carefully so that the weights of irrelevant variable can shrink to zero. However, our method is not as sensitive to the choice of λ_1. We only use λ_1 to obtain a lower bound model in which the relevant variables make enough contribution to explain the variance of y.

Iterative selection procedure: In practice, we use the iterative selection procedure to further improve the performance of GEPNN. This way, the lower bound model can be refined through iterations. This procedure is useful because, different from a linear or additive system, we allow any form of interaction between variables.

Table 3. Result for CCPP, Airfoil and Boston Housing. $\left|\widehat{S}_i\right|_q$ denotes the number of selected variables in original dataset, where q is the number of variables in original dataset. $\left|\widehat{S}_{gi}\right|_q$ denotes the number of selected groups in original dataset, where q is the number of groups in original dataset. The other settings are the same as that of Table [1].

	METHODS	$\lvert\widehat{S}_i\rvert$	$\lvert\widehat{S}_i\rvert_4$	MSFE	MSPE
CCPP	OURS	2.4(0.50)	2.3(0.47)	0.075(0.012)	0.075(0.0044)
	SPINN	3.8(3.19)	2.9(0.66)	0.076(0.017)	0.10(0.0092)
	l_1-NN	3.3(1.89)	2.6(0.52)	0.066(0.017)	0.090(0.0055)
	GAM	6.0(3.62)	3.0(0.67)	0.067(0.013)	0.079(0.0052)
	RF	3.9(0.32)	3.9(0.32)	0.017(0.0019)	0.12(0.0058)
	BART-20	10.3(2.95)	2.9(0.57)	0.070(0.011)	0.14(0.026)
	BART-35	10.8(2.70)	3.0(0.67)	0.041(0.0093)	0.13(0.023)
	BART-50	9.3(4.05)	2.5(0.53)	0.022(0.0046)	0.11(0.014)
	KNOCKOFFS-0.1	1.0(3.16)	0.4(1.26)	-	-
	KNOCKOFFS-0.2	3.9(3.51)	2.4(2.07)	-	-
	KNOCKOFFS-0.3	3.3(3.50)	1.9(1.73)	-	-
	SIS-SCAD	1.2(0.42)	1.2(0.42)	0.092(0.015)	0.098(0.0090)
	METHODS	$\lvert\widehat{S}_i\rvert$	$\lvert\widehat{S}_i\rvert_5$	MSFE	MSPE
AIRFOIL	OURS	3.5(0.97)	3.3(0.67)	23.06(7.34)	25.33(4.91)
	SPINN	21.6(12.22)	3.2(0.42)	0.46(0.18)	70.20(13.54)
	l_1-NN	18.2(11.07)	3.0(1.05)	0.62(0.64)	69.63(7.48)
	GAM	5.1(3.28)	3.2(0.63)	31.56(4.43)	34.42(3.18)
	RF	5(1.15)	3.3(0.48)	5.24(0.42)	35.90(1.72)
	BART-20	12.8(1.40)	3.7(0.67)	19.47(2.76)	36.81(4.21)
	BART-35	9.5(2.42)	3.4(0.70)	14.16(2.35)	34.09(2.80)
	BART-50	6.0(1.70)	3.8(0.80)	12.59(1.92)	34.67(2.22)
	KNOCKOFFS-0.1	0	0	-	-
	KNOCKOFFS-0.2	1.3(2.75)	0.7(1.49)	-	-
	KNOCKOFFS-0.3	7.4(3.92)	4(0)	-	-
	SIS-SCAD	37(0.0)*	4.7(0.48)	10.55(0.98)	39.11(3.81)
	METHODS	$\lvert\widehat{S}_{gi}\rvert$	$\lvert\widehat{S}_{gi}\rvert_{13}$	MSFE	MSPE
BOSTON	OURS	5.1(1.20)	4.1(1.37)	13.25(4.30)	17.37(2.85)
	SPINN	19.0(6.41)	5.2(1.40)	2.32(2.90)	60.57(7.87)
	l_1-NN	12.4(6.77)	3.5(0.85)	1.12(0.54)	62.09(12.78)
	GAM	17.4(19.17)	1.9(1.27)	16.09(6.60)	25.00(4.32)
	RF	6.7(0.82)	6.7(0.82)	2.90(0.48)	21.40(6.11)
	BART-20	16.2(2.00)	6.9(1.19)	9.09(1.86)	27.72(7.10)
	BART-35	19.2(1.60)	7.2(1.32)	5.42(1.33)	25.12(5.33)
	BART-50	17.5(2.32)	7.3(1.16)	3.40(0.81)	24.39(4.15)
	KNOCKOFFS-0.1	12.8(1.32)	11.4(0.7)	-	-
	KNOCKOFFS-0.2	15.3(2.45)	11.7(0.48)	-	-
	KNOCKOFFS-0.3	26.6(6.52)	12.1(0.74)	-	-
	SGL	9.2(4.78)	2.3(0.48)	19.23(2.39)	29.00(5.04)

Incorporating grouping information: The group lasso penalty used in Spinn only groups the weights tied to the same variable. Under this framework, we cannot incorporate the grouping information of variables unless a new penalty is developed. However, our method allows us to easily eliminate groups of variables and allows the groups to overlap with each other.

Theoretical contributions: We show GEPNN can select relevant variables and eliminate irrelevant ones with probability 1 as $n \to \infty$. This property is referred as the "Oracle Property", which is not established in (Feng & Simon, 2017).

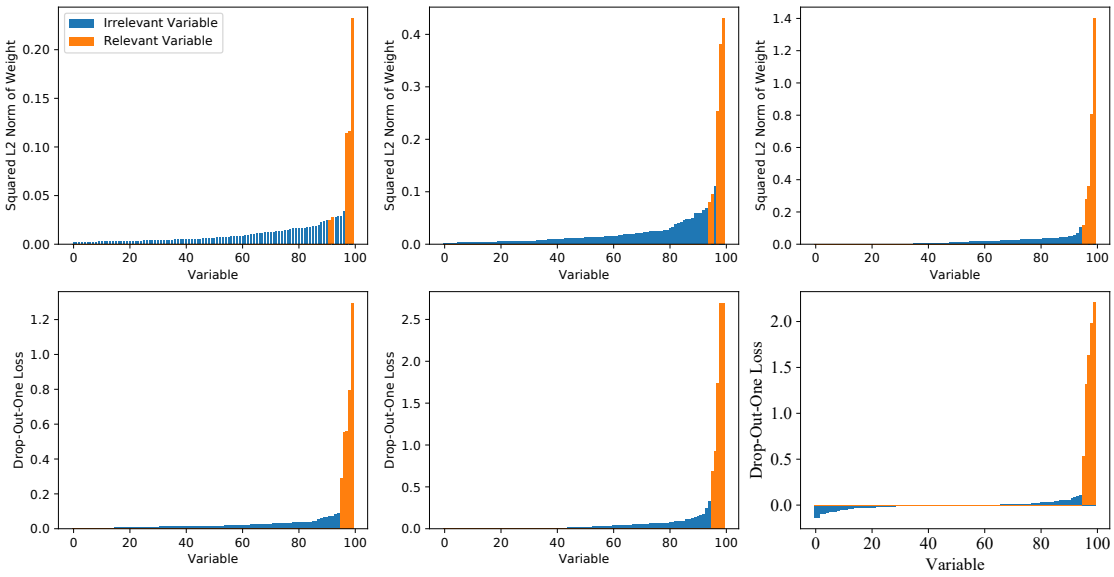

Figure 1. Barplot for $\|\hat{\boldsymbol{w}}_{j,*}\|_2^2$ and $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})$ (sorted by ascending order). The first row is for $\|\hat{\boldsymbol{w}}_{j,*}\|_2^2$ and the second row is for $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})$. The left, middle and right columns are for the cases $p = 500, 200$ and 100, respectively. For cases $p = 200, 500$, we only plot the 100 largest values. The orange bars are for the corresponding $\|\hat{\boldsymbol{w}}_{j,*}\|_2^2$ or $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})$ of relevant variables and the blue bars are for the corresponding $\|\hat{\boldsymbol{w}}_{j,*}\|_2^2$ or $\Delta_{\tilde{n}}\mathcal{L}(\hat{\boldsymbol{\eta}}^{-j}, \hat{\boldsymbol{\eta}})$ of irrelevant variables.

Table 4. Result for statistical test. The numbers are corresponding p-values. $\left|\widehat{S}_i\right|$ *or* $\left|\widehat{S}_{gi}\right|$ is the number of selected variables or variable groups. $\left|\widehat{S}_i\right|_q$ *or* $\left|\widehat{S}_{gi}\right|_q$ is the number of selected variables or variable groups in original dataset. The other settings are the same as that of Table [1].

DATA/METHOD	$\left\|\widehat{S}_i\right\|$ *or* $\left\|\widehat{S}_{gi}\right\|$	$\left\|\widehat{S}_i\right\|_q$ *or* $\left\|\widehat{S}_{gi}\right\|_q$	MSPE
CCLE/RF	$1.16e{-9}$	-	$1.99e{-1}$
CCLE/l_1-NN	$6.24e{-5}$	-	0.3669
CCPP/l_1-NN	0.077	0.072	$1.21e{-6}$
CCPP/GAM	$5.59e{-3}$	$5.31e{-3}$	$2.86e{-2}$
AIRFOIL/BART-35	$5.20e{-6}$	0.37	$1.10e{-4}$
AIRFOIL/GAM	0.63	0.08423	$8.68e{-5}$
BOSTON/RF	$1.54e{-3}$	$6.34e{-5}$	$4.12e{-2}$
BOSTON/BART-50	$6.87e{-8}$	$1.60e{-4}$	$8.53e{-4}$

Other related literature (Liang et al., 2017) uses Bayesian neural networks to select variables; However, compared with our method, it is much more computationally expensive. Our method is also similar to (Couvreur & Bresler, 2000), which can be viewed as an alternative matching pursuit method (Sindhwani & Lozano, 2011; Zhang, 2011). However, (Couvreur & Bresler, 2000) is designed for linear models with low dimension, and its selection procedure is also different from ours. (Alvarez & Salzmann, 2016; Scardapane et al., 2017) also focus on selecting the nodes in neural network. However, their target is to obtain a compact network rather than select input variables in a high dimensional setting.

7. Conclusion

We address the problem of variable selection for high dimensional nonparametric regression. In contrast to previous methods, we do not make structural assumptions such as linearity and additivity on the regression function. We propose a novel approach to eliminate irrelevant variables or groups of variables based on the so-called "drop-out-one loss". We prove the oracle property of the proposed method and compare it with several other variable selection techniques using numerical experiments. The results imply that our method is efficient in selecting relevant variables, eliminating irrelevant variables, and making accurate predictions. For future work, it is of interest to generalize our method to select the network structure, since it is shown in (Alvarez & Salzmann, 2016; Scardapane et al., 2017) that network structure is critical in the performance of neural networks. It is also of interest to study on controlled variable selection by controlling the threshold.

Acknowledgements

The authors thank Prof. Faming Liang, David Newton and the 4 anonymous reviewers for valuable comments, which helped to improve the manuscript.

References

Alvarez, J. M. and Salzmann, M. Learning the number of neurons in deep networks. In *Advances in Neural Information Processing Systems*, pp. 2270–2278, 2016.

Bien, J., Taylor, J., and Tibshirani, R. A lasso for hierarchical interactions. *Annals of statistics*, 41(3):1111, 2013.

Bleich, J., Kapelner, A., George, E. I., and Jensen, S. T. Variable selection for bart: An application to gene regulation. *The Annals of Applied Statistics*, pp. 1750–1781, 2014.

Breiman, L. Random forests. *Machine learning*, 45(1): 5–32, 2001.

Candes, E., Fan, Y., Janson, L., and Lv, J. Panning for gold:model-xknockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2018.

Chang, J., Tang, C. Y., and Wu, Y. Marginal empirical likelihood and sure independence feature screening. *Annals of statistics*, 41(4), 2013.

Chouldechova, A. and Hastie, T. Generalized additive model selection. *arXiv preprint arXiv:1506.03850*, 2015.

Couvreur, C. and Bresler, Y. On the optimality of the backward greedy algorithm for the subset selection problem. *SIAM Journal on Matrix Analysis and Applications*, 21 (3):797–808, 2000.

Cui, H., Li, R., and Zhong, W. Model-free feature screening for ultrahigh dimensional discriminant analysis. *Journal of the American Statistical Association*, 110(510):630–641, 2015.

Curtis, C., Shah, S. P., Chin, S.-F., Turashvili, G., Rueda, O. M., Dunning, M. J., Speed, D., Lynch, A. G., Samarajiwa, S., Yuan, Y., et al. The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature*, 486(7403):346, 2012.

Fan, J. and Li, R. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

Fan, J. and Lv, J. Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(5): 849–911, 2008.

Fan, J., Feng, Y., and Song, R. Nonparametric independence screening in sparse ultra-high-dimensional additive models. *Journal of the American Statistical Association*, 106(494):544–557, 2011.

Feng, J. and Simon, N. Sparse-input neural networks for high-dimensional nonparametric regression and classification. *arXiv preprint arXiv:1711.07592*, 2017.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–256, 2010.

Gong, P., Zhang, C., Lu, Z., Huang, J., and Ye, J. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *International Conference on Machine Learning*, pp. 37–45, 2013.

He, X., Wang, L., Hong, H. G., et al. Quantile-adaptive model-free variable screening for high-dimensional heterogeneous data. *The Annals of Statistics*, 41(1):342–369, 2013.

Huang, J., Horowitz, J. L., and Wei, F. Variable selection in nonparametric additive models. *Annals of statistics*, 38 (4):2282, 2010.

Jönsson, G., Staaf, J., Vallon-Christersson, J., Ringnér, M., Holm, K., Hegardt, C., Gunnarsson, H., Fagerholm, R., Strand, C., Agnarsson, B. A., et al. Genomic subtypes of breast cancer identified by array-comparative genomic hybridization display distinct molecular and clinical characteristics. *Breast Cancer Research*, 12(3):R42, 2010.

Liang, F., Li, Q., and Zhou, L. Bayesian neural networks for selection of drug sensitive genes. *Journal of the American Statistical Association*, (just-accepted), 2017.

Lin, Y., Zhang, H. H., et al. Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34(5):2272–2297, 2006.

Meinshausen, N. and Buhlmann, P. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Sohl-Dickstein, J. On the expressive power of deep neural net-works. 2017. URL https://arxiv.org/pdf/1606.05336.pdf.

Ravikumar, P., Liu, H., Lafferty, J., and Wasserman, L. Spam: Sparse additive models. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, pp. 1201–1208. Curran Associates Inc., 2007.

Scardapane, S., Comminiello, D., Hussain, A., and Uncini, A. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.

Sindhwani, V. and Lozano, A. C. Non-parametric group orthogonal matching pursuit for sparse learning with multiple kernels. In *Advances in Neural Information Processing Systems*, pp. 2519–2527, 2011.

Swirszcz, G., Abe, N., and Lozano, A. C. Grouped orthogonal matching pursuit for variable selection and prediction. In *Advances in Neural Information Processing Systems*, pp. 1150–1158, 2009.

Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Yuan, M. and Lin, Y. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68 (1):49–67, 2006.

Zhang, T. On the consistency of feature selection using greedy least squares regression. *Journal of Machine Learning Research*, 10(Mar):555–568, 2009.

Zhang, T. Adaptive forward-backward greedy algorithm for learning sparse representations. *IEEE transactions on information theory*, 57(7):4689–4708, 2011.

Zhao, P. and Yu, B. On model selection consistency of lasso. *Journal of Machine learning research*, 7(Nov): 2541–2563, 2006.