Sparse Algorithms are not Stable: A No-free-lunch Theorem

Huan Xu and Shie Mannor and Constantine Caramanis

***Abstract*— We consider two widely used notions in machine learning, namely: *sparsity* and *algorithmic stability*. Both notions are deemed desirable in designing algorithms, and are believed to lead to good generalization ability. In this paper, we show that these two notions contradict each other. That is, a sparse algorithm can not be stable and vice versa. Thus, one has to tradeoff sparsity and stability in designing a learning algorithm. In particular, our general result implies that ℓ^1-regularized regression (Lasso) cannot be stable, while ℓ^2-regularized regression is known to have strong stability properties.**

I. Introduction

Regression and classification are important problems with impact in a broad range of applications. Given data points encoded by the rows of a matrix A, and observations or labels $\mathbf{b}$, the basic goal is to find a (linear) relationship between A and $\mathbf{b}$. Various objectives are possible, for example in regression, on may consider minimizing the least squared error, $||A\mathbf{x} - \mathbf{b}||_2$, or perhaps in case of a generative model assumption, minimizing the generalization error, i.e., the expected error of the regressor $\mathbf{x}$ on the next sample generated: $\mathbb{E}||\mathbf{a}^\top \mathbf{x} - b||$. In addition to such objectives, one may ask for solutions, $\mathbf{x}$, that have additional structural properties. In the machine learning literature, much work has focused on obtaining solutions with special properties.

Two properties of particular interest are *sparsity* of the solution, and the *stability* of the algorithm. Stability in this context, refers to the property that when given two very similar data sets, an algorithm's output varies little. More specifically, an algorithm is stable if its output changes very little when given two data sets differing on only one sample (this is known as the leave-one-out error). When this difference decays in the number of samples, that decay rate can be used directly to prove good generalization ability [1]. This stability property is also used extensively in the statistical learning community. For example, in [2] the author uses stability properties of ℓ^2-regularized SVM to establish its consistency.

Similarly, numerous algorithms that encourage sparse solutions have been proposed in virtually all fields in machine learning, including Lasso, ℓ_1-SVM, Deep Belief Network, Sparse PCA (cf [3], [4], [5], [6]) and many others, mainly because of the following reasons: (i) a sparse solution is less complicated and hence generalizes well ([7]); (ii) a sparse solution has good interpretability or equivalently less cost associated with the number non-zero loading ([8], [9], [10], [11]); and (iii) sparse algorithms may be computationally much easier to implement, store, compress, etc.

In this paper, we investigate the mutual relationship of these two concepts. In particular, we show that sparse algorithms are not stable: if an algorithm "encourages sparsity" (in a sense defined precisely below) then its sensitivity to small perturbations of the input data remains bounded away from zero, i.e., it has no uniform stability properties. We define these notions exactly and precisely in Section II.

We prove this no-free-lunch theorem by constructing an instance where the leave-one-out error of the algorithm is bounded away from zero by exploiting the property that a sparse algorithm can have non-unique optimal solutions.

This paper is organized as follows. We make necessary definitions in Section II and provide the no-free-lunch theorem based on these definitions in Section III. Sections II and III are devoted to regression algorithms; in Section IV we generalize the theorem to arbitrary loss functions. Concluding remarks are given in Section V.

II. Definitions and Assumptions

The first part of the paper considers regression algorithms that find a weight vector, $\mathbf{x}^*$ in the *feature space*. The goal of any algorithm we consider is to minimize the loss given a new observation $(\hat{b}, \hat{\mathbf{a}})$. Initially we consider the loss function $l(\mathbf{x}^*, (\hat{b}, \hat{\mathbf{a}})) = |\hat{b} - \hat{\mathbf{a}}^\top \mathbf{x}^*|$. Here $\mathbf{a}$ is the vector of feature values of the observation. In the standard regression problem, the learning algorthm $\mathbb{L}$ obtains the candidate solution $\mathbf{x}^*$ by minimizing the empirical loss $||A\mathbf{x} - \mathbf{b}||_2$, or the regularized empirical loss. For a given objective function, we can compare two solutions $\mathbf{x}^1, \mathbf{x}^2$ by considering their empirical loss. We adopt a somewhat more general framework, considering only the partial ordering induced by any learning algorithm $\mathbb{L}$ and training set $(\mathbf{b}, A)$. That is, given two candidate solutions, $\mathbf{x}^1, \mathbf{x}^2$, we write

$$\mathbf{x}^1 \preceq_{(\mathbf{b},A)} \mathbf{x}^2,$$

if on input $(\mathbf{b}, A)$, the algorithm $\mathbb{L}$ would select $\mathbf{x}^2$ before $\mathbf{x}^1$. In short, given an algorith $\mathbb{L}$, each sample set $(\mathbf{b}, A)$ defines an order relationship $\preceq_{(\mathbf{b},A)}$ among all candidate solutions $\mathbf{x}$. This order relationship defines a family of "best" solutions, and one of these, $\mathbf{x}^*$ is the output of the algorithm. We denote this by writing $\mathbf{x}^* \in \mathbb{L}_{(\mathbf{b},A)}$.

Thus, by defining a data-dependent partial ordering on the space of solutions, we can speak more generically of

This work was partially supported by NSF Grants EFRI-0735905, CNS-0721532, a grant from DTRA, the Natural Sciences and Engineering Research Council of Canada and the Canada Research Chairs Program.

Huan Xu and Shie Mannor are with the department of Electrical and Computer Engineering, McGill University, Montreal, Canada xuhuan@cim.mcgill.ca, shie.mannor@mcgill.ca.

Constantine Caramanis is with the Department of Electrical and Comptuer Engineering, The University of Texas at Austin, Austin, TX USA, and part of the Wireless Communications and Networking Group (WNCG). caramanis@mail.utexas.edu.

algorithms, their stability, and their sparseness. As we define below, an algorithm $\mathbb{L}$ is sparse if the set $\mathbb{L}_{(\mathbf{b},A)}$ of optimal solutions contains a sparse solution, and an algorithm is stable if the sets $\mathbb{L}_{(\mathbf{b},A)}$ and $\mathbb{L}_{(\hat{\mathbf{b}},\hat{A})}$ do not contain solutions that are very far apart, when $(\mathbf{b},A)$ and $(\hat{\mathbf{b}},\hat{A})$ differ on only one point.

We make a few assumptions on the preference ordering, and hence on the algorithms that we consider:

Assumption 1. *(i) Given j, $\mathbf{b}$, A, $\mathbf{x}^1$ and $\mathbf{x}^2$, if*

$$\mathbf{x}^1 \preceq_{(\mathbf{b},A)} \mathbf{x}^2,$$

and

$$x^1_j = x^2_j = 0,$$

then for any $\hat{\mathbf{a}}$,

$$\mathbf{x}^1 \preceq_{(\mathbf{b},\hat{A})} \mathbf{x}^2,$$

where

$$\hat{A} = (\mathbf{a}_1, \cdots, \mathbf{a}_{j-1}, \hat{\mathbf{a}}, \mathbf{a}_{j+1}, \cdots, \mathbf{a}_m).$$

(ii) Given $\mathbf{b}$, A, $\mathbf{x}^1$, $\mathbf{x}^2$, b' and $\mathbf{z}$, if

$$\mathbf{x}^1 \preceq_{(\mathbf{b},A)} \mathbf{x}^2,$$

and

$$b = \mathbf{z}^\top \mathbf{x}^2,$$

then

$$\mathbf{x}^1 \preceq_{(\overline{\mathbf{b}},\overline{A})} \mathbf{x}^2,$$

where

$$\overline{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \quad \overline{A} = \begin{pmatrix} A \\ \mathbf{z}^\top \end{pmatrix}.$$

(iii) Given j, $\mathbf{b}$, A, $\mathbf{x}^1$ and $\mathbf{x}^2$, if

$$\mathbf{x}^1 \preceq_{(\mathbf{b},A)} \mathbf{x}^2,$$

then

$$\hat{\mathbf{x}}^1 \preceq_{(\mathbf{b},\tilde{A})} \hat{\mathbf{x}}^2,$$

where

$$\hat{\mathbf{x}}^i = \begin{pmatrix} \mathbf{x}^i \\ 0 \end{pmatrix}, \ i=1,2; \quad \tilde{A} = (A, \mathbf{0}).$$

(iv) Given $\mathbf{b}$, A, $\mathbf{x}^1$, $\mathbf{x}^2$ and $P \in \mathbb{R}^{m\times m}$ a permutation matrix, if

$$\mathbf{x}^1 \preceq_{(\mathbf{b},A)} \mathbf{x}^2,$$

then

$$P^\top \mathbf{x}^1 \preceq_{(\mathbf{b},AP)} P^\top \mathbf{x}^2.$$

Part (i) says that the value of a column corresponding to a non-selected feature has no effect on the ordering; (ii) says that adding a sample that is perfectly predicted by a particular solution, cannot decrease its place in the partial ordering; (iii) says the order relationship is preserved when a trivial (all zeros) feature is added; (iv) says that the partial ordering and hence the algorithm, is feature-wise symmetric. These assumptions are intuitively appealing and satisfied by most algorithms including, for instance, standard regression, and regularized regression.

In what follows, we will define precisely what we mean by stability and sparseness. We recall the definition of uniform (algorithmic) stability first, as given in [1]. We let $\mathcal{Z}$ denote the space of points and labels (typically this will be a compact subset of $\mathbb{R}^{m+1}$) so that $S \in \mathcal{Z}^n$ denotes a collection of n labelled training points. For regression problems, therefore, we have $S = (\mathbf{b},A) \in \mathcal{Z}^n$. We let $\mathbb{L}$ denote a learning algorithm, and for $(\mathbf{b},A) \in \mathcal{Z}^n$, we let $\mathbb{L}_{(\mathbf{b},A)}$ denote the output of the learning algorithm (i.e., the regression function it has learned from the training data). Then given a loss function l, and a labelled point $s = (\mathbf{z}, b) \in \mathcal{Z}$, $l(\mathbb{L}_{(\mathbf{b},A)}, s)$ denotes the loss of the algorithm that has been trained on the set $(\mathbf{b},A)$, on the data point s. Thus for squared loss, we would have $l(\mathbb{L}_{(\mathbf{b},A)}, s) = \|\mathbb{L}_{(\mathbf{b},A)}(\mathbf{z}) - b\|_2$.

Definition 1. *An algorithm $\mathbb{L}$ has uniform stability β_n with respect to the loss function l if the following holds:*

$$\forall (\mathbf{b},A) \in \mathcal{Z}^n, \forall i \in \{1,\cdots,n\},$$
$$\|l(\mathbb{L}_{(\mathbf{b},A)},\cdot) - l(\mathbb{L}_{(\mathbf{b},A)^{\setminus i}},\cdot)\|_\infty \le \beta_n.$$

Here $\mathbb{L}_{(\mathbf{b},A)^{\setminus i}}$ stands for the learned solution with the i^{th} sample removed from $(\mathbf{b},A)$, i.e., with the i^{th} row of A and the i^{th} element of $\mathbf{b}$ removed.

At first glance, this definition may seem too stringent for any reasonable algorithm to exhibit good stability properties. However, as shown in [1], *Tikhonov-regularized regression has stability that scales as $1/n$.* Stability can be used to establish strong PAC bounds. For example, in [1] they show that if we have n samples, β_n denotes the uniform stability, and M a bound on the loss, then

$$R \le R_{\text{emp}} + 2\beta_n + (4n\beta_n + M)\sqrt{\frac{\ln 1/\delta}{2n}},$$

where R denotes the Bayes loss, and R_{emp} the empirical loss.

Since Lasso is an example of an algorithm that yields sparse solutions, one implication of the results of this paper is that while ℓ^2-regularized regression yields sparse solutions, ℓ^1-regularized regression does not. We show that the stability parameter of Lasso does not decrease in the number of samples (compared to the $O(1/n)$ decay for ℓ^2-regularized regression). In fact, we show that Lasso's stability is, in the following sense, as bad as it gets. To this end, we define the notion of the trivial bound, which is the worst possible error a training algorithm can have for arbitrary training set and testing sample labelled by zero.

Definition 2. *Given a subset from which we can draw n labelled points, $\mathcal{Z} \subseteq \mathbb{R}^{n\times(m+1)}$ and a subset for one unlabelled point, $\mathcal{X} \subseteq \mathbb{R}^n$, a trivial bound for a learning algorithm $\mathbb{L}$ w.r.t. $\mathcal{Z}$ and $\mathcal{X}$ is*

$$\mathfrak{b}(\mathbb{L},\mathcal{Z},\mathcal{X}) \triangleq \max_{(\mathbf{b},A)\in\mathcal{Z},\mathbf{z}\in\mathcal{X}} l\big(\mathbb{L}_{(\mathbf{b},A)},(\mathbf{z},0)\big).$$

As above, $l(\cdot,\cdot)$ is a given loss function.

Notice that the trivial bound does not diminish as the number of samples, n, increases, since by repeatedly choosing the worst sample, the algorithm will yield the same solution.

Our next definition makes precise the notion of sparsity of an algorithm which we use.

Definition 3. *An algorithm $\mathbb{L}$ is said to* identify redundant features *if given $(\mathbf{b}, A)$, there exists $\mathbf{x}^* \in \mathbb{L}_{(\mathbf{b},A)}$ such that if $\mathbf{a}_i = \mathbf{a}_j$, then not both x_i and x_j are nonzero. That is,*

$$\forall i \neq j, \quad \mathbf{a}_i = \mathbf{a}_j \Rightarrow x_i^* x_j^* = 0.$$

Identifying redundant features means that at least one solution of the algorithm does not select both features if they are identical. We note that this is a quite weak notion of sparsity. An algorithm that achieves reasonable sparsity (such as Lasso) should be able to identify redundant features.

III. Main Theorem

The next theorem is the main contribution of this paper. It says that if an algorithm is sparse, in the sense that it identifies redundant features as in the definition above, then that algorithm *is not stable*. One notable example that satisfies this theorem is Lasso.

Theorem 1. *Let $\mathcal{Z} \subseteq \mathbb{R}^{n\times(m+1)}$ denote the domain of sample sets of n points each with m features, and $\mathcal{X} \subseteq \mathbb{R}^{m+1}$ the domain of new observations consisting of a point in $\mathbb{R}^m$, and its label in $\mathbb{R}$. Similarly, let $\hat{\mathcal{Z}} \subseteq \mathbb{R}^{n\times(2m+1)}$ be the domain of sample sets of n points each with $2m$ features, and $\hat{\mathcal{X}} \subseteq \mathbb{R}^{2m+1}$ be the domain of new observations. Suppose that these sets of samples and observations are such that:*

$$(\mathbf{b}, A) \in \mathcal{Z} \Longrightarrow (\mathbf{b}, A, A) \in \hat{\mathcal{Z}}$$
$$(0, \mathbf{z}^\top) \in \mathcal{X} \Longrightarrow (0, \mathbf{z}^\top, \mathbf{z}^\top) \in \hat{\mathcal{X}}.$$

If a learning algorithm $\mathbb{L}$ satisfies Assumption 1 and identifies redundant features, its uniform stability bound β is lower bounded by $\mathfrak{b}(\mathbb{L}, \mathcal{Z}, \mathcal{X})$, and in particular does not go to zero with n.

Proof: Let $(\mathbf{b}, A)$ and $(0, \mathbf{z}^\top)$ be the sample set and the new observation such that they jointly achieve $\mathfrak{b}(\mathbb{L}, \mathcal{Z}, \mathcal{X})$, i.e., for some $\mathbf{x}^* \in \mathbb{L}(\mathbf{b}, A)$, we have

$$\mathfrak{b}(\mathbb{L}, \mathcal{Z}, \mathcal{X}) = l\big(\mathbf{x}^*, (0, \mathbf{z})\big). \tag{1}$$

Let $0^{n\times m}$ be the $n \times m$ 0-matrix, and $\mathbf{0}$ stand for the zero vector of length m. We denote

$$\hat{\mathbf{z}} \triangleq (\mathbf{0}^\top, \mathbf{z}^\top); \qquad \hat{A} \triangleq (A,\, A);$$
$$\tilde{\mathbf{b}} \triangleq \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \qquad \tilde{A} \triangleq \begin{pmatrix} A, & A \\ \mathbf{0}^\top, & \mathbf{z}^\top \end{pmatrix}.$$

We first show that

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{x}^* \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}; \quad \begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}. \tag{2}$$

Notice that $\mathbb{L}$ is feature-wise symmetric and identifies redundant features, hence there exists a $\mathbf{x}'$ such that

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{x}' \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.$$

Since $\mathbf{x}^* \in \mathbb{L}_{(\mathbf{b},A)}$, we have

$$\begin{aligned} & \mathbf{x}' \preceq_{(\mathbf{b},A)} \mathbf{x}^* \\ \Rightarrow \; & \begin{pmatrix} \mathbf{0} \\ \mathbf{x}' \end{pmatrix} \preceq_{(\mathbf{b},(0^{n\times m},A))} \begin{pmatrix} \mathbf{0} \\ \mathbf{x}^* \end{pmatrix} \\ \Rightarrow \; & \begin{pmatrix} \mathbf{0} \\ \mathbf{x}' \end{pmatrix} \preceq_{(\mathbf{b},\hat{A})} \begin{pmatrix} \mathbf{0} \\ \mathbf{x}^* \end{pmatrix} \\ \Rightarrow \; & \begin{pmatrix} \mathbf{0} \\ \mathbf{x}^* \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}. \end{aligned}$$

The first impication follows from Assumption 1(iii), and the second from (i).

Now notice by feature-wise symmetry, we have

$$\begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.$$

Furthermore,

$$0 = (\mathbf{0}^\top, \mathbf{z}^\top) \begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix},$$

and thus by Assumption 1(ii) we have

$$\begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}.$$

Hence (2) holds. This leads to

$$l\big(\mathbb{L}_{(\mathbf{b},\hat{A})}, (0, \hat{\mathbf{z}})\big) = l(\mathbf{x}^*, (0, \mathbf{z})); \quad l\big(\mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}, (0, \hat{\mathbf{z}})\big) = 0.$$

By definition of the uniform bound, we have

$$\beta \geq l\big(\mathbb{L}_{(\mathbf{b},\hat{A})}, (0, \hat{\mathbf{z}})\big) - l\big(\mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}, (0, \hat{\mathbf{z}})\big).$$

Hence by (1) we have $\beta \geq \mathfrak{b}(\mathbb{L}, \mathcal{Z}, \mathcal{X})$, which establishes the theorem.

Theorem 1 not only means that a sparse algorithm is not stable, it also states that, if an algorithm is stable, there is no hope to achieve satisfactory sparsity, since it cannot even identify redundant features. Note that indeed, l_2 regularized regression is stable, and does not identify redundant features.

IV. Generalization to Arbitrary Loss

The results derived so far can easily be generalized to algorithms with arbitrary loss function $l(\mathbf{x}^*, (\hat{b}, \hat{\mathbf{a}})) = f_m(\hat{b}, \hat{a}_1 x_i^*, \cdots, \hat{a}_m x_m^*)$ for some f_m. Here, $\hat{a}_i$ and x_i^* denote the i^{th} component of $\hat{\mathbf{a}} \in \mathbb{R}^m$ and $\mathbf{x}^* \in \mathbb{R}^m$, respectively. We assume that the function $f_m(\cdot)$ satisfies the following conditions

$$\begin{aligned} (a) \quad & f_m(b, v_1, \cdots, v_i, \cdots, v_j, \cdots v_m) = \\ & \qquad f_m(b, v_1, \cdots, v_j, \cdots, v_i, \cdots v_m);\ \forall b, \mathbf{v}, i, j. \\ (b) \quad & f_m(b, v_1, \cdots, v_m) = f_{m+1}(b, v_1, \cdots, v_m, 0);\ \forall b, \mathbf{v}. \end{aligned} \tag{3}$$

We require following modifications of Assumption 1(ii) and Definition 2.

Assumption 2. *(ii) Given $\mathbf{b}$, A, $\mathbf{x}^1$, $\mathbf{x}^2$, b' and $\mathbf{z}$ if*

$$\mathbf{x}^1 \preceq_{(\mathbf{b},A)} \mathbf{x}^2, \quad l(\mathbf{x}^2, (b', \mathbf{z})) \leq l(\mathbf{x}^1, (b', \mathbf{z}))$$

then

$$\mathbf{x}^1 \preceq_{(\overline{\mathbf{b}},\overline{A})} \mathbf{x}^2, \quad \textit{where } \overline{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \quad \overline{A} = \begin{pmatrix} A \\ \mathbf{z}^\top \end{pmatrix}.$$

Definition 4. *Given $\mathcal{Z} \subseteq \mathbb{R}^{n\times(m+1)}$ and $\mathcal{X} \subseteq \mathbb{R}^{m+1}$, a trivial bound for a learning algorithm $\mathbb{L}$ w.r.t. $\mathcal{Z}$ and $\mathcal{X}$ is*

$$\hat{\mathfrak{b}}(\mathbb{L}, \mathcal{Z}, \mathcal{X}) \triangleq \max_{(\mathbf{b},A)\in\mathcal{Z},(b,\mathbf{z})\in\mathcal{X}} \Big\{ l\big(\mathbb{L}_{(\mathbf{b},A)}, (b, \mathbf{z})\big) - l\big(\mathbf{0}, (b,\, \mathbf{z})\big)\Big\}.$$

These modifications account for the fact that under an arbitrary loss function, there may not exist a sample that can be perfectly predicted by the zero vector. With these modifications, we have the same no-free-lunch theorem.

Theorem 2. *As before, let $\mathcal{Z} \subseteq \mathbb{R}^{n\times(m+1)}$, and $\hat{\mathcal{Z}} \subseteq \mathbb{R}^{n\times(2m+1)}$ be the domain of sample sets, and $\mathcal{X} \subseteq \mathbb{R}^{m+1}$, and $\hat{\mathcal{X}} \subseteq \mathbb{R}^{2m+1}$ be the domain of new observations, with m and $2m$ features respectively. Suppose, as before, that these sets satisfy*

$$(\mathbf{b}, A) \in \mathcal{Z} \Longrightarrow (\mathbf{b}, A, A) \in \hat{\mathcal{Z}}$$
$$(b', \mathbf{z}^\top) \in \mathcal{X} \Longrightarrow (b', \mathbf{z}^\top, \mathbf{z}^\top) \in \hat{\mathcal{X}}.$$

If a learning algorithm $\mathbb{L}$ satisfies Assumption 2 and identifies redundant features, its uniform stability bound β is lower bounded by $\hat{\mathfrak{b}}(\mathbb{L}, \mathcal{Z}, \mathcal{X})$.

Proof: This proof follows a similar line of reasoning as the proof of Theorem 1. Let $(\mathbf{b}, A)$ and $(b', \mathbf{z}^\top)$ be the sample set and the new observation such that they jointly achieve $\hat{\mathfrak{b}}(\mathbb{L}, \mathcal{Z}, \mathcal{X})$, i.e., let $\mathbf{x}^* \in \mathbb{L}(\mathbf{b}, A)$, and

$$\begin{aligned}\hat{\mathfrak{b}}(\mathbb{L}, \mathcal{Z}, \mathcal{X}) &= l\big(\mathbf{x}^*, (b', \mathbf{z})\big) - l\big(\mathbf{0}, (b',\, \mathbf{z})\big)\\ &= f_m(b', x_1^* z_1, \cdots, x_m^* z_m) - f(b', 0, \cdots, 0).\end{aligned}$$

Let $0^{n\times m}$ be the $n \times m$ 0-matrix, and $\mathbf{0}$ stand for the zero vector of length m. We denote

$$\hat{\mathbf{z}} \triangleq (\mathbf{0}^\top,\, \mathbf{z}^\top); \qquad \hat{A} \triangleq (A,\, A);$$
$$\tilde{\mathbf{b}} \triangleq \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \qquad \tilde{A} \triangleq \begin{pmatrix} A, & A \\ \mathbf{0}^\top, & \mathbf{z}^\top \end{pmatrix}.$$

To prove the theorem, it suffices show that there exists $\mathbf{x}^1$, $\mathbf{x}^2$ such that

$$\mathbf{x}^1 \in \mathbb{L}_{(\mathbf{b},\hat{A})}; \quad \mathbf{x}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})},$$

and

$$l\big(\mathbf{x}^1, (b', \hat{\mathbf{z}})\big) - l\big(\mathbf{x}^2, (b', \hat{\mathbf{z}})\big) \geq \hat{\mathfrak{b}}(\mathbb{L}, \mathcal{Z}, \mathcal{X})$$

where again,

$$\hat{\mathfrak{b}}(\mathbb{L}, \mathcal{Z}, \mathcal{X}) = f_m(b', x_1^* z_1, \cdots, x_m^* z_m) - f_m(b', 0, \cdots, 0).$$

By an identical argument as that of Proof 1, we have

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{x}^* \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.$$

Hence

$$\exists \mathbf{x}^1 \in \mathbb{L}_{(\mathbf{b},\hat{A})}: \tag{4}$$
$$\begin{aligned} l\left(\mathbf{x}^1, (b', \hat{\mathbf{z}})\right) &= l\left(\begin{pmatrix} \mathbf{0} \\ \mathbf{x}^* \end{pmatrix}, (b', \hat{\mathbf{z}})\right) \qquad (5)\\ &= f_m(b', x_1^* z_1, \cdots, x_1^* z_m). \end{aligned}$$

The last equality follows from Equation (3) easily.

Now notice that by feature-wise symmetry, we have

$$\begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.$$

Hence

$$\exists \mathbf{x}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}: \tag{6}$$
$$\begin{aligned} l\left(\mathbf{x}^2, (b', \hat{\mathbf{z}})\right) &\leq l\left(\begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix}, (b', \hat{\mathbf{z}})\right) \qquad (7)\\ &= f_m(b', 0, \cdots, 0). \end{aligned}$$

The last equality follows from Equation (3). The inequality here holds because by Assumption 2(ii), if there is no $\mathbf{x}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}$ that satisfies the inequality, then we have

$$\mathbf{x}^2 \preceq_{(\tilde{\mathbf{b}},\tilde{A})} \begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix}$$

which then implies that

$$\Rightarrow \quad \begin{pmatrix} \mathbf{x}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})},$$

which is absurd.

Combining (4) and (6) proves the theorem.

V. Conclusion

In this paper, we prove a no-free-lunch theorem show that sparsity and stability are at odds with each other. We show that if an algorithm is sparse, then its uniform stability is lower bounded by a nonzero constant. This also shows that any stable algorithm cannot be sparse. Thus we show that these two widely used concepts, namely *sparsity* and *algorithmic stability* conflict with each other. At a high level, this theorem provides us with additional insight into these concepts and their interrelation, and it furthermore implies that, a tradeoff between these two concepts is unavoidable in designing learning algorithms. On the other hand, given that both sparsity and stability are desirable properties, one interesting direction is to understand the full implications of having one of them. That is, what other properties must a sparse solution have? Given that sparse algorithms often perform well and have strong statistical behavior, one may further ask for other significant notions of stability that are not in conflict with sparsity.

References

[1] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

[2] I. Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

[3] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[4] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Advances in Neural Information Processing Systems 16*, 2003.

[5] G. Hinto and R. Salakhutdinov. Reducing the dimensionality of data with nerual networks. *Science*, 313:504–507, 2006.

[6] A. d'Aspremont, L El Ghaoui, M. Jordan, and G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

[7] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1445–1480, 1998.

[8] R. R. Coifman and M. V. Wickerhauser. Entropy-based algorithms for best-basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, 1992.

[9] S. Mallat and Z. Zhang. Matching Pursuits with time-frequence dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[10] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[11] D. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.