Learning Sparse Classifiers: Continuous and Mixed Integer Optimization Perspectives

Antoine Dedieu TONIO.DEDIEU@GMAIL.COM
Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Hussein Hazimeh HAZIMEH@MIT.EDU
Operations Research Center
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Rahul Mazumder RAHULMAZ@MIT.EDU
Sloan School of Management
Operations Research Center
MIT Center for Statistics
Massachusetts Institute of Technology
Cambridge, MA 02142, USA

Abstract

We consider a discrete optimization formulation for learning sparse classifiers, where the outcome depends upon a linear combination of a small subset of features. Recent work has shown that mixed integer programming (MIP) can be used to solve (to optimality) ℓ_0-regularized regression problems at scales much larger than what was conventionally considered possible. Despite their usefulness, MIP-based global optimization approaches are significantly slower compared to the relatively mature algorithms for ℓ_1-regularization and heuristics for nonconvex regularized problems. We aim to bridge this gap in computation times by developing new MIP-based algorithms for ℓ_0-regularized classification. We propose two classes of scalable algorithms: an exact algorithm that can handle $p \approx 50,000$ features in a few minutes, and approximate algorithms that can address instances with $p \approx 10^6$ in times comparable to the fast ℓ_1-based algorithms. Our exact algorithm is based on the novel idea of *integrality generation*, which solves the original problem (with p binary variables) via a sequence of mixed integer programs that involve a small number of binary variables. Our approximate algorithms are based on coordinate descent and local combinatorial search. In addition, we present new estimation error bounds for a class of ℓ_0-regularized estimators. Experiments on real and synthetic data demonstrate that our approach leads to models with considerably improved statistical performance (especially, variable selection) when compared to competing methods.

Keywords: sparsity, sparse classification, l0 regularization, mixed integer programming

1. Introduction

We consider the problem of sparse linear classification, where the output depends upon a linear combination of a small subset of features. This is a core problem in high-dimensional

statistics (Hastie et al., 2015) where the number of features p is comparable to or exceeds the number of samples n. In such settings, sparsity can be useful from a statistical viewpoint and can lead to more interpretable models. We consider the typical binary classification problem with samples $(\mathbf{x}_i, y_i), i = 1, \ldots, n$, features $\mathbf{x}_i \in \mathbb{R}^p$, and outcome $y_i \in \{-1, +1\}$. In the spirit of best-subset selection in linear regression (Miller, 2002), we consider minimizing the empirical risk (i.e., a surrogate for the misclassification error) while penalizing the number of nonzero coefficients:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta} \rangle, y_i\right) + \lambda_0 \|\boldsymbol{\beta}\|_0, \tag{1}$$

where $f : \mathbb{R} \times \{-1, +1\} \to \mathbb{R}$ is the loss function (for example, hinge or logistic loss). The term $\|\boldsymbol{\beta}\|_0$ is the ℓ_0 (pseudo)-norm of $\boldsymbol{\beta}$ which is equal to the number of nonzeros in $\boldsymbol{\beta}$, and $\lambda_0 > 0$ is a regularization parameter which controls the number of nonzeros in $\boldsymbol{\beta}$. We ignore the intercept term in the above and throughout the paper to simplify the presentation. Problem (1) is known to be NP-Hard and poses computational challenges (Natarajan, 1995). In this paper, we introduce scalable algorithms for this optimization problem using techniques based on both continuous and discrete optimization, specifically, mixed integer programming (Wolsey and Nemhauser, 1999).

There is an impressive body of work on obtaining approximate solutions to Problem (1): popular candidates include greedy (a.k.a. stepwise) procedures (Bahmani et al., 2013), proximal gradient methods (Blumensath and Davies, 2009), among others. The ℓ_1-norm (Tibshirani, 1996) is often used as a convex surrogate to the ℓ_0-norm, leading to a convex optimization problem. Nonconvex continuous penalties (such as MCP and SCAD) (Zhang, 2010) provide better approximations of the ℓ_0-penalty but lead to nonconvex problems, for which gradient-based methods (Gong et al., 2013; Parikh and Boyd, 2014; Li and Lin, 2015) and coordinate descent (Breheny and Huang, 2011; Mazumder et al., 2011) are often used. These algorithms may not deliver optimal solutions for the associated nonconvex problem. Fairly recently, there has been considerable interest in exploring Mixed Integer Programming (MIP)-based methods (Wolsey and Nemhauser, 1999; Bertsimas et al., 2016; Ustun and Rudin, 2016; Sato et al., 2016) to solve variants of Problem (1) to optimality. MIP-based methods create a branch-and-bound tree that simultaneously leads to feasible solutions and corresponding lower-bounds (a.k.a. dual bounds). Therefore, these methods deliver optimality certificates for the nonconvex optimization problem. Despite their appeal in delivering nearly optimal solutions to Problem (1), MIP-based algorithms are usually computationally expensive compared to convex relaxations or greedy (heuristic) algorithms (Hazimeh and Mazumder, 2020; Hastie et al., 2020)—possibly limiting their use in time-sensitive applications that arise in practice.

The vanilla version of best-subset selection is often perceived as a gold-standard for high-dimensional sparse linear regression, when the signal-to-noise ratio (SNR) is high. However, it suffers from overfitting when the SNR becomes moderately low (Friedman et al., 2001; Mazumder et al., 2017; Hastie et al., 2020). A possible way to mitigate this shortcoming is by imposing additional continuous regularization—see for example, Mazumder et al. (2017); Hazimeh and Mazumder (2020) for studies in the (linear) regression setting. Thus, we consider an extended family of estimators which combines ℓ_0 and ℓ_q (for $q \in \{1, 2\}$)

regularization:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \quad \frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}\rangle, y_i\right) + \lambda_0\|\boldsymbol{\beta}\|_0 + \lambda_q\|\boldsymbol{\beta}\|_q^q, \tag{2}$$

where the regularization parameter $\lambda_0 \geq 0$ explicitly controls the sparsity in $\boldsymbol{\beta}$, and $\lambda_q \geq 0$ controls the amount of continuous shrinkage on the nonzero coefficients of $\boldsymbol{\beta}$ (for example, the margin in linear SVM). In what follows, for notational convenience, we will refer to the combination of regularizers $\lambda_0\|\boldsymbol{\beta}\|_0$ and $\lambda_q\|\boldsymbol{\beta}\|_q^q$, as the ℓ_0-ℓ_q penalty. For flexibility, our framework allows for both choices of $q \in \{1, 2\}$, and the value of q needs to be specified a-priori by the practitioner. When $q = 2$, Problem (2) seeks to deliver a solution $\boldsymbol{\beta}$ with few nonzeros (controlled by the ℓ_0-penalty) and a small ℓ_2-norm (controlled by the ridge penalty). Similarly, when $q = 1$, we seek a model $\boldsymbol{\beta}$ that has a small ℓ_1-norm and a small ℓ_0-norm. If λ_1 is large, the ℓ_1-penalty may also encourage zeros in the coefficients. Note that the primary role of the ℓ_0-penalty is to control the number of nonzeros in $\boldsymbol{\beta}$; and that of the ℓ_1-penalty is to shrink the model coefficients. In our numerical experiments we observe that both choices of $q \in \{1, 2\}$ work quite well, with no penalty uniformly dominating the other. We refer the reader to Mazumder et al. (2017) for complementary discussions in the regression setting.

A primary focus of our work is to propose new scalable algorithms for solving Problem (2), with certificates of optimality (suitably defined). Problem (2) can be expressed using MIP formulations. However, these formulations lead to computational challenges for off-the-shelf commercial MIP solvers (such as Gurobi and CPLEX). To this end, we propose a new MIP-based algorithm that we call "integrality generation", which allows for solving instances of Problem (2) with $p \approx 50,000$ (where n is small) to optimality within a few minutes.[1] This appears to be well beyond the capabilities of state-of-the-art MIP solvers, including recent MIP-based approaches, as outlined below. To obtain high-quality solutions for larger problem instances, in times comparable to the fast ℓ_1-based solvers (Friedman et al., 2010), we propose approximate algorithms based on coordinate descent (CD) (Wright, 2015) and local combinatorial optimization,[2] where the latter leads to higher quality solutions compared to CD. Our CD and local combinatorial optimization algorithms are publicly available through our fast C++/R toolkit `L0Learn`: on CRAN at `https://cran.r-project.org/package=L0Learn` and also at `https://github.com/hazimehh/L0Learn`.

From a statistical viewpoint, we establish new upper bounds on the estimation error for solutions obtained by globally minimizing ℓ_0-based estimators (2). These error bounds (rates) appear to be better than current known bounds for ℓ_1-regularization; and have rates similar to the optimal minimax rates for sparse least squares regression (Raskutti et al., 2011), achieved by ℓ_0-based regression procedures.

Related Work and Contributions: There is a vast body of work on developing optimization algorithms and understanding the statistical properties of various sparse estimators (Hastie et al., 2015; Bühlmann and Van De Geer, 2011). We present a brief overview of work that relates to our paper.

1. Empirically, we observe the runtime to depend upon the number of nonzeros in the solution. The runtime can increase if the number of nonzeros in an optimal solution becomes large—see Section 5 for details.
2. The local combinatorial optimization algorithms are based on solving MIP problems over restricted search-spaces; and are usually much faster to solve compared to the full problem (2).

Computation: An impressive body of work has developed fast algorithms for minimizing the empirical risk regularized with convex or nonconvex proxies to the ℓ_0-norm, e.g., Friedman et al. (2010); Breheny and Huang (2011); Mazumder et al. (2011); Nesterov (2013); Shalev-Shwartz and Zhang (2012). Below, we discuss related work that directly optimize objective functions involving an ℓ_0 norm (in the objective or as a constraint).

Until recently, global optimization with ℓ_0-penalization was rarely used beyond $p = 30$ as popular software packages for best-subset selection (for example, `leaps` and `bestglm`) are unable to handle larger instances. Bertsimas et al. (2016) demonstrated that ℓ_0-regularized regression problems could be solved to near-optimality for $p \approx 10^3$ by leveraging advances in first-order methods and the capabilities of modern MIP solvers such as Gurobi. Bertsimas and King (2017); Sato et al. (2016) extend the work of Bertsimas et al. (2016) to solve ℓ_0-regularized logistic regression by using an outer-approximation approach that can address problems with p in the order of a few hundreds. Bertsimas et al. (2020) propose a cutting plane algorithm for the ℓ_0-constrained least squares problem with additional ridge regularization—they can handle problems with $n \approx p$, when the feature correlations are low and/or the amount of ridge regularization is taken to be sufficiently large. Bertsimas et al. (2017) adapt Bertsimas et al. (2020)'s work to solve classification problems (e.g., with logistic or hinge loss). The approach of Bertsimas et al. (2017) appears to require a fairly high amount of ridge regularization for the cutting plane algorithm to work well.

A separate line of research investigates algorithms to obtain feasible solutions for ℓ_0-regularized problems. These algorithms do not provide dual bounds like MIP-based algorithms, but can be computationally much faster. These include: (i) first-order optimization algorithms based on hard thresholding, such as Iterative Hard Thresholding (IHT) (Blumensath and Davies, 2009) and GraSP (Bahmani et al., 2013), (ii) second-order optimization algorithms inspired by the Newton method such as NTGP (Yuan and Liu, 2017), NHTP (Zhou et al., 2021), and NSLR (Wang et al., 2019), and (iii) coordinate descent methods based on greedy and random coordinate selection rules (Beck and Eldar, 2013; Patrascu and Necoara, 2015).

Hazimeh and Mazumder (2020) present algorithms that offer a *bridge* between MIP-based global optimization and good feasible solutions for ℓ_0-regularized problems, by using a combination of CD and local combinatorial optimization. The current paper is similar in spirit, but makes new contributions. We extend the work of Hazimeh and Mazumder (2020) (which is tailored to the least squares loss function) to address the more general class of problems in (2). Our algorithms can deliver solutions with better statistical performance (for example, in terms of variable selection and prediction error) compared to the popular fast algorithms for sparse learning (e.g., based on ℓ_1 and MCP regularizers). Unlike heuristics that simply deliver an upper bound, MIP-based approaches attempt to solve (2) to optimality. They can (i) certify via dual bounds the quality of solutions obtained by our CD and local search algorithms; and (ii) improve the solution if it is not optimal. However, as off-the-shelf MIP-solvers do not scale well, we present a new method: the *Integrality Generation Algorithm* (IGA) (see Section 3) that allows us to *solve* (to optimality) the MIP problems for instances that are larger than current methods (Bertsimas and King, 2017; Bertsimas et al., 2017; Sato et al., 2016). The key idea behind our proposed IGA is to solve a sequence of relaxations of (2) by allowing only a subset of variables to be binary. On the

contrary, a direct MIP formulation for (2) requires p many binary variables; and can be prohibitively expensive for moderate values of p.

Statistical Properties: Statistical properties of high-dimensional linear regression have been widely studied (Candes and Davenport, 2013; Raskutti et al., 2011; Bunea et al., 2007; Candes and Tao, 2007; Bickel et al., 2009). One important statistical performance measure is the ℓ_2-estimation error defined as $\|\boldsymbol{\beta}^* - \hat{\boldsymbol{\beta}}\|_2^2$, where $\boldsymbol{\beta}^*$ is the k-sparse vector used in generating the true model and $\hat{\boldsymbol{\beta}}$ is an estimator. For regression problems, Candes and Davenport (2013); Raskutti et al. (2011) established a $(k/n)\log(p/k)$ lower bound on the ℓ_2-estimation error. This optimal minimax rate is known to be achieved by a global minimizer of an ℓ_0-regularized estimator (Bunea et al., 2007). It is well known that the Dantzig Selector and Lasso estimators achieve a $(k/n)\log(p)$ error rate (Candes and Tao, 2007; Bickel et al., 2009) under suitable assumptions for the high-dimensional regression setting. Compared to regression, there has been limited work in deriving estimation error bounds for classification tasks. Tarigan and Van De Geer (2006) study margin adaptation for ℓ_1-norm SVM. A sizable amount of work focuses on the analysis of generalization error and risk bounds (Greenshtein, 2006; Van de Geer, 2008). Zhang et al. (2016) study variable selection consistency of a nonconvex penalized SVM estimator, using a local linear approximation method with a suitable initialization. Recently, Peng et al. (2016) proved a $(k/n)\log(p)$ upper-bound for the ℓ_2-estimation error of ℓ_1-regularized support vector machines (SVM), where k is the number of nonzeros in the estimator that minimizes the population risk. Ravikumar et al. (2010) show consistent neighborhood selection for high-dimensional Ising model using an ℓ_1-regularized logistic regression estimator. Plan and Vershynin (2013) show that one can obtain an error rate of $k/n\log(p/k)$ for 1-bit compressed sensing problems. In this paper, we present (to our knowledge) new ℓ_2-estimation error bounds for a (global) minimizer of Problem (1)—our framework applies to a family of loss functions including the hinge and logistic loss functions.

Our Contributions: We summarize our contributions below:

- We develop fast first-order algorithms based on cyclic CD and local combinatorial search to (approximately) solve Problem (2) (see Section 2). We prove a new result which establishes the convergence of cyclic CD under an asymptotic linear rate. We show that combinatorial search leads to solutions of higher quality than IHT and CD-based methods. We discuss how solutions from the ℓ_0-penalized formulation, i.e., Problem (2), can be used to obtain solutions to the cardinality constrained variant of (2). We open source these algorithms through our sparse learning toolkit `L0Learn`.

- We propose a new algorithm: IGA, for solving Problem (2) to optimality. On some problems, our algorithm reduces the time for solving a MIP formulation of Problem (2) from the order of hours to seconds, and it can solve high-dimensional instances with $p \approx 50,000$ and small n. The algorithm is presented in Section 3.

- We establish upper bounds on the squared ℓ_2-estimation error for a cardinality constrained variant of Problem (2). Our $(k/n)\log(p/k)$ upper bound matches the optimal minimax rate known for regression.

- On a series of high-dimensional synthetic and real data sets (with $p \approx 10^5$), we show that our proposed algorithms can achieve significantly better statistical performance

in terms of prediction (AUC), variable selection accuracy, and support sizes, compared to state-of-the-art algorithms (based on ℓ_1 and local solutions to ℓ_0 and MCP regularizers). Our proposed CD algorithm compares favorably in terms of runtime compared to current popular toolkits (Friedman et al., 2010; Breheny and Huang, 2011; Bahmani et al., 2013; Zhou et al., 2021) for sparse classification.

1.1 Preliminaries and Notation

For convenience, we introduce the following notation:

$$g(\boldsymbol{\beta}) \stackrel{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}\rangle, y_i\right) \quad \text{and} \quad G(\boldsymbol{\beta}) \stackrel{\text{def}}{=} g(\boldsymbol{\beta}) + \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_2^2.$$

Problem (2) is an instance of the following (more general) problem:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} P(\boldsymbol{\beta}) \stackrel{\text{def}}{=} G(\boldsymbol{\beta}) + \lambda_0\|\boldsymbol{\beta}\|_0. \tag{3}$$

In particular, Problem (2) with $q = 1$ is equivalent to Problem (3) with $\lambda_2 = 0$. Similarly, Problem (2) with $q = 2$ is equivalent to Problem (3) with $\lambda_1 = 0$. Problem (3) will be the focus in our algorithmic development.

We denote the set $\{1, 2, \ldots, p\}$ by $[p]$ and the canonical basis for $\mathbb{R}^p$ by $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_p$. For $\boldsymbol{\beta} \in \mathbb{R}^p$, we use $\text{Supp}(\boldsymbol{\beta})$ to denote the support of $\boldsymbol{\beta}$, i.e., the indices of its nonzero entries. For $S \subseteq [p]$, $\boldsymbol{\beta}_S \in \mathbb{R}^{|S|}$ denotes the subvector of $\boldsymbol{\beta}$ with indices in S. Moreover, for a differentiable function $g(\boldsymbol{\beta})$, we use the notation $\nabla_S g(\boldsymbol{\beta})$ to refer to the subvector of the gradient $\nabla g(\boldsymbol{\beta})$ restricted to coordinates in S. We let $\mathbb{Z}$ and $\mathbb{Z}_+$ denote the set of integers and non-negative integers, respectively. A convex function $g(\boldsymbol{\beta})$ is said to be μ-strongly convex if $\boldsymbol{\beta} \mapsto g(\boldsymbol{\beta}) - \mu\|\boldsymbol{\beta}\|_2^2/2$ is convex. A function $h(\boldsymbol{\beta})$ is said to be Lipschitz with parameter L if $\|h(\boldsymbol{\beta}) - h(\boldsymbol{\alpha})\|_2 \leq L\|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2$ for all $\boldsymbol{\beta}, \boldsymbol{\alpha}$ in the domain of the function.

1.2 Examples of Loss Functions Considered

In Table 1, we give examples of popular classification loss functions that fall within the premise of our algorithmic framework and statistical theory. The column "FO & Local Search" indicates whether these loss functions are amenable to our first-order and local search algorithms (discussed in Section 2).[3] The column "MIP" indicates whether the loss function leads to an optimization problem that can be solved (to optimality) via the MIP methods discussed in Section 3. Finally, the column "Error Bounds" indicates if the statistical error bounds (estimation error) discussed in Section 4 apply to the loss function.

2. First-Order and Local Combinatorial Search Algorithms

Here we present fast cyclic CD and local combinatorial search algorithms for obtaining high-quality local minima (we make this notion precise later) for Problem (3). Our framework assumes that $g(\boldsymbol{\beta})$ is differentiable and has a Lipschitz continuous gradient. We first present

3. That is, these algorithms are guaranteed to converge to a stationary point (or a local optimum) for the corresponding optimization problems.

Loss	$f(\hat{v}, v)$	FO & Local Search	MIP	Error Bounds
Logistic	$\log(1+e^{-\hat{v}v})$	✓	✓	✓
Squared Hinge	$\max(0, 1-\hat{v}v)^2$	✓	✓	✗
Hinge	$\max(0, 1-\hat{v}v)$	✓*	✓	✓

Table 1: Examples of loss functions we consider. "*" denotes that our proposed first-order and local search methods apply upon using Nesterov (2012)'s smoothing on the non-smooth loss function.

a brief overview of the key ideas presented in this section, before diving into the technical details.

Due to the nonconvexity of (3), the quality of the solution obtained depends on the algorithm—with local search and MIP-based algorithms leading to solutions of higher quality. The fixed points of the algorithms considered satisfy certain necessary optimality conditions for (3), leading to different classes of local minima. In terms of solution quality, there is a hierarchy among these classes. We show that for Problem (3), the minima corresponding to the different algorithms satisfy the following hierarchy:

$$\text{MIP Minima} \subseteq \text{Local Search Minima} \subseteq \text{CD Minima} \subseteq \text{IHT Minima}. \tag{4}$$

The fixed points of the IHT algorithm contain the fixed points of the CD algorithm. As we move to the left in the hierarchy, the fixed points of the algorithms satisfy stricter necessary optimality conditions. At the top of the hierarchy, we have the global minimizers, which can be obtained by solving a MIP formulation of (3).

Our CD and local search algorithms can run in times comparable to the fast ℓ_1-regularized approaches (Friedman et al., 2010). These algorithms can lead to high-quality solutions that can be used as warm starts for MIP-based algorithms. The MIP framework of Section 3 can be used to certify the quality of these solutions via dual bounds, and to improve over them (if they are sub-optimal).

In Section 2.1, we introduce cyclic CD for Problem (3) and study its convergence properties. Section 2.2 discusses how the solutions of cyclic CD can be improved by local search and presents a fast heuristic for performing local search in high dimensions. In Section 2.3, we discuss how our algorithms can be used to obtain high-quality (feasible) solutions to the *cardinality constrained* counterpart of (3), in which the complexity measure $\|\boldsymbol{\beta}\|_0$ appears as a constraint and not a penalty as in (3). Finally, in Section 2.4, we briefly present implementation aspects of our toolkit `L0Learn`.

2.1 Cyclic Coordinate Descent: Algorithm and Computational Guarantees

We describe a cyclic CD algorithm for Problem (3) and establish its convergence to stationary points of (3).

Why cyclic CD? We briefly discuss our rationale for choosing cyclic CD. Cyclic CD has been shown to be among the fastest algorithms for fitting generalized linear models with convex and nonconvex regularization (e.g., ℓ_1, MCP, and SCAD) (Friedman et al., 2010;

Mazumder et al., 2011; Breheny and Huang, 2011). Indeed, it can effectively exploit sparsity and active-set updates, making it suitable for solving high-dimensional problems (e.g., with $p \sim 10^6$ and small n). Algorithms that require evaluation of the full gradient at every iteration (such as proximal gradient, stepwise, IHT or greedy CD algorithms) have difficulties in scaling with p (Nesterov, 2012). In an earlier work, Patrascu and Necoara (2015) proposed random CD for problems similar to (3) (without an ℓ_1-regularization term in the objective). However, recent studies have shown that cyclic CD can be faster than random CD (Beck and Tetruashvili, 2013; Gurbuzbalaban et al., 2017; Hazimeh and Mazumder, 2020). Furthermore, for ℓ_0-regularized regression problems, cyclic CD is empirically seen to obtain solutions of higher quality (e.g., in terms of optimization and statistical performance) compared to random CD (Hazimeh and Mazumder, 2020).

Setup. Our cyclic CD algorithm for (3) applies to problems where $g(\boldsymbol{\beta})$ is convex, continuously differentiable, and non-negative. Moreover, we will assume that the gradient of $\boldsymbol{\beta} \mapsto g(\boldsymbol{\beta})$ is coordinate-wise Lipschitz continuous, i.e., for every $i \in [p]$, $\boldsymbol{\beta} \in \mathbb{R}^p$ and $s \in \mathbb{R}$, we have:

$$|\nabla_i g(\boldsymbol{\beta} + \boldsymbol{e}_i s) - \nabla_i g(\boldsymbol{\beta})| \leq L_i |s|, \tag{5}$$

where $L_i > 0$ is the Lipschitz constant for coordinate i. This assumption leads to the block Descent Lemma (Bertsekas, 2016), which states that

$$g(\boldsymbol{\beta} + \boldsymbol{e}_i s) \leq g(\boldsymbol{\beta}) + s\nabla_i g(\boldsymbol{\beta}) + \frac{1}{2} L_i s^2. \tag{6}$$

Several popular loss functions for classification fall under the above setup. For example, logistic loss and squared hinge loss satisfy (5) with $L_i = \|\boldsymbol{X}_i\|_2^2/4n$ and $L_i = 2\|\boldsymbol{X}_i\|_2^2/n$, respectively (here, $\mathbf{X}_i$ denotes the i-th column of the data matrix $\mathbf{X}$).

CD Algorithm. Cyclic CD (Bertsekas, 2016) updates one coordinate at a time (with others held fixed) in a cyclical fashion. Given a solution $\boldsymbol{\beta} \in \mathbb{R}^p$, we attempt to find a new solution by changing the i-th coordinate of $\boldsymbol{\beta}$—i.e., we find $\boldsymbol{\alpha}$ such that $\alpha_j = \beta_j$ for all $j \neq i$ and α_i minimizes the one-dimensional function: $\beta_i \mapsto P(\boldsymbol{\beta})$. However, for the examples we consider (e.g., logistic and squared hinge losses), there is no closed-form expression for this minimization problem. This makes the algorithm computationally inefficient compared to g being the squared error loss (Hazimeh and Mazumder, 2020). Using (6), we consider a quadratic upper bound $\tilde{g}(\boldsymbol{\alpha}; \boldsymbol{\beta})$ for $g(\boldsymbol{\alpha})$ as follows:

$$g(\boldsymbol{\alpha}) \leq \tilde{g}(\boldsymbol{\alpha}; \boldsymbol{\beta}) \stackrel{\text{def}}{=} g(\boldsymbol{\beta}) + (\alpha_i - \beta_i)\nabla_i g(\boldsymbol{\beta}) + \frac{\hat{L}_i}{2}(\alpha_i - \beta_i)^2, \tag{7}$$

where $\hat{L}_i$ is a constant which satisfies $\hat{L}_i > L_i$. (For notational convenience, we hide the dependence of $\tilde{g}$ on i.) Let us define the function

$$\psi(\boldsymbol{\alpha}) = \sum_i \psi_i(\alpha_i) \quad \text{where} \quad \psi_i(\alpha_i) = \lambda_0 \mathbb{1}(\alpha_i \neq 0) + \lambda_1 |\alpha_i| + \lambda_2 \alpha_i^2.$$

Adding $\psi(\boldsymbol{\alpha})$ to both sides of equation (7), we get:

$$P(\boldsymbol{\alpha}) \leq \widetilde{P}_{\hat{L}_i}(\boldsymbol{\alpha}; \boldsymbol{\beta}) \stackrel{\text{def}}{=} \tilde{g}(\boldsymbol{\alpha}; \boldsymbol{\beta}) + \psi(\boldsymbol{\alpha}). \tag{8}$$

We can approximately minimize $P(\boldsymbol{\alpha})$ w.r.t. α_i (with other coordinates held fixed) by minimizing its upper bound $\widetilde{P}_{\hat{L}_i}(\boldsymbol{\alpha};\boldsymbol{\beta})$ w.r.t. α_i. A solution $\hat{\alpha}_i$ for this one-dimensional optimization problem is given by

$$\hat{\alpha}_i \in \operatorname*{argmin}_{\alpha_i} \widetilde{P}_{\hat{L}_i}(\boldsymbol{\alpha};\boldsymbol{\beta}) \quad = \quad \operatorname*{argmin}_{\alpha_i} \frac{\hat{L}_i}{2}\left(\alpha_i - \left(\beta_i - \frac{1}{\hat{L}_i}\nabla_i g(\boldsymbol{\beta})\right)\right)^2 + \psi_i(\alpha_i). \tag{9}$$

Let $\hat{\boldsymbol{\alpha}}$ be a vector whose i-th component is $\hat{\alpha}_i$, and $\hat{\alpha}_j = \beta_j$ for all $j \neq i$. Note that $P(\hat{\boldsymbol{\alpha}}) \leq P(\boldsymbol{\beta})$—i.e., updating the i-th coordinate via (9) with all other coefficients held fixed, leads to a decrease in the objective value $P(\boldsymbol{\beta})$. A solution of (9) can be computed in closed-form; and is given by the thresholding operator $T : \mathbb{R} \to \mathbb{R}$ defined as follows:

$$T(c;\boldsymbol{\lambda},\hat{L}_i) = \begin{cases} \frac{\hat{L}_i}{\hat{L}_i+2\lambda_2}\left(|c| - \frac{\lambda_1}{\hat{L}_i}\right)\text{sign}(c) & \text{if } \frac{\hat{L}_i}{\hat{L}_i+2\lambda_2}\left(|c| - \frac{\lambda_1}{\hat{L}_i}\right) \geq \sqrt{\frac{2\lambda_0}{\hat{L}_i+2\lambda_2}} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where $c = \beta_i - \nabla_i g(\boldsymbol{\beta})/\hat{L}_i$ and $\boldsymbol{\lambda} = (\lambda_0, \lambda_1, \lambda_2)$.

Algorithm 1 below, summarizes our cyclic CD algorithm.

Algorithm 1: Cyclic Coordinate Descent (CD)

- **Input:** Initialization $\boldsymbol{\beta}^0$ and constant $\hat{L}_i > L_i$ for every $i \in [p]$
- **Repeat for** $l = 0, 1, 2, \ldots$ **until convergence:**
  1. $i \leftarrow 1 + (l \mod p)$ and $\boldsymbol{\beta}_j^{l+1} \leftarrow \boldsymbol{\beta}_j^l$ for all $j \neq i$
  2. Update $\beta_i^{l+1} \leftarrow \operatorname{argmin}_{\beta_i} \widetilde{P}_{\hat{L}_i}(\beta_1^l, \ldots, \beta_i, \ldots, \beta_p^l; \boldsymbol{\beta}^l)$ using (10) with $c = \beta_i^l - \nabla_i g(\boldsymbol{\beta}^l)/\hat{L}_i$

Computational Guarantees. The convergence of cyclic CD has been extensively studied for certain classes of continuous objective functions, e.g., see Tseng (2001); Bertsekas (2016); Beck and Tetruashvili (2013) and the references therein. However, these results do not apply to our objective function due to the discontinuity in the ℓ_0-norm. In Theorem 1, we establish a new result which shows that cyclic CD (Algorithm 1) converges at an asymptotic linear rate, and we present a characterization of the corresponding solution. Theorem 1 is established under the following assumption:

Assumption 1 *Problem* (3) *satisfies at least one of the following conditions:*

1. *Strong convexity of the continuous regularizer, i.e.,* $\lambda_2 > 0$.
2. *Restricted Strong Convexity: For some* $u \in [p]$, *the function* $\boldsymbol{\beta}_S \mapsto g(\boldsymbol{\beta}_S)$ *is strongly convex for every* $S \subseteq [p]$ *such that* $|S| \leq u$. *Moreover,* λ_0 *and the initial solution* $\boldsymbol{\beta}^0$ *are chosen such that* $P(\boldsymbol{\beta}^0) < u\lambda_0$.

Theorem 1 *Let* $\{\boldsymbol{\beta}^l\}$ *be the sequence of iterates generated by Algorithm 1. Suppose that Assumption 1 holds, then:*

1. *The support of $\boldsymbol{\beta}^l$ stabilizes in a finite number of iterations, i.e., there exists an integer N and support $S \subset [p]$ such that $Supp(\boldsymbol{\beta}^l) = S$ for all $l \geq N$.*

2. *Let S be the support as defined in Part 1. The sequence $\{\boldsymbol{\beta}^l\}$ converges to a solution $\boldsymbol{\beta}^*$ with support S, satisfying:*

$$\begin{aligned} &\boldsymbol{\beta}^*_S \in \underset{\boldsymbol{\beta}_S}{\operatorname{argmin}}\, G(\boldsymbol{\beta}_S) \\ &|\beta^*_i| \geq \sqrt{\frac{2\lambda_0}{\hat{L}_i + 2\lambda_2}} \quad \textit{for } i \in S \\ \textit{and} \qquad &|\nabla_i g(\boldsymbol{\beta}^*)| - \lambda_1 \leq \sqrt{2\lambda_0(\hat{L}_i + 2\lambda_2)} \quad \textit{for } i \in S^c. \end{aligned} \tag{11}$$

3. *Let S be the support as defined in Part 1. Let us define $H(\boldsymbol{\beta}_S) \stackrel{\text{def}}{=} g(\boldsymbol{\beta}_S) + \lambda_2\|\boldsymbol{\beta}_S\|_2^2$. Let σ_S be the strong convexity parameter of $\boldsymbol{\beta}_S \mapsto H(\boldsymbol{\beta}_S)$, and let L_S be the Lipschitz constant of $\boldsymbol{\beta}_S \mapsto \nabla_S H(\boldsymbol{\beta}_S)$. Denote $\hat{L}_{\max} = \max_{i\in S} \hat{L}_i$ and $\hat{L}_{\min} = \min_{i\in S} \hat{L}_i$. Then, there exists an integer N' such that the following holds for all $t \geq N'$:*

$$P(\boldsymbol{\beta}^{(t+1)p}) - P(\boldsymbol{\beta}^*) \leq \left(1 - \frac{\sigma_S}{\gamma}\right)\left(P(\boldsymbol{\beta}^{tp}) - P(\boldsymbol{\beta}^*)\right), \tag{12}$$

where $\gamma^{-1} = 2\hat{L}_{\max}(1 + |S|L_S^2\hat{L}_{\min}^{-2})$.

We provide a proof of Theorem 1 in the appendix. The proof is different from that of CD for ℓ_0-regularized regression (Hazimeh and Mazumder, 2020) since we use inexact minimization for every coordinate update, whereas Hazimeh and Mazumder (2020) use exact minimization. At a high level, the proof proceeds as follows. In Part 1, we prove a sufficient decrease condition which establishes that the support stabilizes in a finite number of iterations. In Part 2, we show that under Assumption 1, the objective function is strongly convex when restricted to the stabilized support. After restriction to the stabilized support, we obtain convergence from standard results on cyclic CD (Bertsekas, 2016). In Part 3, we show an asymptotic linear rate of convergence for Algorithm 1. To establish this rate, we extend the linear rate of convergence of cyclic CD for smooth strongly convex functions by Beck and Tetruashvili (2013) to our objective function (note that due to the presence of the ℓ_1-norm, our objective is not smooth even after support stabilization).

Stationary Points of CD versus IHT: The conditions in (11) describe a fixed point of the cyclic CD algorithm and are necessary optimality conditions for Problem (3). We now show that the stationary conditions (11) are strictly contained within the class of stationary points arising from the IHT algorithm (Blumensath and Davies, 2009; Beck and Eldar, 2013; Bertsimas et al., 2016). Recall that IHT can be interpreted as a proximal gradient algorithm, whose updates for Problem (3) are given by:

$$\boldsymbol{\beta}^{l+1} \in \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \left\{\frac{1}{2\tau}\|\boldsymbol{\beta} - (\boldsymbol{\beta}^l - \tau\nabla g(\boldsymbol{\beta}^l))\|_2^2 + \psi(\boldsymbol{\beta})\right\}, \tag{13}$$

where $\tau > 0$ is a step size. Let L be the Lipschitz constant of $\boldsymbol{\beta} \mapsto \nabla g(\boldsymbol{\beta})$, and let $\hat{L}$ be any constant satisfying $\hat{L} > L$. Update (13) is guaranteed to converge to a stationary

point if $\tau = 1/\hat{L}$ (e.g., see Lu 2014; Hazimeh and Mazumder 2020). Note that $\tilde{\boldsymbol{\beta}}$ is a fixed point for (13) if it satisfies (11) with $\hat{L}_i$ replaced with $\hat{L}$. The component-wise Lipschitz constant L_i always satisfies $L_i \leq L$. For high-dimensional problems, we may have $L_i \ll L$ (see discussions in Beck and Eldar 2013; Hazimeh and Mazumder 2020 for problems where L grows with p but L_i is constant). Hence, the CD optimality conditions in (11) are more restrictive than IHT—justifying a part of the hierarchy mentioned in (4). An important practical consequence of this result is that CD may lead to solutions of higher quality than IHT.

Remark 2 *A solution $\boldsymbol{\beta}^*$ that satisfies the CD or IHT stationarity conditions is a local minimizer in the traditional sense used in nonlinear optimization.*[4] *Thus, we use the terms stationary point and local minimizer interchangeably in our exposition.*

2.2 Local Combinatorial Search

We propose a local combinatorial search algorithm to improve the quality of solutions obtained by Algorithm 1. Given a solution from Algorithm 1, the idea is to perform small perturbations to its support in an attempt to improve the objective. This approach has been recently shown to be very effective (e.g, in terms of statistical performance) for ℓ_0-regularized regression (Hazimeh and Mazumder, 2020), especially under difficult statistical settings (high feature correlations or n is small compared to p). Here, we extend the approach of Hazimeh and Mazumder (2020) to general loss functions, discussed in Section 2.1. As we consider a general loss function, performing exact local minimization becomes computationally expensive—we thus resort to an approximate minimization scheme. This makes our approach different from the least squares setting considered in Hazimeh and Mazumder (2020).

Our local search algorithm is iterative. It performs the following two steps at every iteration t:

1. **Coordinate Descent**: We run cyclic CD (Algorithm 1) initialized from the current solution, to obtain a solution $\boldsymbol{\beta}^t$ with support S.

2. **Combinatorial Search**: We attempt to improve $\boldsymbol{\beta}^t$ by making a change to its current support S via a *swap* operation. In particular, we search for two subsets of coordinates $S_1 \subset S$ and $S_2 \subset S^c$, each of size at most m, such that removing coordinates S_1 from the support, adding S_2 to the support, and then optimizing over the coefficients in S_2, improves the current objective value.

To present an optimization formulation for the combinatorial search step (discussed above), we introduce some notation. Let U^S denote a $p \times p$ matrix whose i-th row is $\boldsymbol{e}_i^T$ if $i \in S$ and zero otherwise. Thus, for any $\boldsymbol{\beta} \in \mathbb{R}^p$, $(U^S\boldsymbol{\beta})_i = \beta_i$ if $i \in S$ and $(U^S\boldsymbol{\beta})_i = 0$ if $i \notin S$. The combinatorial search step solves the following optimization problem:

$$\min_{S_1, S_2, \boldsymbol{\beta}} \quad P(\boldsymbol{\beta}^t - U^{S_1}\boldsymbol{\beta}^t + U^{S_2}\boldsymbol{\beta}) \qquad \text{s.t.} \qquad S_1 \subset S, S_2 \subset S^c, |S_1| \leq m, |S_2| \leq m, \tag{14}$$

4. That is, given a stationary point $\boldsymbol{\beta}^*$, there is a small $\epsilon > 0$ such that any $\boldsymbol{\beta}$ lying in the set $\|\boldsymbol{\beta} - \boldsymbol{\beta}^*\|_2 \leq \epsilon$ will have an objective that is at least as large as the current objective value $P(\boldsymbol{\beta}^*)$.

where the optimization variables are the subsets S_1, S_2 and the coefficients of $\boldsymbol{\beta}$ restricted to S_2. If there is a feasible solution $\hat{\boldsymbol{\beta}}$ to (14) satisfying $P(\hat{\boldsymbol{\beta}}) < P(\boldsymbol{\beta}^t)$, then we move to $\hat{\boldsymbol{\beta}}$. Otherwise, the current solution $\boldsymbol{\beta}^t$ cannot be improved by swapping subsets of coordinates, and the algorithm terminates. We summarize the algorithm below.

Algorithm 2: CD with Local Combinatorial Search

- **Input:** Initialization $\hat{\boldsymbol{\beta}}^0$ and swap subset size m.
- **Repeat for** $t = 1, 2, \ldots$**:**
  1. $\boldsymbol{\beta}^t \leftarrow$ Output of cyclic CD initialized from $\hat{\boldsymbol{\beta}}^{t-1}$. Let $S \leftarrow \text{Supp}(\boldsymbol{\beta}^t)$.
  2. Find a feasible solution $\hat{\boldsymbol{\beta}}$ to (14) satisfying $P(\hat{\boldsymbol{\beta}}) < P(\boldsymbol{\beta}^t)$.
  3. If Step 2 succeeds, then set $\hat{\boldsymbol{\beta}}^t \leftarrow \hat{\boldsymbol{\beta}}$. Otherwise, if Step 2 fails, **terminate**.

Theorem 3 shows that Algorithm 2 terminates in a finite number of iterations and provides a description of the resulting solution.

Theorem 3 *Let $\{\boldsymbol{\beta}^t\}$ be the sequence of iterates generated by Algorithm 2. Then, under Assumption 1, $\boldsymbol{\beta}^t$ converges in finitely many steps to a solution $\boldsymbol{\beta}^*$ (say). Let $S = Supp(\boldsymbol{\beta}^*)$. Then, $\boldsymbol{\beta}^*$ satisfies the stationary conditions in* (11) *(see Theorem 1). In addition, for every $S_1 \subset S$ and $S_2 \subset S^c$ with $|S_1| \leq m$, $|S_2| \leq m$, the solution $\boldsymbol{\beta}^*$ satisfies the following condition:*

$$P(\boldsymbol{\beta}^*) \leq \min_{\boldsymbol{\beta}} \quad P(\boldsymbol{\beta}^* - U^{S_1}\boldsymbol{\beta}^* + U^{S_2}\boldsymbol{\beta}). \tag{15}$$

Algorithm 2 improves the solutions obtained from Algorithm 1. This observation along with the discussion in Section 2.1, establishes the hierarchy of local minima in (4). The choice of m in Algorithm 2 controls the quality of the local minima returned—larger values of m will lead to solutions with better objectives. For a sufficiently large value of m, Algorithm 2 will deliver a global minimizer of Problem (3). The computation time of solving Problem (14) increases with m. We have observed empirically that small choices of m (e.g., $m = 1$) can lead to a global minimizer of Problem (3) even for some challenging high-dimensional problems where the features are highly correlated (see the experiments in Section 5).

Problem (14) can be formulated using MIP—this is discussed in Section 3, where we also present methods to solve it for large problems. The MIP-based framework allows us to (i) obtain good feasible solutions (if they are available) or (ii) certify (via dual bounds) that the current solution cannot be improved by swaps corresponding to size m. Note that due to the restricted search space, solving (14) for small values of m can be much easier than solving Problem (3) using MIP solvers. A solution $\boldsymbol{\beta}^*$ obtained from Algorithm 2 has an appealing interpretation: being a fixed point of (15), $\boldsymbol{\beta}^*$ cannot be improved by locally perturbing its support. This serves as a certificate describing the quality of the current (locally optimal) solution $\boldsymbol{\beta}^*$.

In what follows, we present a fast method to obtain a good solution to Problem (14) for the special case of $m = 1$.

Speeding up Combinatorial Search when $m = 1$: For Problem (14), we first check if removing variable i, without adding any new variables to the support, improves the

Algorithm 3: Fast Heuristic for Local Search when $m = 1$

- **Input**: Restricted set size $q \in \mathbb{Z}_+$ such that $q \leq p - |S|$.
- **For every** $i \in S$:
  1. If $P(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t) < P(\boldsymbol{\beta}^t)$ then **terminate** and return $\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t$.
  2. Compute $\nabla_{S^c} g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)$ and let J be the set of indices of the q components with the largest values of $|\nabla_j g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)|$ for $j \in S^c$.
  3. For every $j \in J$:
     Solve $\hat{\beta}_j \in \operatorname{argmin}_{\beta_j \in \mathbb{R}} G(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t + \boldsymbol{e}_j\beta_j)$ by iteratively applying the thresholding operator in (10) (with $\hat{L}_i = L_i$ and $\lambda_0 = 0$). If $P(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t + \boldsymbol{e}_j\hat{\beta}_j) < P(\boldsymbol{\beta}^t)$, **terminate** and return $\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t + \boldsymbol{e}_j\hat{\beta}_j$.

objective, i.e., $P(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t) < P(\boldsymbol{\beta}^t)$. If the latter inequality holds, we declare a success in Step 2 (Algorithm 2). Otherwise, we find a feasible solution to Problem (14) by solving

$$\min_{\beta_j} \quad G(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t + \boldsymbol{e}_j\beta_j) \tag{16}$$

for every pair $S_1 = \{i\}$ and $S_2 = \{j\}$. Performing the full minimization in (16) for every (i, j) can be expensive—so we propose an approximate scheme that is found to work relatively well in our numerical experience. We perform a *few* proximal gradient updates by applying the thresholding operator defined in (10) with the choice $\hat{L}_j = L_j$, $\lambda_0 = 0$, and using $\beta_j = 0$ as an initial value, to approximately minimize (16)—this helps us identify if the inclusion of coordinate j leads to a success in Step 2.

The method outlined above requires approximately solving Problem (16) for $|S|(p-|S|)$ many (i, j)-pairs (in the worst case). This cost can be further reduced if we select j from a small subset of coordinates outside the current support S, i.e., $j \in J \subset S^c$, where $|J| < p - |S|$. We choose J so that it corresponds to the q largest (absolute) values of the gradient $|\nabla_j g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)|$, $j \in S^c$. As explained in Section A.4, this choice of J ensures that we search among coordinates $j \in S^c$ that lead to the maximal decrease in the current objective with one step of a proximal coordinate update initialized from $\beta_j = 0$. We summarize the proposed method in Algorithm 3.

The cost of applying the thresholding operator in step 3 of Algorithm 3 is $\mathcal{O}(n)$.[5] For squared error loss, the cost can be improved to $\mathcal{O}(1)$ by reusing previously computed quantities from CD (see Hazimeh and Mazumder 2020 for details). Our numerical experience suggests that using the above heuristic with values of $q \approx 0.05 \times p$ often leads to the same solutions returned by full exhaustive search. Moreover, when some of the features are highly correlated, Algorithm 2 with the heuristic above (or solving Problem 14 exactly) performs better in terms of variable selection and prediction performance compared to state-of-the-art sparse learning algorithms (see Section 5.2 for numerical results).

5. Assuming that the derivative of $f(\cdot, v)$ w.r.t the first argument can be computed in $\mathcal{O}(1)$, which is the case for common loss functions that we consider here.

2.3 Solutions for the Cardinality Constrained Formulation

Algorithms 1 and 2 deliver good solutions for the ℓ_0-penalized problem (3). We now discuss how they can be used to obtain solutions to the cardinality constrained version:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} G(\boldsymbol{\beta}) \quad \text{s.t.} \quad \|\boldsymbol{\beta}\|_0 \leq k, \tag{17}$$

where k controls the support size of $\boldsymbol{\beta}$. While the unconstrained formulation (3) is amenable to fast CD-based algorithms, some support sizes are often skipped as λ_0 is varied. For example, if we decrease λ_0 to λ_0', the support size of the new solution can differ by more than one, even if λ_0' is taken to be arbitrarily close to λ_0. On the other hand, formulation (17) can typically return a solution with any desired support size,[6] and it may be preferable over the unconstrained formulation in some applications due to its explicit control of the support size.

Suppose we wish to obtain a solution to Problem (17) for a support size k that is not available from a sequence of solutions from (3). We propose to apply the IHT algorithm on Problem (17), initialized by a solution from Algorithm 1 or 2. This leads to the following update sequence

$$\beta^{l+1} \leftarrow \underset{\|\boldsymbol{\beta}\|_0\leq k}{\operatorname{argmin}} \left\{ \frac{1}{2\tau} \left\| \boldsymbol{\beta} - \left(\boldsymbol{\beta}^l - \tau\nabla g(\boldsymbol{\beta}^l)\right)\right\|_2^2 + \lambda_1\|\boldsymbol{\beta}\|_1 + \lambda_2\|\boldsymbol{\beta}\|_2^2 \right\}, \tag{18}$$

for $l \geq 0$, with initial solution $\boldsymbol{\beta}^0$ (available from the ℓ_0-penalized formulation) and $\tau > 0$ is a step size (e.g., see Theorem 4). We note that update (18) typically returns a solution of support size k but there can be degenerate cases where a support size of k is not possible (see footnote 6).

Beck and Eldar (2013) has shown that greedy CD-like algorithms perform better than IHT for a class of problems similar to (17) (without ℓ_1-regularization). However, it is computationally expensive to apply greedy CD methods to (17) for the problem sizes we study here. Note that since we initialize IHT with a solution from Problem (3), it converges rapidly to a high-quality feasible solution for Problem (17).

Theorem 4, which follows from Mazumder et al. (2017), establishes that IHT is guaranteed to converge, and provides a characterization of its fixed points.

Theorem 4 *Let $\{\boldsymbol{\beta}^l\}$ be a sequence generated by the IHT algorithm updates* (18)*. Let L be the Lipschitz constant of $\nabla g(\boldsymbol{\beta})$ and let $\hat{L} > L$. Then, the sequence $\{\boldsymbol{\beta}^l\}$ converges for a step size $\tau = 1/\hat{L}$. Moreover, $\boldsymbol{\beta}^*$ with support S is a fixed point of* (18) *iff $\|\boldsymbol{\beta}^*\|_0 \leq k$,*

$$\boldsymbol{\beta}^*_S \in \underset{\boldsymbol{\beta}_S}{\operatorname{argmin}}\ G(\boldsymbol{\beta}_S) \quad \text{and} \quad |\nabla_i g(\boldsymbol{\beta}^*)| \leq \delta_{(k)} \quad \text{for } i \in S^c,$$

where $\delta_j = |\hat{L}\beta^*_j - \nabla_j g(\boldsymbol{\beta}^*)|$ for $j \in [p]$ and $\delta_{(k)}$ is the kth largest value of $\{\delta_j\}_1^p$.

Lemma 5 shows that if a solution obtained by Algorithm 1 or 2 has a support size k, then it is a fixed point for the IHT update (18).

6. Exceptions can happen if for a subset $T \subset [p]$ of size k, the minimum of $\boldsymbol{\beta}_T \mapsto G(\boldsymbol{\beta}_T)$ has some coordinates in $\boldsymbol{\beta}_T$ exactly set to zero.

Lemma 5 *Let $\gamma > 1$ be a constant and $\boldsymbol{\beta}^*$ with support size k be a solution for Problem* (3) *obtained by using Algorithm 1 or 2 with $\hat{L}_i = \gamma L_i$. Set $\hat{L} = \gamma L$ and $\tau = 1/\hat{L}$ in* (18)*. Then, $\boldsymbol{\beta}^*$ is a fixed point of the update* (18)*.*

The converse of Lemma 5 is not true, i.e., a fixed point of update (18) may not be a fixed point for Algorithm 1 or 2—see our earlier discussion around hierarchy (4).[7] Thus, we can generally expect the solutions returned by Algorithm 1 or 2 to be of higher quality than those returned by IHT.

The following summarizes our procedure to obtain a path of solutions for Problem (17) (here, we assume that (λ_1, λ_2) are fixed and λ_0 varies):

1. Run Algorithm 1 or 2 for a sequence of λ_0 values to obtain a regularization path.
2. To obtain a solution to Problem (17) with a support size (say k) that is not available in Step 1, we run the IHT updates (18). The IHT updates are initialized with a solution from Step 1 having a support size smaller than k.

As the above procedure uses high-quality solutions obtained by Algorithm 1 or 2 as advanced initializations for IHT, we expect to obtain notable performance benefits as compared to using IHT alone for generating the regularization path. Also, note that if a support size k is available from Step 1, then there is no need to run IHT (see Lemma 5).

2.4 L0Learn: A Fast Toolkit for ℓ_0-regularized Learning

We implemented the algorithms discussed above in `L0Learn`: a fast sparse learning toolkit written in C++ along with an R interface. We currently support the logistic and squared-hinge loss functions,[8] but the toolkit can be expanded and we intend to incorporate additional loss functions in the future. Following Friedman et al. (2010); Hazimeh and Mazumder (2020), we used several computational tricks to speed up the algorithms and improve the solution quality—these include: warm starts, active sets, correlation screening, a (partially) greedy heuristic to cycle through the coordinates, and efficient methods for updating the gradients by exploiting sparsity. Note that Problem (3) can lead to the same solution if two values of λ_0 are close to one another. To avoid this issue, we dynamically select a sequence of λ_0-values—this is an extension of an idea appearing in Hazimeh and Mazumder (2020) for the least squares loss function.

3. Mixed Integer Programming Algorithms

We present MIP formulations and a new scalable algorithm: IGA, for solving Problem (3) to optimality. Compared to off-the-shelf MIP solvers (e.g., the commercial solver Gurobi), IGA leads to certifiably optimal solutions in significantly reduced computation times. IGA also applies to the local search problem of Section 2.2. We remind the reader that while Algorithm 1 (and Algorithm 2 for small values of m) leads to good feasible solutions quite fast, it does not deliver certificates of optimality (via dual bounds). The MIP framework can be used to certify the quality of solutions obtained by Algorithms 1 or 2 and potentially improve upon them.

7. Assuming that we obtain the same support sizes for both the constrained and unconstrained formulations.
8. This builds upon our earlier functionality for least squares loss (Hazimeh and Mazumder, 2020).

3.1 MIP Formulations

Problem (3) admits the following MIP formulation:

$$\min_{\boldsymbol{\beta},\mathbf{z}} \quad \left\{ G(\boldsymbol{\beta}) + \lambda_0 \sum_{i=1}^{p} z_i \right\} \quad \text{s.t.} \qquad |\beta_i| \leq \mathcal{M} z_i,\ z_i \in \{0,1\},\ i \in [p] \tag{19}$$

where $\mathcal{M}$ is a constant chosen large enough so that some optimal solution $\boldsymbol{\beta}^*$ to (2) satisfies $\|\boldsymbol{\beta}^*\|_\infty \leq \mathcal{M}$. Algorithm 1 and Algorithm 2 (with $m = 1$) can be used to obtain good estimates of $\mathcal{M}$—see Bertsimas et al. (2016); Mazumder et al. (2017) for possible alternatives on choosing $\mathcal{M}$. In (19), the binary variable z_i controls whether β_i is set to zero or not. If $z_i = 0$ then $\beta_i = 0$, while if $z_i = 1$ then $\beta_i \in [-\mathcal{M}, \mathcal{M}]$ is allowed to be 'free'. For the hinge loss with $q = 1$, the problem is a Mixed Integer Linear Program (MILP). For $q = 2$ and the hinge loss function, (19) becomes a Mixed Integer Quadratic Program (MIQP). Similarly, the squared hinge loss leads to a MIQP (for both $q \in \{1, 2\}$). MILPs and MIQPs can be solved by state-of-the-art MIP solvers (e.g., Gurobi and CPLEX) for small-to-moderate sized instances. For other nonlinear convex loss functions such as logistic loss, two approaches are possible: (i) nonlinear branch and bound (Belotti et al., 2013) or (ii) outer-approximation in which a sequence of MILPs is solved until convergence (for example, see Bertsimas et al. 2017, 2020; Sato et al. 2016).[9] We refer the reader to Lee and Leyffer (2011) for a review of related approaches.

The local combinatorial search problem in (14) can be cast as a variant of (19) with additional constraints; and is given by the following MIP:

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{z},\boldsymbol{\theta}} \quad & G(\boldsymbol{\theta}) + \lambda_0 \sum_{i=1}^{p} z_i && \text{(20a)}\\
\text{s.t.} \quad & \boldsymbol{\theta} = \boldsymbol{\beta}^t - \sum_{i \in S} \mathbf{e}_i \beta_i^t (1 - z_i) + \sum_{i \in S^c} \mathbf{e}_i \beta_i && \text{(20b)}\\
& |\beta_i| \leq \mathcal{M} z_i, \quad i \in S^c && \text{(20c)}\\
& \sum_{i \in S} z_i \geq |S| - m && \text{(20d)}\\
& \sum_{i \in S^c} z_i \leq m && \text{(20e)}\\
& z_i \in \{0,1\}, \quad i \in [p] && \text{(20f)}
\end{aligned}$$

where $S = \text{Supp}(\boldsymbol{\beta}^t)$. The binary variables $z_i, i \in [p]$ perform the role of *selecting* the subsets $S_1 \subset S$ and $S_2 \subset S^c$ (described in (14)). Particularly, for $i \in S$, $z_i = 0$ means that $i \in S_1$—i.e., variable i should be removed from the current support. Similarly, for $i \in S^c$, $z_i = 1$ means that $i \in S_2$—i.e., variable j should be added to the support in (20). Constraints (20d) and (20e) enforce $|S_1| \leq m$ and $|S_2| \leq m$, respectively. The constraint in (20b) forces the variable $\boldsymbol{\theta}$ to be equal to $\boldsymbol{\beta}^t - U^{S_1}\boldsymbol{\beta}^t + U^{S_2}\boldsymbol{\beta}$.

9. In this method, one obtains a convex piece-wise linear lower bound to a nonlinear convex problem. In other words, this leads to a polyhedral outer approximation (aka outer approximation) to the epigraph of the nonlinear convex function.

Note that (20) has a smaller search space compared to the full formulation in (19)—there are additional constraints in (20d) and (20e), and the number of free continuous variables is $|S^c|$, as opposed to p in (19). This reduced search space usually leads to notable reductions in the runtime compared to solving Problem (19).

3.2 Scaling up the MIP via the Integrality Generation Algorithm (IGA)

While state-of-the-art MIP solvers and outer-approximation based MILP approaches (Bertsimas et al., 2017; Sato et al., 2016) lead to impressive improvements over earlier MIP-based approaches, they often have long run times when solving high-dimensional classification problems with large p and small n (e.g., $p = 50,000$ and $n = 1000$). Our proposed algorithm, IGA, can solve (19) to *global* optimality, for high-dimensional instances that appear to be beyond the capabilities of current MIP-based approaches. Loosely speaking, IGA solves a sequence of MIP-based relaxations or subproblems of Problem (19); and exits upon obtaining a global optimality certificate for (19). The aforementioned MIP subproblems are obtained by relaxing a subset of the binary variables $\{z_i\}_1^p$ to lie within $[0, 1]$, while the remaining variables are retained as binary. Upon solving the relaxed problem and examining the integrality of the continuous z_i's, we create another (tighter) relaxation by allowing more variables to be binary—we continue in this fashion till convergence. The algorithm is formally described below.

The first step in our algorithm is to obtain a good upper bound for Problem (19)—this can be obtained by Algorithm 1 or 2. Let $\mathcal{I}$ denote the corresponding support of this solution. We then consider a relaxation of Problem (19) by allowing the binary variables in $\mathcal{I}^c$ to be continuous:

$$\begin{aligned}
\min_{\boldsymbol{\beta},\mathbf{z}} \quad & G(\boldsymbol{\beta}) + \lambda_0 \sum_{i=1}^{p} z_i && \text{(21a)}\\
\text{s.t.} \quad & |\beta_i| \leq \mathcal{M} z_i, \quad i \in [p] && \text{(21b)}\\
& z_i \in [0, 1], \quad i \in \mathcal{I}^c && \text{(21c)}\\
& z_i \in \{0, 1\}, \quad i \in \mathcal{I}. && \text{(21d)}
\end{aligned}$$

The relaxation (21) is a MIP (and thus nonconvex). The optimal objective of Problem (21) is a lower bound to Problem (19). In formulation (21), we place integrality constraints on $z_i, i \in \mathcal{I}$—all remaining variables z_i, $i \in \mathcal{I}^c$ are continuous. Let $\boldsymbol{\beta}^u, \mathbf{z}^u$ be the solution obtained from the u-th iteration of the algorithm. Then, in iteration $(u + 1)$, we set $\mathcal{I} \leftarrow \mathcal{I} \cup \{i \mid z_i^u \neq 0, i \in \mathcal{I}^c\}$ and solve Problem (21) (with warm-starting enabled). If at some iteration u, the vector $\mathbf{z}^u$ is integral, then solution $\boldsymbol{\beta}^u, \mathbf{z}^u$ must be optimal for Problem (19) and the algorithm terminates. We note that along the iterations, we obtain tighter lower bounds on the optimal objective of Problem (19). Depending on the available computational budget, we can decide to terminate the algorithm at an early stage with a corresponding lower bound. The algorithm is summarized below:

Algorithm 4: Integrality Generation Algorithm (IGA)

- Initialize $\mathcal{I}$ to the support of a solution obtained by Algorithm 1 or 2.
- **For** $u = 1, 2, \ldots$ perform the following steps till convergence:
  1. Solve the relaxed MIP (21) to obtain a solution $\boldsymbol{\beta}^u, \mathbf{z}^u$.
  2. Update $\mathcal{I} \leftarrow \mathcal{I} \cup \{i \mid z_i^u \neq 0, i \in \mathcal{I}^c\}$.

As we demonstrate in Section 5, Algorithm 4 can lead to significant speed-ups: it reduces the time to solve several sparse classification instances from the order of *hours* to *seconds*. This allows us to solve instances with $p \approx 50,000$ and $n \approx 1000$ within reasonable computation times. These instances are much larger than what has been reported in the literature prior to this work (see for example, Sato et al. 2016; Bertsimas et al. 2017). A main reason behind the success of Algorithm 4 is that Problem (21) leads to a solution $\mathbf{z}$ with very few nonzero coordinates. Hence, a small number of indices are added to $\mathcal{I}$ in step 2 of the algorithm. Since $\mathcal{I}$ is typically small, (21) can be often solved significantly faster than the full MIP in (19)—the branch-and-bound algorithm has a smaller number of variables to branch on. Lemma 6 provides some intuition on why the solutions of (21) are sparse.

Lemma 6 *Problem* (21) *can be equivalently written as:*

$$\min_{\boldsymbol{\beta}, \mathbf{z}_{\mathcal{I}}} \quad G(\boldsymbol{\beta}) + \frac{\lambda_0}{\mathcal{M}} \sum_{i \in \mathcal{I}^c} |\beta_i| + \lambda_0 \sum_{i \in \mathcal{I}} z_i \tag{22a}$$

$$s.t. \quad |\beta_i| \leq \mathcal{M} z_i, \quad i \in \mathcal{I} \tag{22b}$$

$$|\beta_i| \leq \mathcal{M}, \quad i \in \mathcal{I}^c \tag{22c}$$

$$z_i \in \{0, 1\}, \quad i \in \mathcal{I}. \tag{22d}$$

We have observed empirically that in (22), the ℓ_1-regularization term $\sum_{i \in \mathcal{I}^c} |\beta_i|$ encourages sparse solutions, i.e., many of the components $\beta_i, i \in \mathcal{I}^c$ are set to zero. Consequently, the corresponding z_i's in (21) are mostly zero at an optimal solution. The sparsity level is controlled by the regularization parameter $\lambda_0/\mathcal{M}$—larger values will lead to more z_i's being set to zero in (21). Thus, we expect Algorithm 4 to work well when λ_0 is set to a sufficiently large value (to obtain sufficiently sparse solutions). Note that Problem (22) is different from the Lasso as it involves additional integrality constraints.

Optimality Gap and Early Termination: Each iteration of Algorithm 4 provides an improved lower bound to Problem (19). This lower bound, along with a good feasible solution (e.g., obtained from Algorithm 1 or 2), leads to an optimality gap: given an upper bound UB and a lower bound LB, the MIP optimality gap is defined by (UB - LB)/LB. This optimality gap serves as a certificate of global optimality. In particular, an early termination of Algorithm 4 leads to a solution with an associated certificate of optimality.

Choice of $\mathcal{I}$: The performance of Algorithm 4 depends on the initial set $\mathcal{I}$. If the initial $\mathcal{I}$ is close to the support of an optimal solution to (19), then our numerical experience suggests that Algorithm 4 can terminate within a few iterations. Moreover, our experiments

(see Section 5.2) suggest that Algorithm 2 can obtain an optimal or a near-optimal solution to Problem (19) quickly, leading to a high-quality initialization for $\mathcal{I}$.

In practice, if at iteration u of Algorithm 4, the set $\{i \mid z_i^u \neq 0\}$ is large, then we add only a small subset of it to $\mathcal{I}$ (in our implementation, we choose the 10 largest fractional z_i's). Alternatively, while expanding $\mathcal{I}$, we can use a larger cutoff for the fractional z_i's, i.e., we can take the indices $\{i \mid z_i^u \geq \tau\}$ for some value of $\tau \in (0, 1)$. This usually helps in maintaining a small size in $\mathcal{I}$, which allows for solving the MIP subproblem (21) relatively quickly.

IGA for Local Combinatorial Search: While the discussion above was centered around the full MIP (19)—the IGA framework (and in particular, Algorithm 4) extends, in principle, to the local combinatorial search problem in (20) for $m \geq 2$.

4. Statistical Properties: Error Bounds

We derive non-asymptotic upper bounds on the coefficient estimation error for a family of ℓ_0-constrained classification estimators (this includes the loss functions discussed in Section 1.2, among others). For our analysis, we assume that: $(\mathbf{x}_i, y_i), i \in [n]$ are i.i.d. draws from an unknown distribution $\mathbb{P}$. Using the notation of Section 2, we consider a loss function f and define its population risk $\mathcal{L}(\boldsymbol{\beta}) = \mathbb{E}\left(f\left(\langle \mathbf{x}, \boldsymbol{\beta}\rangle; y\right)\right)$, where the expectation is w.r.t. the (population) distribution $\mathbb{P}$. We let $\boldsymbol{\beta}^*$ denote a minimizer of the risk, that is:

$$\boldsymbol{\beta}^* \in \operatorname*{argmin}_{\boldsymbol{\beta} \in \mathbb{R}^p} \; \mathcal{L}(\boldsymbol{\beta}) := \mathbb{E}\left(f\left(\langle \mathbf{x}, \boldsymbol{\beta}\rangle; y\right)\right). \tag{23}$$

In the rest of this section, we let $k = \|\boldsymbol{\beta}^*\|_0$ and $R = \|\boldsymbol{\beta}^*\|_2$—i.e., the number of nonzeros and the Euclidean norm (respectively) of $\boldsymbol{\beta}^*$. We assume $R \geq 1$. To estimate $\boldsymbol{\beta}^*$, we consider the following estimator:[10]

$$\hat{\boldsymbol{\beta}} \;\; \in \;\; \operatorname*{argmin}_{\substack{\boldsymbol{\beta} \in \mathbb{R}^p \\ \|\boldsymbol{\beta}\|_0 \leq k, \; \|\boldsymbol{\beta}\|_2 \leq 2R}} \;\; \frac{1}{n} \sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}\rangle; y_i\right), \tag{24}$$

which minimizes the empirical loss with a constraint on the number of nonzeros in $\boldsymbol{\beta}$ and a bound on the ℓ_2-norm of $\boldsymbol{\beta}$. The ℓ_2-norm constraint in (24) makes $\boldsymbol{\beta}^*$ feasible for Problem (24); and ensures that $\hat{\boldsymbol{\beta}}$ lies in a bounded set (which is useful for the technical analysis).

Section 4.1 presents the assumptions we need for our analysis—see the works of Peng et al. (2016), Ravikumar et al. (2010) and Belloni and Chernozhukov (2011) for related assumptions in the context of ℓ_1-based classification and quantile regression procedures. In Section 4.2, we establish a high probability upper bound on $\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2$.

4.1 Assumptions

We first present some assumptions for establishing the error bounds.

Loss Function. We list our basic assumptions on the loss function.

Assumption 2 *The function $t \mapsto f(t; y)$ is non-negative, convex, and Lipschitz continuous with constant L, that is $|f(t_1; y) - f(t_2; y)| \leq L|t_1 - t_2|, \; \forall t_1, t_2$.*

10. We drop the dependence of $\hat{\boldsymbol{\beta}}$ on R, k for notational convenience.

We let $\partial f(t;y)$ denote a subgradient of $t \mapsto f(t;y)$—i.e., $f(t_2;y) - f(t_1;y) \geq \partial f(t_1;y)(t_2 - t_1),\ \forall t_1, t_2$. Note that the hinge loss satisfies Assumption 2 with $L = 1$; and has a subgradient given by $\partial f(t;y) = \mathbf{1}\,(1 - yt \geq 0)\, y$. The logistic loss function satisfies Assumption 2 with $L = 1$; and its subgradient coincides with the gradient.

Differentiability of the Population Risk. The following assumption is on the uniqueness of $\boldsymbol{\beta}^*$ and differentiability of the population risk $\mathcal{L}$.

Assumption 3 *Problem* (23) *has a unique minimizer. The population risk $\boldsymbol{\beta} \mapsto \mathcal{L}(\boldsymbol{\beta})$ is twice continuously differentiable, with gradient $\nabla\mathcal{L}(\boldsymbol{\beta})$ and Hessian $\nabla^2\mathcal{L}(\boldsymbol{\beta})$. In particular, the following holds:*

$$\nabla\mathcal{L}(\boldsymbol{\beta}) = \mathbb{E}\left(\partial f\left(\langle \mathbf{x}, \boldsymbol{\beta}\rangle; y\right)\mathbf{x}\right). \tag{25}$$

When f is the hinge loss, Koo et al. (2008) discuss conditions under which Assumption 3 holds. In particular, Assumption (A1) in Koo et al. (2008) requires that the conditional density functions of $\mathbf{x}$ for the two classes are continuous and have finite second moments. Under this assumption, the Hessian $\nabla^2\mathcal{L}(\boldsymbol{\beta})$ is well defined and continuous in $\boldsymbol{\beta}$. Under Assumption (A4) (Koo et al., 2008), the Hessian is positive definite at an optimal solution, therefore (23) has a unique solution. We refer the reader to Ravikumar et al. (2010); Van de Geer (2008) for discussions pertaining to logistic regression.

Restricted Eigenvalue Conditions. Assumption 4 is a *restricted eigenvalue condition* similar to that used in regression problems (Bickel et al., 2009; Bühlmann and Van De Geer, 2011). For an integer $\ell > 0$, we assume that the quadratic forms associated with the Hessian matrix $\nabla^2\mathcal{L}(\boldsymbol{\beta}^*)$ and the covariance matrix $n^{-1}\mathbf{X}^T\mathbf{X}$ are respectively lower-bounded and upper-bounded on the set of 2ℓ-sparse vectors.

Assumption 4 *Let $\ell > 0$ be an integer. Assumption 4(ℓ) is said to hold if there exists constants $\kappa(\ell), \lambda(\ell) > 0$ such that almost surely the following holds:*

$$\kappa(\ell) \leq \inf_{\mathbf{z}\neq 0, \|\mathbf{z}\|_0 \leq 2\ell} \left\{ \frac{\mathbf{z}^T\nabla^2\mathcal{L}(\boldsymbol{\beta}^*)\mathbf{z}}{\|\mathbf{z}\|_2^2} \right\} \qquad \text{and} \qquad \lambda(\ell) \geq \sup_{\mathbf{z}\neq 0, \|\mathbf{z}\|_0 \leq 2\ell} \left\{ \frac{\|\mathbf{X}\mathbf{z}\|_2^2}{n\|\mathbf{z}\|_2^2} \right\}.$$

In the rest of this section, we consider Assumption 4 with $\ell = k$. Assumption (A4) in Peng et al. (2016) for linear SVM is similar to our Assumption 4. For logistic regression, related assumptions appear in the literature, e.g., Assumptions A1 and A2 in Ravikumar et al. (2010) (in the form of a dependency and an incoherence condition on the population Fisher information matrix).

Growth condition. As $\boldsymbol{\beta}^*$ minimizes the population risk, we have $\nabla\mathcal{L}(\boldsymbol{\beta}^*) = 0$. Under the above regularity assumptions and when Assumption 4(k) is satisfied, the population risk is lower-bounded by a quadratic function in a neighborhood of $\boldsymbol{\beta}^*$. By continuity, we let $r(k)$ denote the maximal radius for which the following lower bound holds:

$$r(k) = \max\left\{ r > 0 \;\middle|\; \mathcal{L}(\boldsymbol{\beta}^* + \mathbf{z}) \geq \mathcal{L}(\boldsymbol{\beta}^*) + \frac{1}{4}\kappa(k)\|\mathbf{z}\|_2^2 \quad \forall \mathbf{z} \quad \text{s.t.} \quad \|\mathbf{z}\|_0 \leq 2k, \|\mathbf{z}\|_2 \leq r \right\}. \tag{26}$$

Below we make an assumption on the growth condition.

Assumption 5 *Let $\delta \in (0, 1/2)$. We say that Assumption 5(δ) holds if the parameters n, p, k, R satisfy:*

$$\frac{24L}{\kappa(k)}\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right)+\log\left(1/\delta\right)\right)} < r(k),$$

and the following holds: $(k/n)\log(p/k) \leq 1$ and $7ne \leq 3L\sqrt{\lambda(k)}p\log\left(p/k\right)$.

Assumption 5 is similar to the scaling conditions in Ravikumar et al. (2010)[Theorem 1] for logistic regression. Belloni and Chernozhukov (2011) also makes use of a growth condition for their analysis in ℓ_1-sparse quantile regression problems. Note that although Assumption 5 is not required to prove Theorem 7, we will need it to derive the error bound of Theorem 8. We now proceed to derive an upper bound on coefficient estimation error.

4.2 Main Result

We first show (in Theorem 7) that the loss function f satisfies a form of restricted strong convexity (Negahban et al., 2009) around $\boldsymbol{\beta}^*$, a minimizer of (23).

Theorem 7 *Let $\boldsymbol{h} = \hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*$, $\delta \in (0, 1/2)$, and $\tau = 6L\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right)+\log\left(1/\delta\right)\right)}$. If Assumptions 2, 3, 4(k) and 5(δ) are satisfied, then with probability at least $1-\delta$, the following holds:*

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}^* + \mathbf{h}\rangle; y_i\right) - \frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}^*\rangle; y_i\right) \\ \geq \frac{1}{4}\kappa(k)\left\{\|\mathbf{h}\|_2^2 \wedge r(k)\|\mathbf{h}\|_2\right\} - \tau\|\mathbf{h}\|_2 \vee \tau^2.\end{aligned} \tag{27}$$

Theorem 7 is used to derive the error bound presented in Theorem 8, which is the main result in this section.

Theorem 8 *Let $\delta \in (0, 1/2)$ and assume that Assumptions 2, 3, 4(k) and 5(δ) hold. Then the estimator $\hat{\boldsymbol{\beta}}$ defined as a solution of Problem (24) satisfies with probability at least $1-\delta$:*

$$\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2 \lesssim L^2\lambda(k)\widetilde{\kappa}^2\left(\frac{k\log(Rp/k)}{n} + \frac{\log(1/\delta)}{n}\right) \tag{28}$$

where, $\widetilde{\kappa} = \max\{1/\kappa(k), 1\}$.

In (28), the symbol "$\lesssim$" stands for "$\leq$" up to a universal constant. The proof of Theorem 8 is presented in Appendix A.7. The rate appearing in Theorem 8, of the order of $k/n\log(p/k)$, is the best known rate for a (sparse) classifier; and this coincides with the optimal scaling in the case of regression. In comparison, Peng et al. (2016) and Ravikumar et al. (2010) derived a bound scaling as $k/n\log(p)$ for ℓ_1-regularized SVM (with hinge loss) and logistic regression, respectively. Note that the bound in Theorem 8 holds for any sufficiently small $\delta > 0$. Consequently, by integration, we obtain the following result in expectation.

Corollary 1 *Suppose Assumptions 2, 3, 4(k) hold true and Assumption 5(δ) is true for δ small enough, then:*

$$\mathbb{E}\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2 \lesssim L^2\lambda(k)\widetilde{\kappa}^2\frac{k\log(Rp/k)}{n},$$

where $\widetilde{\kappa}$ is defined in Theorem 8.

5. Experiments

In this section, we compare the statistical and computational performance of our proposed algorithms versus the state of the art, on both synthetic and real data sets.

5.1 Experimental Setup

Data Generation. For the synthetic data sets, we generate a multivariate Gaussian data matrix $\mathbf{X}_{n\times p} \sim \text{MVN}(\mathbf{0}, \mathbf{\Sigma})$ and a sparse vector of coefficients $\boldsymbol{\beta}^\dagger$ with $k^\dagger$ nonzero entries, such that $\beta_i^\dagger = 1$ for $k^\dagger$ equi-spaced indices $i \in [p]$. Every coordinate y_i of the outcome vector $\mathbf{y} \in \{-1, 1\}^n$ is then sampled independently from a Bernoulli distribution with success probability: $P(y_i = 1|\mathbf{x}_i) = (1 + \exp(-s\langle\boldsymbol{\beta}^\dagger, \mathbf{x}_i\rangle))^{-1}$, where $\mathbf{x}_i$ denotes the i-th row of $\mathbf{X}$, and s is a parameter that controls the signal-to-noise ratio. Specifically, smaller values of s increase the variance in the response y, and when $s \to \infty$ the generated data becomes linearly separable.

Algorithms and Tuning. We compare our proposal, as implemented in our package `L0Learn`, with: (i) ℓ_1-regularized logistic regression (`glmnet` package Friedman et al., 2010), (ii) MCP-regularized logistic regression (`ncvreg` package Breheny and Huang, 2011), and (iii) two packages for sparsity constrained minimization (based on hard thresholding): `GraSP` (Bahmani et al., 2013) and `NHTP` (Zhou et al., 2021). For `GraSP` and `NHTP`, we use cardinality constrained logistic regression with ridge regularization—that is, we optimize Problem (17) with $\lambda_1 = 0$. Tuning is done on a separate validation set under the fixed design setting, i.e., we use the same features used for training but a new outcome vector $\mathbf{y}'$ (independent of $\mathbf{y}$). The tuning parameters are selected so as to minimize the loss function on the validation set (e.g., for regularized logistic regression, we minimize the unregularized negative log-likelihood).

We use ℓ_0-ℓ_q as a shorthand to denote the penalty $\lambda_0\|\boldsymbol{\beta}\|_0 + \lambda_q\|\boldsymbol{\beta}\|_q^q$ for $q \in \{1, 2\}$. For all penalties that involve 2 tuning parameters—i.e., ℓ_0-ℓ_q (for $q \in \{1, 2\}$), MCP, `GraSP`, and `NHTP`, we sweep the parameters over a two-dimensional grid. For our penalties, we choose 100 λ_0 values as described in Section 2.4. For `GraSP` and `NHTP`, we sweep the number of nonzeros between 1 and 100. For ℓ_0-ℓ_1, and we choose a sequence of 10 λ_1-values in $[a, b]$, where a corresponds to a zero solution and $b = 10^{-4}a$. Similarly, for ℓ_0-ℓ_2, `GraSP`, and `NHTP`, we choose 10 λ_2 values between 10^{-4} and 100 for the experiment in Section 5.2; and between 10^{-8} and 10^{-4} for that in Section 5.3. For MCP, the sequence of 100 λ values is set to the default values selected by `ncvreg`, and we vary the second parameter γ over 10 values between 1.5 and 25. For the ℓ_1-penalty, the grid of 100 λ values is set to the default sequence chosen by `glmnet`.

Performance Measures. We use the following measures to evaluate the performance of an estimator $\hat{\boldsymbol{\beta}}$:

- **AUC**: The area under the curve of the ROC plot.
- **Recovery F1 Score**: This is the F1 score for support recovery, i.e., it is the harmonic mean of precision and recall: F1 Score $= 2PR/(P + R)$, where P is the precision given by $|\text{Supp}(\hat{\boldsymbol{\beta}}) \cap \text{Supp}(\boldsymbol{\beta}^\dagger)|/|\text{Supp}(\hat{\boldsymbol{\beta}})|$, and R is the recall given by $|\text{Supp}(\hat{\boldsymbol{\beta}}) \cap \text{Supp}(\boldsymbol{\beta}^\dagger)|/|\text{Supp}(\boldsymbol{\beta}^\dagger)|$. An F1 Score of 1 implies full support recovery; and a value of zero implies that the supports of the true and estimated coefficients have no overlap.

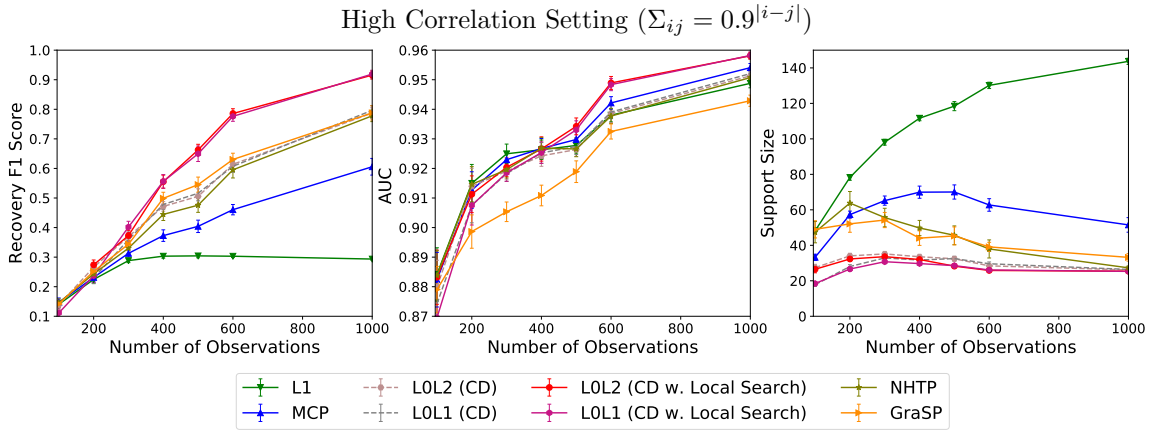

Figure 1: Performance for varying $n \in [100, 10^3]$ and $\Sigma_{ij} = 0.9^{|i-j|}, p = 1000, k^\dagger = 25, s = 1$. In this high-correlation setting, our proposed algorithm (using local search) for the ℓ_0-ℓ_q (for both $q \in \{1, 2\}$) penalized estimators—denoted as L0L2 (CD w. Local Search) and L0L1 (CD w. Local Search) in the figure—seems to outperform state-of-the-art methods (MCP, ℓ_1, `GraSP` and `NHTP`) in terms of both variable selection and prediction. The variable selection performance improvement (higher F1 score and smaller support size) is more notable than AUC. Local search with CD shows benefits compared to its variants that do not employ local search, the latter denoted by L0L1 (CD) and L0L2 (CD) in the figure.

- **Support Size**: The number of nonzeros in $\hat{\boldsymbol{\beta}}$.
- **False Positives**: This is equal to $|\text{Supp}(\hat{\boldsymbol{\beta}}) \setminus \text{Supp}(\boldsymbol{\beta}^\dagger)|$.

5.2 Performance for Varying Sample Sizes

In this experiment, we fix p and vary the number of observations n to study its effect on the performance of the different algorithms and penalties. We hypothesize that when the statistical setting is difficult (e.g., features are highly correlated and/or n is small), good optimization algorithms for ℓ_0-regularized problems lead to estimators that can significantly outperform estimators obtained from convex regularizers and common (heuristic) algorithms for nonconvex regularizers. To demonstrate our hypothesis, we perform experiments on the following data sets:

- **High Correlation**: $\Sigma_{ij} = 0.9^{|i-j|}, p = 1000, k^\dagger = 25, s = 1$
- **Medium Correlation**: $\Sigma_{ij} = 0.5^{|i-j|}, p = 1000, k^\dagger = 25, s = 1$

For each of the above settings, we use the logistic loss function and consider 20 random repetitions. We report the averaged results for the high correlation setting in Figure 1 and for the medium correlation setting in Figure 2.

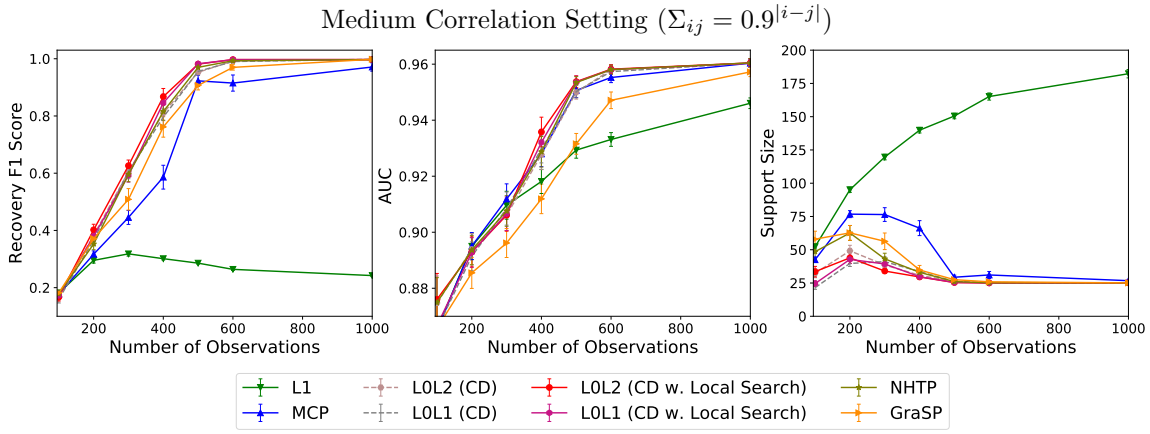

Figure 2: Performance for varying n and $\Sigma_{ij} = 0.5^{|i-j|}, p = 1000, k^{\dagger} = 25, s = 1$. In contrast to Figure 1, this is a medium-correlation setting. Once again Algorithm 2 (CD with local search) for the ℓ_0-ℓ_1 and ℓ_0-ℓ_2 penalties seems to perform quite well in terms of variable selection and prediction performance, though its improvement over the other methods appears to be less prominent compared to the high-correlation setting in Figure 1. In this example, local search does not seem to offer much improvement over pure CD (i.e., Algorithm 1).

Figure 1 shows that in the high correlation setting, Algorithm 2 (with $m = 1$) for the ℓ_0-ℓ_2 penalty and the ℓ_0-ℓ_1 penalty—denoted by L0L2 (CD w. Local Search) and L0L1 (CD w. Local Search) (respectively) in the figure—achieves the best support recovery and AUC across different sample sizes n. We note that in this example, the best F1 score falls below 0.8 for n smaller than 600—suggesting that none of the algorithms can do full support recovery. For larger values of n, the difference between Algorithm 2 and others become more pronounced in terms of F1 score, suggesting an important edge in terms of variable selection performance. Moreover, our algorithms select the smallest support size (i.e., the most parsimonious model) for all n. In contrast, the ℓ_1 penalty (i.e., L1) selects significantly larger support sizes (exceeding 100 in some cases) and suffers in terms of support recovery. It is also worth mentioning that the other ℓ_0-based algorithms, i.e., Algorithm 1, `NHTP` and `GraSP`, outperform MCP and L1 in terms of support recovery (F1 score) and support sizes. The performance of `NHTP` and `GraSP` appears to be similar in terms of support sizes and F1 score, though `NHTP` appears to be performing better compared to `GraSP` in terms of AUC.

In Figure 2, for the medium correlation setting, we see that Algorithms 1 and 2 perform similarly: local search (with CD) performs similar to CD (without local search)—Algorithm 1 can recover the correct support with around 500 observations. In this case, the performance of MCP becomes similar to the ℓ_0-ℓ_q penalized estimators in terms of AUC, though there are notable differences in terms of variable selection properties. Compared to Figure 1, we observe that ℓ_1 performs better in this example, but still appears to be generally outperformed by other algorithms in terms of all measures. We also observe that `NHTP`'s

	Setting 1		Setting 2	
Penalty/Loss	FP	$\|\hat{\boldsymbol{\beta}}\|_0$	FP	$\|\hat{\boldsymbol{\beta}}\|_0$
ℓ_0-ℓ_2/Logistic (Algorithm 1)	0.0 ± 0.0	30.0 ± 0.0	21.6 ± 0.9	26.2 ± 0.5
ℓ_0-ℓ_1/Logistic (Algorithm 1)	0.0 ± 0.0	30.0 ± 0.0	11.2 ± 0.7	14.8 ± 0.3
ℓ_0-ℓ_2/Logistic (Algorithm 2)	0.0 ± 0.0	30.0 ± 0.0	11.5 ± 0.4	14.6 ± 0.3
ℓ_0-ℓ_1/Logistic (Algorithm 2)	0.0 ± 0.0	30.0 ± 0.0	11.2 ± 0.4	14.5 ± 0.3
ℓ_0-ℓ_2/Sq. Hinge (Algorithm 1)	2.8 ± 0.3	32.8 ± 0.3	23.9 ± 0.7	27.0 ± 0.4
ℓ_1/Logistic	617.2 ± 8.3	647.2 ± 8.3	242.2 ± 4.8	256.9 ± 5.3
MCP/Logistic	0.0 ± 0.0	30.0 ± 0.0	80.1 ± 10.9	91.5 ± 12.1
NHTP/Logistic	44.7 ± 5.6	74.7 ± 5.6	92.5 ± 0.6	100.0 ± 0.0
GraSP/Logistic	50.5 ± 5.6	80.5 ± 5.6	45.3 ± 3.0	54.4 ± 2.8

Table 2: Variable selection performance for different penalty and loss combinations, under high-dimensional settings. FP refers to the number of false positives. We consider ten repetitions and report the averages (and standard errors) across the repetitions.

performance is comparable to the best methods in terms of F1 score and AUC. However, it results in models that are more dense compared to L0L1 and L0L2 (both Algorithms 1 and 2). That being said, we will see in Section 5.5 that in terms of running time, Algorithm 1 has a significant edge over `NHTP`. Furthermore, for larger problem instances (Section 5.3) the performance of `NHTP` suffers in terms of variable selection properties compared to the methods we propose in this paper. `NHTP` appears to perform better than `GraSP` across all metrics in this setting.

Figures 1 and 2 suggest that when the correlations are high, local search with ℓ_0 can notably outperform competing methods (especially, in terms of variable selection). When the correlations are medium to low, the differences across the different nonconvex methods become less pronounced. The performance of nonconvex penalized estimators (especially, the ℓ_0-based estimators) is generally better than ℓ_1-based estimators in these examples.

5.3 Performance on Larger Instances

We now study the performance of the different algorithms for some large values of p under the following settings:

- **Setting 1**: $\boldsymbol{\Sigma} = \mathbf{I}, n = 1000, p = 50,000, s = 1000$, and $k^\dagger = 30$
- **Setting 2**: $\Sigma_{ij} = 0.3$ for $i \neq j$, $\Sigma_{ii} = 1$, $n = 1000, p = 10^5, s = 1000$, and $k^\dagger = 20$.

Table 2 reports the results available from Algorithms 1 and 2 ($m = 1$) versus the other methods, for different loss functions. In Settings 1 and 2 above, all methods achieve an AUC of 1 (approximately), and the main differences across the methods lie in variable selection. For Setting 1, both Algorithms 1 and 2 applied to the logistic loss; and `ncvreg` for the MCP penalty (with logistic loss) correctly recover the support. In contrast, ℓ_1 captures a large number of false positives, leading to large support sizes. Both `NHTP` and `GraSP` have a considerable number of false positives and result in large support sizes.

In Setting 2, none of the algorithms correctly recovered the support. This setting is more difficult than Setting 1 as it has higher correlation and a larger p. In Setting 2, we observe that CD with local search can have an edge over plain CD (see, logistic loss with ℓ_0-ℓ_2 penalty). In terms of small FP and compactness of model size, CD with local search appears to be the winner, with Algorithm 1 being quite close. Both Algorithms 1 and 2 appear to work better compared to earlier methods in this setting. We note that in Setting 2, our proposed algorithms select supports with roughly 3 times fewer nonzeros than the MCP penalized problem, and 10 times fewer nonzeros than those delivered by the ℓ_1-penalized problem. For both settings, our proposed methods offer important improvements over `NHTP` and `GraSP` as well.

5.4 Performance on Real Data Sets

We compare the performance of ℓ_1 with ℓ_0-ℓ_q ($q \in \{1, 2\}$) regularization, using the logistic loss function.[11] We consider the following three binary classification data sets taken from the NIPS 2003 Feature Selection Challenge (Guyon et al., 2005):

- **Arcene**: This data set is used to identify cancer vs non-cancerous patterns in mass-spectrometric data. The data matrix is dense with $p = 10,000$ features. We used 140 observations for training and 40 observations for testing.
- **Dorothea**: This data set is used to distinguish active chemical compounds in a drug. The data matrix is sparse with $p = 100,000$ features. We used 805 observations for training and 230 observations for testing.
- **Dexter:** The task here is to identify text documents discussing corporate acquisitions. The data matrix is sparse with $p = 20,000$ features. We used 420 observations for training and 120 observations for testing.

We obtained regularization paths for ℓ_1 using `glmnet` and for ℓ_0-ℓ_2 and ℓ_0-ℓ_1 using both Algorithm 1 and Algorithm 2 (with $m = 1$). In Figure 3, we plot the support size versus the test AUC for ℓ_1 and Algorithm 2 (with $m = 1$) using ℓ_0-ℓ_2. To avoid overcrowded plots, the results for our other algorithms and penalties are presented in Appendix B. For the Arcene data set, ℓ_0-ℓ_2 with $\lambda_2 = 1$ (i.e., the red curve) outperforms ℓ_1 (the green curve) for most of the support sizes, and it reaches a peak AUC of 0.9 whereas ℓ_1 does not exceed 0.84. The other choices of λ_2 also achieve a higher peak AUC than ℓ_1 but do not uniformly outperform it. A similar pattern occurs for the Dorothea data set, where ℓ_0-ℓ_2 with $\lambda_2 = 1$ achieves a peak AUC of 0.9 at around 100 features, whereas ℓ_1 needs around 160 features to achieve the same peak AUC. For the Dexter data set, ℓ_0-ℓ_2 with $\lambda_2 = 10^{-2}$ (the blue curve) achieves a peak AUC of around 0.98 using less than 10 features, whereas ℓ_1 requires around 40 features to achieve a similar AUC. In this case, larger choices of λ_2 do not lead to any significant gains in AUC. This phenomenon is probably due to a higher signal in this data

11. We tried MCP regularization using `ncvreg`, however it ran into convergence issues and did not terminate in over an hour. We also tried NHTP, but it ran into convergence issues and took more than 2 hours to solve for a single solution in the regularization path. Also, note that based on our experiment in Section 5.5, GraSP is slower than NHTP and cannot handle such high-dimensional problems. Therefore, we do not include MCP, NHTP and GraSP in this experiment.

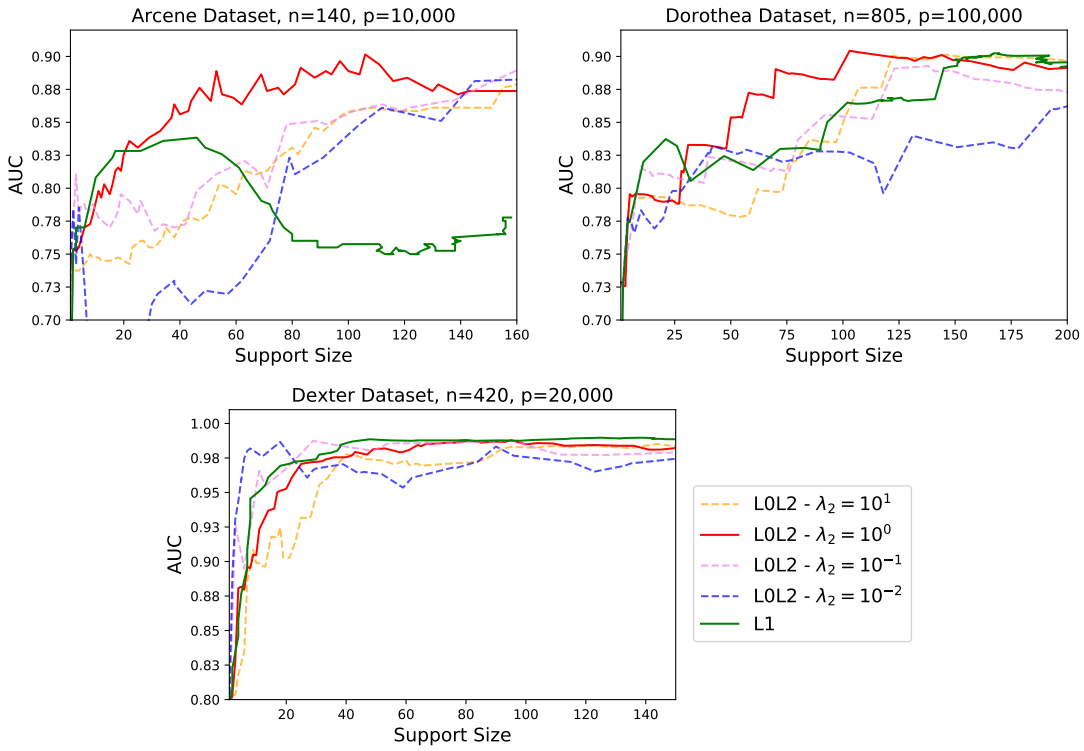

Figure 3: Plots of the AUC versus the support size for the Arcene, Dorothea, and Dexter data sets. The green curves correspond to logistic regression with ℓ_1 regularization. The other curves correspond to logistic regression with ℓ_0-ℓ_2 regularization (using Algorithm 2 with $m = 1$) for different values of λ_2 (see legend).

set compared to the previous two. Overall, we conclude that for all the three data sets, Algorithm 2 for ℓ_0-ℓ_2 can achieve higher AUC-values with (much) smaller support sizes.

5.5 Timings

In this section, we compare the running time of our algorithms versus several state-of-the-art algorithms for sparse classification.

5.5.1 Obtaining good solutions: Upper Bounds

We study the running time of fitting sparse logistic regression models, using the following packages: our package `L0Learn`, `glmnet`, `ncvreg`, and `NHTP`.[12] In `L0Learn`, we consider both Algorithm 1 (denoted by L0Learn 1) and Algorithm 2 with $m = 1$ (denoted by L0Learn 2). We generate synthetic data as described in Section 5.1, with $n = 1000, s = 1, \mathbf{\Sigma} = I$,

12. We also considered `GraSP`, but we found it to be several orders of magnitude slower than the other toolkits, e.g., it takes 3701 seconds at $p = 10^5$. All the other toolkits require less than 70 seconds at $p = 10^5$. Therefore, we did not include `GraSP` in our timing experiments.

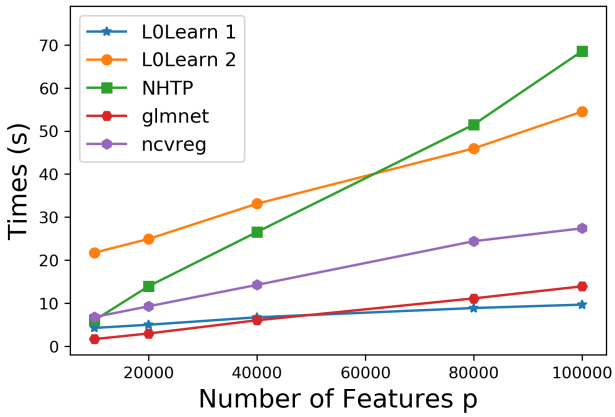

Figure 4: Runtimes to obtain good feasible solutions: Time (s) for obtaining a regularization path (with 100 solutions) for different values of p (as discussed in Section 5.5.1). L0Learn 1 runs Algorithm 1 (CD), while L0Learn 2 runs Algorithm 2 (CD with Local Search).

$k^\dagger = 5$, and we vary p. For all toolkits except `NHTP`,[13] we set the convergence threshold to 10^{-6} and solve for a regularization path with 100 solutions. For `L0Learn`, we use a grid of λ_0-values varying between $\lambda_0^{\max}$ (the value which sets all coefficients to zero) and $0.001\lambda_0^{\max}$. For `L0Learn` and `NHTP`, we set $\lambda_2 = 10^{-7}$. For `glmnet` and `ncvreg`, we compute a path using the default choices of tuning parameters. The experiments were carried out on a machine with a 6-core Intel Core i7-8750H processor and 16GB of RAM, running macOS 10.15.7, R 4.0.3 (with vecLib's BLAS implementation), and MATLAB R2020b. The running times are reported in Figure 4.

Figure 4 indicates that `L0Learn` 1 (Algorithm 1) and `glmnet` achieve the fastest runtimes across the p range. For $p > 40,000$, `L0Learn` 1 (Algorithm 1) runs slightly faster `glmnet`. `L0Learn` 2 (Algorithm 2) is slower due to the local search, but it can obtain solutions in reasonable times, e.g., less than a minute for 100 solutions at $p = 10^5$. It is important to note that the toolkits are optimizing for different objective functions, and the speed-ups in `L0Learn` are partly due to the nature of ℓ_0-regularization which selects fewer nonzeros compared to those available from ℓ_1 or MCP regularization.

5.5.2 Timings for MIP Algorithms: Global Optimality Certificates

We compare the running time to solve Problem (19) by Algorithm 4 (IGA) versus solving (19) directly i.e., without using IGA. We consider the hinge loss and take $q = 2$. We use Gurobi's MIP solver for our experiments. We set $\mathbf{\Sigma} = \mathbf{I}$ and $k^\dagger = 5$ as described in Section 5.1. We then generate $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$ and set $y_i = \text{sign}(\mathbf{x}_i'\boldsymbol{\beta}^\dagger + \epsilon_i)$, where the signal-to-noise ratio SNR=$\text{Var}(\mathbf{X}\boldsymbol{\beta}^\dagger)/\sigma^2 = 10$. We set $n = 1000$ and vary $p \in \{10^4, 2\times10^4, 3\times10^4, 4\times10^4, 5\times10^4\}$. For a fixed $\lambda_2 = 10$, λ_0 is chosen such that the final (optimal) solution has 5 nonzeros. The parameter $\mathcal{M}$ is set to $1.2\|\tilde{\boldsymbol{\beta}}\|_\infty$, where $\tilde{\boldsymbol{\beta}}$ is

13. `NHTP` does not have a parameter to control the tolerance for convergence.

Tookit/p	10000	20000	30000	40000	50000
IGA (this paper)	35	80	117	169	297*
Gurobi	4446	21817	-	-	-

Table 3: Runtimes to certify global optimality: Time(s) for solving an ℓ_0-regularized problem with a hinge loss function, to optimality. "-" denotes that the algorithm does not terminate in a day. "*" indicates that the algorithm is terminated with a 0.05% optimality gap.

the warm start obtained from Algorithm 2. We note that in all cases, the optimal solution recovers the support of the true solution $\boldsymbol{\beta}^\dagger$.

For this experiment, we use a machine with a 12-core Intel Xeon E5 @ 2.7 GHz and 64GB of RAM, running OSX 10.13.6 and Gurobi v8.1. The timings are reported in Table 3. The results indicate significant speed-ups. For example, for $p = 20,000$ our algorithm terminates in 80 seconds whereas Gurobi takes 6 hours to solve (19) (to global optimality). For larger values of p, Gurobi cannot terminate in a day, while our algorithm can terminate to optimality in few minutes. It is worth noting that, empirically we observe IGA can attain low running times when sparse solutions are desired (< 30 nonzeros in our experience). This observation also applies to the state-of-the-art MIP solvers for sparse regression, e.g., see Bertsimas et al. (2016, 2020); Hazimeh et al. (2020). For denser solutions, IGA can still be useful for obtaining optimality gaps, but certifying optimality with a very small optimality gap is expected to take longer.

6. Conclusion

We considered the problem of linear classification regularized with a combination of the ℓ_0 and ℓ_q (for $q \in \{1, 2\}$) penalties. We developed both approximate and exact algorithms for this problem. Our approximate algorithms are based on coordinate descent and local combinatorial search. We established convergence guarantees for these algorithms and demonstrated empirically that they can run in times comparable to the fast ℓ_1-based solvers. Our exact algorithm can solve to optimality high-dimensional instances with $p \approx 50,000$. This scalability is achieved through the novel idea of integrality generation, which solves a sequence of mixed integer programs with a small number of binary variables, until converging to a globally optimal solution. We also established new estimation error bounds for a class of ℓ_0-regularized classification problems and showed that these bounds compare favorably with the best known bounds for ℓ_1 regularization. We carried out experiments on both synthetic and real datasets with p up to 10^5. The results demonstrate that our ℓ_0-based combinatorial algorithms can have a significant statistical edge (in terms of variable selection and prediction) compared to state-of-the-art methods for sparse classification, such as those based on ℓ_1 regularization or simple greedy procedures for ℓ_0 regularization.

There are multiple promising directions for future work. From a modeling perspective, our work can be generalized to structured sparsity problems based on ℓ_0 regularization (Hazimeh et al., 2021). Recent work shows that specialized BnB solvers can be highly

scalable for ℓ_0-regularized regression (Hazimeh et al., 2020). One promising direction is to scale our proposed integrality generation algorithm further by developing specialized BnB solvers for the corresponding MIP subproblems.

Acknowledgments

We would like to thank the anonymous reviewers for their comments that helped improve the paper. The authors acknowledge research funding from the Office of Naval Research [Grants ONR-N000141512342 and ONR-N000141812298 (Young Investigator Award)], and the National Science Foundation [Grant NSF-IIS-1718258].

Appendix A. Proofs and Technical Details

A.1 Proof of Theorem 1

We first present Lemma 9, which establishes that after updating a single coordinate, there is a sufficient decrease in the objective function. The result of this Lemma will be used in the proof of the theorem.

Lemma 9 *Let $\{\beta^l\}$ be the sequence of iterates generated by Algorithm 1. Then, the following holds for any l and $i = 1 + (l \mod p)$:*

$$P(\boldsymbol{\beta}^l) - P(\boldsymbol{\beta}^{l+1}) \geq \frac{\hat{L}_i - L_i}{2}(\beta_i^{l+1} - \beta_i^l)^2. \tag{29}$$

Proof Consider some $l \geq 0$ and fix $i = 1 + (l \mod p)$ (this corresponds to the coordinate being updated via CD). Applying (6) to $\boldsymbol{\beta}^l$ and $\boldsymbol{\beta}^{l+1}$ and adding $\psi(\boldsymbol{\beta}^{l+1})$ to both sides, we have:

$$P(\boldsymbol{\beta}^{l+1}) \leq g(\boldsymbol{\beta}^l) + (\beta_i^{l+1} - \beta_i^l)\nabla_i g(\boldsymbol{\beta}^l) + \frac{L_i}{2}(\beta_i^{l+1} - \beta_i^l)^2 + \psi(\boldsymbol{\beta}^{l+1}).$$

By writing $\frac{L_i}{2}(\beta_i^{l+1} - \beta_i^l)^2$ as the sum of two terms: $\frac{L_i - \hat{L}_i}{2}(\beta_i^{l+1} - \beta_i^l)^2 + \frac{\hat{L}_i}{2}(\beta_i^{l+1} - \beta_i^l)^2$ and regrouping the terms we get:

$$P(\boldsymbol{\beta}^{l+1}) \leq \widetilde{P}_{\hat{L}_i}(\boldsymbol{\beta}^{l+1}; \boldsymbol{\beta}^l) + \frac{L_i - \hat{L}_i}{2}(\beta_i^{l+1} - \beta_i^l)^2, \tag{30}$$

where, in the above equation, we used the definition of $\widetilde{P}$ from (8). Recall from the definition of Algorithm 1 that $\beta_i^{l+1} \in \operatorname{argmin}_{\beta_i} \widetilde{P}_{\hat{L}_i}(\beta_1^l, \ldots, \beta_i, , \ldots, \beta_p^l; \boldsymbol{\beta}^l)$ which implies that $\widetilde{P}_{\hat{L}_i}(\boldsymbol{\beta}^l; \boldsymbol{\beta}^l) \geq \widetilde{P}_{\hat{L}_i}(\boldsymbol{\beta}^{l+1}; \boldsymbol{\beta}^l)$. But $\widetilde{P}_{\hat{L}_i}(\boldsymbol{\beta}^l; \boldsymbol{\beta}^l) = P(\boldsymbol{\beta}^l)$ (this follows directly from (8)). Therefore, we have $P(\boldsymbol{\beta}^l) \geq \widetilde{P}_{\hat{L}_i}(\boldsymbol{\beta}^{l+1}; \boldsymbol{\beta}^l)$. Plugging the latter inequality into (30) and rearranging the terms, we arrive to the result of the lemma. ■

Next, we present the proof of Theorem 1.
Proof

- **Part 1**: We will show that $\text{Supp}(\boldsymbol{\beta}^l) \neq \text{Supp}(\boldsymbol{\beta}^{l+1})$ cannot happen infinitely often. Suppose for some l we have $\text{Supp}(\boldsymbol{\beta}^l) \neq \text{Supp}(\boldsymbol{\beta}^{l+1})$. Then, for $i = 1 + (l \mod p)$, one of the following two cases must hold: (I) $\beta_i^l = 0 \neq \beta_i^{l+1}$ or (II) $\beta_i^l \neq 0 = \beta_i^{l+1}$. Let us consider case (I). Recall that the minimization step in Algorithm 1 is done using the thresholding operator defined in (10). Since $\beta_i^{l+1} \neq 0$, (10) implies that $|\beta_i^{l+1}| \geq \sqrt{\frac{2\lambda_0}{\hat{L}_i + 2\lambda_2}}$. Plugging the latter bound into the result of Lemma 9, we arrive to

$$P(\boldsymbol{\beta}^l) - P(\boldsymbol{\beta}^{l+1}) \geq \frac{\hat{L}_i - L_i}{\hat{L}_i + 2\lambda_2}\lambda_0, \tag{31}$$

  where the r.h.s. of the above is positive, due to the choice $\hat{L}_i > L_i$. The same argument can be used for case (II) to arrive to (31). Thus, whenever the support changes, (31) applies, and consequently the objective function decreases by a constant value. Therefore, the support cannot change infinitely often (as $P(\boldsymbol{\beta})$ is bounded below).

- **Part 2**: First, we will show that under Assumption 1, the function $\boldsymbol{\beta}_S \mapsto G(\boldsymbol{\beta}_S)$ is strongly convex. This holds for the first case of Assumption 1, i.e., when $\lambda_2 > 0$. Next, we will consider the second case of Assumption 1 which states that $P(\boldsymbol{\beta}^0) < \lambda_0 u$ and $g(\boldsymbol{\beta})$ is strongly convex when restricted to a support of size at most u. Since Algorithm 1 is a descent algorithm, we get $P(\boldsymbol{\beta}^l) < \lambda_0 u$ for any $l \geq 0$. This implies $\|\boldsymbol{\beta}^l\|_0 < u$ for any $l \geq 0$, and thus $|S| < u$. Consequently, the function $\boldsymbol{\beta}_S \mapsto g(\boldsymbol{\beta}_S)$ is strongly convex (and so is $\boldsymbol{\beta}_S \mapsto G(\boldsymbol{\beta}_S)$).

  After support stabilization (from Part 1), the iterates generated by Algorithm 1 are the same as those generated by CD for minimizing the strongly convex function $G(\boldsymbol{\beta}_S)$, which is guaranteed to converge (e.g., see Bertsekas 2016). Therefore, we conclude that $\{\boldsymbol{\beta}^l\}$ converges to a stationary solution $\boldsymbol{\beta}^*$ satisfying $\boldsymbol{\beta}^*_S \in \text{argmin}_{\boldsymbol{\beta}_S} G(\boldsymbol{\beta}_S)$ and $\boldsymbol{\beta}^*_{S^c} = \mathbf{0}$.

  Finally, we will prove the two inequalities in (11). Fix some $i \in S$. Then, from the definition of the thresholding operator in (10), the following holds for every $l > N$ (i.e., after support stabilization) at which coordinate i is updated:

$$|\beta_i^l| \geq \sqrt{\frac{2\lambda_0}{\hat{L}_i + 2\lambda_2}}. \tag{32}$$

  Taking the limit as $l \to \infty$ we arrive to the first inequality in (11), which also implies that $\text{Supp}(\boldsymbol{\beta}^*) = S$. Now let us fix some $i \in S^c$. For every $l > N$ at which coordinate i is updated, we have $\beta_i^l = 0$ and the following condition holds by the definition of the thresholding operator in (10):

$$\frac{\hat{L}_i}{\hat{L}_i + 2\lambda_2}\Big(\Big|\frac{\nabla_i g(\boldsymbol{\beta}^l)}{\hat{L}_i}\Big| - \frac{\lambda_1}{\hat{L}_i}\Big) < \sqrt{\frac{2\lambda_0}{\hat{L}_i + 2\lambda_2}}. \tag{33}$$

  Taking the limit as $l \to \infty$ in the above and simplifying, we arrive to the second inequality in (11).

- **Part 3**: The support stabilization on S (from Part 1) and the fact that $\boldsymbol{\beta}^l \to \boldsymbol{\beta}^*$ where $\text{Supp}(\boldsymbol{\beta}^*) = S$ (from Part 2), directly imply that $\text{sign}(\boldsymbol{\beta}^l_S)$ stabilizes in a finite number of iterations. That is, there exists an integer N' and a vector $\boldsymbol{t} \in \{-1, 1\}^{|S|}$ such that $\text{sign}(\boldsymbol{\beta}^l_S) = \boldsymbol{t}$ for all $l \geq N'$. Therefore, for $l \geq N'$, the iterates of Algorithm 1 are the same as those generated by running coordinate descent to minimize the continuously differentiable function:

$$g(\boldsymbol{\beta}) + \lambda_0|S| + \lambda_1(\sum_{i:t_i>0} \beta_i - \sum_{i:t_i<0} \beta_i) + \lambda_2\|\boldsymbol{\beta}\|_2^2. \tag{34}$$

  Beck and Tetruashvili (2013) established a linear rate of convergence for the case of CD applied to a class of strongly convex and continuously differentiable functions, which includes 34. Applying Beck and Tetruashvili (2013)'s result to (34) leads to (12).

■

A.2 Proof of Theorem 3

Proof First, we will show that the algorithm terminates in a finite number of iterations. Suppose the algorithm does not terminate after T iterations. Then, we have a sequence $\{\boldsymbol{\beta}^t\}_0^T$ of stationary solutions (since these are the outputs of Algorithm 1—see Theorem 1). Let $S_t = \text{Supp}(\boldsymbol{\beta}^t)$. Each stationary solution $\boldsymbol{\beta}^t$ is a minimizer of the convex function $\boldsymbol{\beta}_{S_t} \mapsto G(\boldsymbol{\beta}_{S_t})$, and consequently $P(\boldsymbol{\beta}^t) = \min\{P(\boldsymbol{\beta}) \mid \boldsymbol{\beta} \in \mathbb{R}^p, \text{Supp}(\boldsymbol{\beta}) = S_t\}$. Moreover, the definition of Step 2 and the descent property of cyclic CD imply $P(\boldsymbol{\beta}^T) < P(\boldsymbol{\beta}^{T-1}) < \cdots < P(\boldsymbol{\beta}^0)$. Therefore, the same support cannot appear more than once in the sequence $\{\boldsymbol{\beta}^t\}_0^T$, and we conclude that the algorithm terminates in a finite number of iterations with a solution $\boldsymbol{\beta}^*$. Finally, we note that $\boldsymbol{\beta}^*$ is the output of Algorithm 1 so it must satisfy the characterization given in Theorem 1. Moreover, the search in Step 2 must fail at $\boldsymbol{\beta}^*$ (otherwise, the algorithm does not terminate), and thus (15) holds. ■

A.3 Proof of Lemma 5

Proof First, we recall that $\delta_j = |\hat{L}\beta_j^* - \nabla_j g(\boldsymbol{\beta}^*)|$ for any $j \in [p]$. Let $S = \text{Supp}(\boldsymbol{\beta}^*)$ and fix some $j \in S$. By Theorem 1, we have $\boldsymbol{\beta}^*_S \in \text{argmin}_{\boldsymbol{\beta}_S}\ G(\boldsymbol{\beta}_S)$, which is equivalent to $0 \in \partial G(\boldsymbol{\beta}^*_S)$. The zero subgradient condition directly implies that $\nabla_j g(\boldsymbol{\beta}^*_S) = -\lambda_1 \text{sign}(\beta_j^*) - 2\lambda_2|\beta_j^*|$. Substituting this expression into δ_j, we get:

$$\begin{aligned}
\delta_j &= |(\hat{L} + 2\lambda_2)\beta_j^* + \lambda_1 \text{sign}(\beta_j^*)| \\
&= (\hat{L} + 2\lambda_2)|\beta_j^*| + \lambda_1 \\
&\geq \sqrt{2\lambda_0(\hat{L} + 2\lambda_2)} + \lambda_1,
\end{aligned} \tag{35}$$

where (35) follows from inequality $|\beta_j^*| \geq \sqrt{\frac{2\lambda_0}{\hat{L}_j+2\lambda_2}}$ (due to Theorem 1) and the fact that $\hat{L} \geq \hat{L}_j$. Now fix some $i \notin S$. Using Theorem 1 and $\hat{L} \geq \hat{L}_i$:

$$\delta_i = |\nabla_i g(\boldsymbol{\beta}^*)| \leq \sqrt{2\lambda_0(\hat{L}_i + 2\lambda_2)} + \lambda_1 \leq \sqrt{2\lambda_0(\hat{L} + 2\lambda_2)} + \lambda_1. \tag{36}$$

Inequalities (35) and (36) imply that $\delta_j \geq \delta_i$ for any $j \in S$ and $i \notin S$. Since $\|\boldsymbol{\beta}^*\|_0 = k$, we have $\delta_{(k)} \geq \delta_i$ for any $i \notin S$, which combined with the fact that $\boldsymbol{\beta}_S^* \in \operatorname{argmin}_{\boldsymbol{\beta}_S} G(\boldsymbol{\beta}_S)$ (from Theorem 1) implies that $\boldsymbol{\beta}^*$ satisfies the fixed point conditions for IHT stated in Theorem 4. ■

A.4 Choice of J: sorted gradients

Let β_j^1 denote the solution obtained after the first application of the thresholding operator to minimize the function in (16). Note that we are interested in coordinates with nonzero β_j^1 (because in step 1 of Algorithm 3, we check whether β_j should be zero). Coordinates with nonzero β_j^1 must satisfy $|\nabla_j g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)| - \lambda_1 > 0$ (this follows from (10) with $\lambda_0 = 0$). Using (6), it can be readily seen that we have the following lower bound on the improvement in the objective if $|\nabla_j g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)| - \lambda_1 > 0$:

$$P(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t) - P(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t + \boldsymbol{e}_j\beta_j^1) \geq \frac{1}{2(L_j + 2\lambda_2)}(|\nabla_j g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)| - \lambda_1)^2 - \lambda_0. \tag{37}$$

The choices of $j \in S^c$ with larger $|\nabla_j g(\boldsymbol{\beta}^t - \boldsymbol{e}_i\beta_i^t)|$ have a larger r.h.s. in inequality (37) and thus are expected to have lower objectives. Therefore, instead of searching across all values of $j \in S^c$ in Step 2 of Algorithm 2, we restrict $j \in J$.

A.5 Proof of Lemma 6

Proof Problem (21) can be rewritten as

$$\begin{aligned}\min_{\boldsymbol{\beta}, \boldsymbol{z}_I} \Big[\min_{\boldsymbol{z}_{\mathcal{I}^c}} \quad & G(\boldsymbol{\beta}) + \lambda_0 \sum_{i=1}^{p} z_i \Big] \\ & |\beta_i| \leq \mathcal{M} z_i, \quad i \in [p] \\ & z_i \in [0,1], \quad i \notin \mathcal{I} \\ & z_i \in \{0,1\}, \quad i \in \mathcal{I}.\end{aligned}$$

Solving the inner minimization problem leads to $z_i = |\beta_i|/\mathcal{M}$ for every $i \in \mathcal{I}^c$. Plugging the latter solution into the objective function, we arrive to the result of the lemma. ■

A.6 Proof of Theorem 7

A.6.1 A useful proposition

Proposition 1 (stated below) is an essential step in the proof of Theorem 7. This allows us to control the supremum of a random variable (of interest) over a bounded set of $2k$ sparse vectors.

Proposition 1 *Let $\delta \in (0, 1/2)$, $\tau = 6L\sqrt{\frac{\lambda(k)}{n}\left(k \log\left(Rp/k\right) + \log\left(1/\delta\right)\right)}$ and define*

$$\Delta(\mathbf{z}) = \frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}^* + \mathbf{z}\rangle; y_i\right) - \frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}^*\rangle; y_i\right), \quad \forall \mathbf{z}. \tag{38}$$

If Assumptions 2, 4(k) and 5(δ) hold then:

$$\mathbb{P}\left(\sup_{\substack{\mathbf{z}\in\mathbb{R}^p \\ \|\mathbf{z}\|_0 \leq 2k,\ \|\mathbf{z}\|_2 \leq 3R}} \left\{\left|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)\right| - \tau\|\mathbf{z}\|_2 \vee \tau^2\right\} \geq 0\right) \leq \delta.$$

Proof We divide the proof into 3 steps. First, we upper-bound the quantity $|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)|$ for any $2k$ sparse vector with Hoeffding inequality. Second, we extend the result to the maximum over an ϵ-net. We finally control the maximum over the compact set and derive our proposition.

Step 1: We fix $\mathbf{z} \in \mathbb{R}^p$ such that $\|\mathbf{z}\|_0 \leq 2k$ and introduce the random variables $Z_i, \forall i$ as follows

$$Z_i = f\left(\langle \mathbf{x}_i, \boldsymbol{\beta^*} + \mathbf{z}\rangle; y_i\right) - f\left(\langle \mathbf{x}_i, \boldsymbol{\beta^*}\rangle; y_i\right).$$

Assumption 2 guarantees that $f(.; y)$ is Lipschitz with constant L, which leads to:

$$|Z_i| \leq L\left|\langle \mathbf{x}_i, \mathbf{z}\rangle\right|.$$

Note that $\Delta(\mathbf{z}) = \frac{1}{n}\sum_{i=1}^{n} Z_i$. We introduce a small quantity $\eta > 0$ later explicited in the proof. Using Hoeffding's inequality and Assumption $4(k)$ it holds: $\forall t > 0$,

$$\begin{aligned}
\mathbb{P}\left(\left|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)\right| \geq t\left(\|\mathbf{z}\|_2 \vee \eta\right)\middle|\mathbf{X}\right) &\leq 2\exp\left(-\frac{2n^2t^2\left(\|\mathbf{z}\|_2 \vee \eta\right)^2}{\sum_{i=1}^{n} L^2\langle \mathbf{x}_i, \mathbf{z}\rangle^2}\right) \\
&= 2\exp\left(-\frac{2n^2t^2\left(\|\mathbf{z}\|_2^2 \vee \eta^2\right)}{L^2\|\mathbf{X}\mathbf{z}\|_2^2}\right) \\
&\leq 2\exp\left(-\frac{2nt^2\left(\|\mathbf{z}\|_2^2 \vee \eta^2\right)}{L^2\lambda(k)\|\mathbf{z}\|_2^2}\right) \\
&\leq 2\exp\left(-\frac{2nt^2}{L^2\lambda(k)}\right).
\end{aligned} \tag{39}$$

Note that in the above display, the r.h.s. bound does not depend upon $\mathbf{X}$, the conditioning event in the l.h.s. of display (39).

Step 2: We consider an ϵ-net argument to extend the result to any $2k$ sparse vector satisfying $\|\mathbf{z}\|_2 \leq 3R$. We recall that an ϵ-net of a set $\mathcal{I}$ is a subset $\mathcal{N}$ of $\mathcal{I}$ such that each element of $\mathcal{I}$ is at a distance at most ϵ of $\mathcal{N}$.

Lemma 1.18 in Rigollet (2015) proves that for any value $\epsilon \in (0, 1)$, the ball $\left\{\mathbf{z} \in \mathbb{R}^d : \|\mathbf{z}\|_2 \leq 3R\right\}$ has an ϵ-net of cardinality $|\mathcal{N}| \leq \left(\frac{6R+1}{\epsilon}\right)^d$. Consequently, we can

fix an ϵ-net $\mathcal{N}_{k,R}$ of the set $\mathcal{I}_{k,R} = \{\mathbf{z} \in \mathbb{R}^p : \|\mathbf{z}\|_0 = 2k \; ; \; \|\mathbf{z}\|_2 \leq 3R\}$ with cardinality $\binom{p}{2k}\left(\frac{6R+1}{\epsilon}\right)^{2k}$. This along with equation (39) leads to: $\forall t > 0$,

$$\begin{aligned}\mathbb{P}\left(\sup_{\mathbf{z}\in\mathcal{N}_{k,R}}\left(\left|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)\right| - t\left(\|\mathbf{z}\|_2 \vee \eta\right)\right) \geq 0\right) \\ \leq \binom{p}{2k}\left(\frac{6R+1}{\epsilon}\right)^{2k} 2\exp\left(-\frac{2nt^2}{L^2\lambda(k)}\right).\end{aligned} \tag{40}$$

Step 3: We finally extend the result to any vector in $\mathcal{I}_{k,R}$. This is done by expressing the supremum of the random variable $|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)|$ over the entire set $\mathcal{I}_{k,R}$ with respect to its supremum over the ϵ-net $\mathcal{N}_{k,R}$.

For $\mathbf{z} \in \mathcal{I}_{k,R}$, there exists $\mathbf{z}_0 \in \mathcal{N}_{k,R}$ such that $\|\mathbf{z} - \mathbf{z}_0\|_2 \leq \epsilon$. With Assumption 2 and Cauchy-Schwartz inequality, we obtain:

$$|\Delta(\mathbf{z}) - \Delta(\mathbf{z}_0)| \leq \frac{1}{n}\sum_{i=1}^{n} L\left|\langle \mathbf{x}_i, \mathbf{z} - \mathbf{z}_0\rangle\right| \leq \frac{1}{\sqrt{n}} L\|\mathbf{X}(\mathbf{z} - \mathbf{z}_0)\|_2 \leq L\sqrt{\lambda(k)}\epsilon. \tag{41}$$

For a fixed value of t, we define the operator

$$f_t(\mathbf{z}) = |\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)| - t\left(\|\mathbf{z}\|_2 \vee \eta\right), \; \forall \mathbf{z}.$$

Using the (reverse) triangle inequality, it holds that:

$$|\Delta(\mathbf{z}) - \Delta(\mathbf{z}_0)| \geq |\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)| - |\Delta(\mathbf{z}_0) - \mathbb{E}\left(\Delta(\mathbf{z}_0)\right)| - \left|\mathbb{E}\left(\Delta(\mathbf{z})\right) - \mathbb{E}\left(\Delta(\mathbf{z}_0)\right)\right|. \tag{42}$$

In addition, note that:

$$\left(\|\mathbf{z} - \mathbf{z}_0\|_2 \vee \eta\right) \quad + \quad \left(\|\mathbf{z}\|_2 \vee \eta\right) \quad \geq \quad \|\mathbf{z}_0\|_2 \vee \eta. \tag{43}$$

Using (42) and (43) in $f_t(\mathbf{z})$ we have:

$$f_t(\mathbf{z_0}) \geq f_t(\mathbf{z}) - |\Delta(\mathbf{z}) - \Delta(\mathbf{z}_0)| - \left|\mathbb{E}\left(\Delta(\mathbf{z}) - \Delta(\mathbf{z}_0)\right)\right| - t\left(\|\mathbf{z} - \mathbf{z}_0\|_2 \vee \eta\right). \tag{44}$$

Now using (41) and Jensen's inequality, we have:

$$\begin{aligned}|\Delta(\mathbf{z}) - \Delta(\mathbf{z}_0)| &\leq L\sqrt{\lambda(k)}\epsilon \\ \left|\mathbb{E}\left(\Delta(\mathbf{z}) - \Delta(\mathbf{z}_0)\right)\right| &\leq L\sqrt{\lambda(k)}\epsilon.\end{aligned} \tag{45}$$

Applying (45) and $\|\mathbf{z} - \mathbf{z}_0\|_2 \leq \epsilon$ to the right hand side of (44), we have:

$$f_t(\mathbf{z}_0) \geq f_t(\mathbf{z}) - 2L\sqrt{\lambda(k)}\epsilon - t(\epsilon \vee \eta). \tag{46}$$

Suppose we choose

$$\begin{aligned}\eta &= 4L\sqrt{\frac{\lambda(k)}{n}k\log\left(p/k\right)} \\ \epsilon = \frac{\eta^2}{4L\sqrt{\lambda(k)}} \quad &= \sqrt{\lambda(k)}\frac{4Lk\log\left(p/k\right)}{n}\end{aligned}$$

$$t \geq \eta/4.$$

This implies that:

$$\epsilon \leq \eta \qquad \text{(using } k/n\log(p/k) \leq 1 \text{ by Assumption 5)},$$

$$L\sqrt{\lambda(k)}\epsilon \leq t\eta \qquad \text{(using } t \geq \eta/4 \implies t\eta \geq \frac{\eta^2}{4} = L\sqrt{\lambda(k)}\epsilon).$$

Using the above, we obtain a lower bound to the r.h.s. of (46). This leads to the following chain of inequalities:

$$\begin{aligned} f_t(\mathbf{z}) - 2L\sqrt{\lambda(k)}\epsilon - t(\epsilon \vee \eta) &\geq f_t(\mathbf{z}) - 3t\eta \\ &= |\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)| - t\left(\|\mathbf{z}\|_2 \vee \eta\right) - 3t\eta \\ &\geq |\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)| - 4t\left(\|\mathbf{z}\|_2 \vee \eta\right) \\ &= f_{4t}(\mathbf{z}). \end{aligned} \tag{47}$$

Consequently, Equations (46) and (47) lead to:

$$\forall t \geq \eta/4,\ \forall \mathbf{z} \in \mathcal{I}_{k,R},\ \exists \mathbf{z}_0 \in \mathcal{N}_{k,R} :\ f_{4t}(\mathbf{z}) \leq f_t(\mathbf{z}_0),$$

which can be equivalently written as:

$$\sup_{\mathbf{z}\in\mathcal{I}_{k,R}} f_t(\mathbf{z}) \leq \sup_{\mathbf{y}\in\mathcal{N}_{k,R}} f_{t/4}(\mathbf{y}),\ \forall t \geq \eta. \tag{48}$$

Lemma 2.7 in Rigollet (2015) gives the relation $\binom{p}{2k} \leq \left(\frac{pe}{2k}\right)^{2k}$. Using equation (48) along with the union-bound (40), it holds that: $\forall t \geq \eta$,

$$\begin{aligned} \mathbb{P}\left(\sup_{\mathbf{z}\in\mathcal{I}_{k,R}} f_t(\mathbf{z}) \geq 0\right) &\leq \mathbb{P}\left(\sup_{\mathbf{z}\in\mathcal{N}_{k,R}} f_{t/4}(\mathbf{z}) \geq 0\right) \\ &\leq \binom{p}{2k}\left(\frac{6R+1}{\epsilon}\right)^{2k} 2\exp\left(-\frac{2nt^2}{16L^2\lambda(k)}\right) \\ &\leq \left(\frac{pe}{2k}\right)^{2k}\left(\frac{6R+1}{\epsilon}\right)^{2k} 2\exp\left(-\frac{2nt^2}{16L^2\lambda(k)}\right) \\ &\leq 2\left(\frac{pe}{2k}\ \frac{7R}{\epsilon}\right)^{2k}\exp\left(-\frac{2nt^2}{16L^2\lambda(k)}\right). \end{aligned} \tag{49}$$

By Assumption 5 we have $7en \leq 3L\sqrt{\lambda(k)}p\log(p/k)$; and using the definition of ϵ (above), it follows that:

$$\frac{7e}{2\epsilon} \leq \frac{p}{k} \leq \frac{Rp}{k},$$

where in the last inequality we used the assumption $R \geq 1$. Using this in (49) we have:

$$\mathbb{P}\left(\sup_{\mathbf{z}\in\mathcal{I}_{k,R}} f_t(\mathbf{z}) \geq 0\right) \leq 2\left(\frac{Rp}{k}\right)^{4k}\exp\left(-\frac{2nt^2}{16L^2\lambda(k)}\right),\ \forall t \geq \eta. \tag{50}$$

We want the right-hand side of Equation (50) to be smaller than δ. To this end, we need to select $t^2 \geq \frac{16L^2\lambda(k)}{2n}\left[4k\log\left(\frac{Rp}{k}\right)+\log(2)+\log\left(\frac{1}{\delta}\right)\right]$ and $t \geq \eta$. A possible choice for t that we use is:

$$\tau = 6L\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right)+\log\left(1/\delta\right)\right)}.$$

We conclude that with probability at least $1-\delta$:

$$\sup_{\mathbf{z}\in\mathcal{I}_{k,R}} f_\tau(\mathbf{z}) \leq 0.$$

Note that

$$f_\tau(\mathbf{z}) = \left|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)\right| - \tau\left(\|\mathbf{z}\|_2 \vee \eta\right) \geq \left|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)\right| - \tau\|\mathbf{z}\|_2 \vee \tau^2.$$

It then holds with probability at least $1-\delta$ that:

$$\sup_{\substack{\mathbf{z}\in\mathbb{R}^p \\ \|\mathbf{z}\|_0\leq 2k,\ \|\mathbf{z}\|_2\leq 3R}} \left\{\left|\Delta(\mathbf{z}) - \mathbb{E}\left(\Delta(\mathbf{z})\right)\right| - \tau\|\mathbf{z}\|_2 \vee \tau^2\right\} \leq 0,$$

which concludes the proof. ■

A.6.2 Proof of Theorem 7

Proof The $2k$ sparse vector $\mathbf{h} = \hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*$ satisfies: $\|\mathbf{h}\|_2 \leq \|\hat{\boldsymbol{\beta}}\|_2 + \|\boldsymbol{\beta}^*\|_2 \leq 3R$. Thus applying Proposition 1 to $\Delta(\mathbf{h})$ (see the definition in Equation 38) it holds with probability at least $1-\delta$ that:

$$\begin{aligned}
\Delta(\mathbf{h}) &\geq \mathbb{E}\left(\Delta(\mathbf{h})\right) - \tau\|\mathbf{h}\|_2 \vee \tau^2 \\
&= \frac{1}{n}\mathbb{E}\left(\sum_{i=1}^n f\left(\langle\mathbf{x}_i, \boldsymbol{\beta}^* + \mathbf{h}\rangle; y_i\right) - \sum_{i=1}^n f\left(\langle\mathbf{x}_i, \boldsymbol{\beta}^*\rangle; y_i\right)\right) - \tau\|\mathbf{h}\|_2 \vee \tau^2 \\
&= \mathbb{E}\left[f\left(\langle\mathbf{x}_i, \boldsymbol{\beta}^* + \mathbf{h}\rangle; y_i\right) - f\left(\langle\mathbf{x}_i, \boldsymbol{\beta}^*\rangle; y_i\right)\right] - \tau\|\mathbf{h}\|_2 \vee \tau^2 \\
&= \mathcal{L}(\boldsymbol{\beta}^* + \mathbf{h}) - \mathcal{L}(\boldsymbol{\beta}^*) - \tau\|\mathbf{h}\|_2 \vee \tau^2.
\end{aligned} \tag{51}$$

To control the difference $\mathcal{L}(\boldsymbol{\beta}^* + \mathbf{h}) - \mathcal{L}(\boldsymbol{\beta}^*)$ we consider the following two cases.
Case 1 ($\|\mathbf{h}\|_2 \leq r(k)$): If $\|\mathbf{h}\|_2 \leq r(k)$ then by definition of $r(k)$ in (26) it holds that

$$\mathcal{L}(\boldsymbol{\beta}^* + \mathbf{h}) - \mathcal{L}(\boldsymbol{\beta}^*) \geq \frac{1}{4}\kappa(k)\|\mathbf{h}\|_2^2. \tag{52}$$

Case 2: We consider the case when $\|\mathbf{h}\|_2 > r(k)$. Since $\boldsymbol{\beta} \mapsto \mathcal{L}(\boldsymbol{\beta})$ is convex, the one-dimensional function $t \to \mathcal{L}\left(\boldsymbol{\beta}^* + t\mathbf{z}\right)$ is convex and it holds:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\beta}^* + \mathbf{h}) - \mathcal{L}(\boldsymbol{\beta}^*) &\geq \frac{\|\mathbf{h}\|_2}{r(k)} \left\{ \mathcal{L}\left(\boldsymbol{\beta}^* + \frac{r(k)}{\|\mathbf{h}\|_2}\mathbf{h}\right) - \mathcal{L}(\boldsymbol{\beta}^*) \right\} \\
&\geq \frac{\|\mathbf{h}\|_2}{r(k)} \inf_{\substack{\mathbf{z}:\ \|\mathbf{z}\|_0 \leq 2k \\ \|\mathbf{z}\|_2 = r(k)}} \{\mathcal{L}(\boldsymbol{\beta}^* + \mathbf{z}) - \mathcal{L}(\boldsymbol{\beta}^*)\} \\
&\geq \frac{\|\mathbf{h}\|_2}{r(k)} \ \frac{1}{4}\kappa(k) r(k)^2 \\
&= \frac{1}{4}\kappa(k) r(k) \|\mathbf{h}\|_2.
\end{aligned} \tag{53}$$

In (53), the first inequality follows by observing that the one-dimensional function $x \mapsto (g(a+x) - g(a))/x$ defined on $x > 0$ is increasing for any one-dimensional convex function $x \mapsto g(x)$. The last inequality in display (53) follows from the definition of $r(k)$ as in (26).

Combining Equations (51), (52) and (53), we obtain the following restricted strong convexity property (which holds with probability at least $1 - \delta$):

$$\Delta(\mathbf{h}) \geq \frac{1}{4}\kappa(k) \left\{ \|\mathbf{h}\|_2^2 \wedge r(k)\|\mathbf{h}\|_2 \right\} - \tau\|\mathbf{h}\|_2 \vee \tau^2.$$

■

A.7 Proof of Theorem 8

Proof Using the observation that $\hat{\boldsymbol{\beta}}$ is a minimizer of Problem (24); and the representation $\hat{\boldsymbol{\beta}} = \boldsymbol{\beta}^* + \mathbf{h}$, it holds that:

$$\frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}^* + \mathbf{h}\rangle; y_i\right) \leq \frac{1}{n}\sum_{i=1}^{n} f\left(\langle \mathbf{x}_i, \boldsymbol{\beta}^*\rangle; y_i\right).$$

This relation is equivalent to saying that $\Delta(\mathbf{h}) \leq 0$. Consequently, by combining this relation with (27) (see Theorem 7), it holds with probability at least $1 - \delta$:

$$\frac{1}{4}\kappa(k) \left\{ \|\mathbf{h}\|_2^2 \wedge r(k)\|\mathbf{h}\|_2 \right\} \leq \tau\|\mathbf{h}\|_2 \vee \tau^2. \tag{54}$$

We now consider two cases.
Case 1: Let $\|\mathbf{h}\|_2 \leq \tau$. Then we have that

$$\|\mathbf{h}\|_2 \leq \tau = 6L\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right) + \log\left(1/\delta\right)\right)}.$$

Case 2: We consider the case where $\|\mathbf{h}\|_2 > \tau$. With probability $1 - \delta$ it holds (from Inequality 54) that:

$$\frac{1}{4}\kappa(k) \left\{ \|\mathbf{h}\|_2^2 \wedge r(k)\|\mathbf{h}\|_2 \right\} \leq \tau\|\mathbf{h}\|_2,$$

which can be simplified to:

$$\|\mathbf{h}\|_2 \wedge r(k) \leq \frac{4\tau}{\kappa(k)} = \frac{24L}{\kappa(k)}\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right)+\log\left(1/\delta\right)\right)}. \tag{55}$$

Note that by Assumption 5(δ), the r.h.s. of the above is smaller than $r(k)$. The l.h.s. of (55) is the minimum of two terms $\|\mathbf{h}\|_2$ and $r(k)$; and since this is lower than $r(k)$, we conclude that

$$\|\mathbf{h}\|_2 \leq \frac{24L}{\kappa(k)}\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right)+\log\left(1/\delta\right)\right)}.$$

Combining the results from Cases 1 and 2, we conclude that with probability at least $1-\delta$, the following holds:

$$\|\mathbf{h}\|_2 \leq CL\max\left\{1, \frac{1}{\kappa(k)}\right\}\sqrt{\frac{\lambda(k)}{n}\left(k\log\left(Rp/k\right)+\log\left(1/\delta\right)\right)},$$

where C is an universal constant. This concludes the proof of the theorem.

■

A.8 Proof of Corollary 1:

Proof Let us define the random variable:

$$W = \frac{1}{L^2\widetilde{\kappa}^2\lambda(k)}\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2.$$

Since the difference $\hat{\boldsymbol{\beta}}-\boldsymbol{\beta}^*$ is bounded, then W is upper-bounded by a constant. In addition, because Assumption 5 is satisfied for $\delta > 0$ small enough, Theorem 8 leads to the existence of a constant C such that

$$\mathbb{P}\left(W \leq \frac{C}{n}\left(k\log(Rp/k)+\log(1/\delta)\right)\right) \geq 1-\delta,\ \forall\delta\in(0,1).$$

This relation can be equivalently expressed as

$$\mathbb{P}\left(W/C \geq \frac{k\log(p/k)}{n} + t\right) \leq e^{-nt},\ \forall t \geq 0.$$

Let us define $H = (k/n)\log(p/k)$. By integration it holds:

$$\begin{aligned}
\mathbb{E}(W) &= \int_0^{+\infty} C\mathbb{P}\left(|W|/C \geq t\right)dt \\
&\leq \int_0^{+\infty} C\mathbb{P}\left(|W|/C \geq t+H\right)dt + CH \\
&\leq \int_0^{+\infty} Ce^{-nt}dt + CH = \frac{C}{n} + CH \leq 2CH
\end{aligned} \tag{56}$$

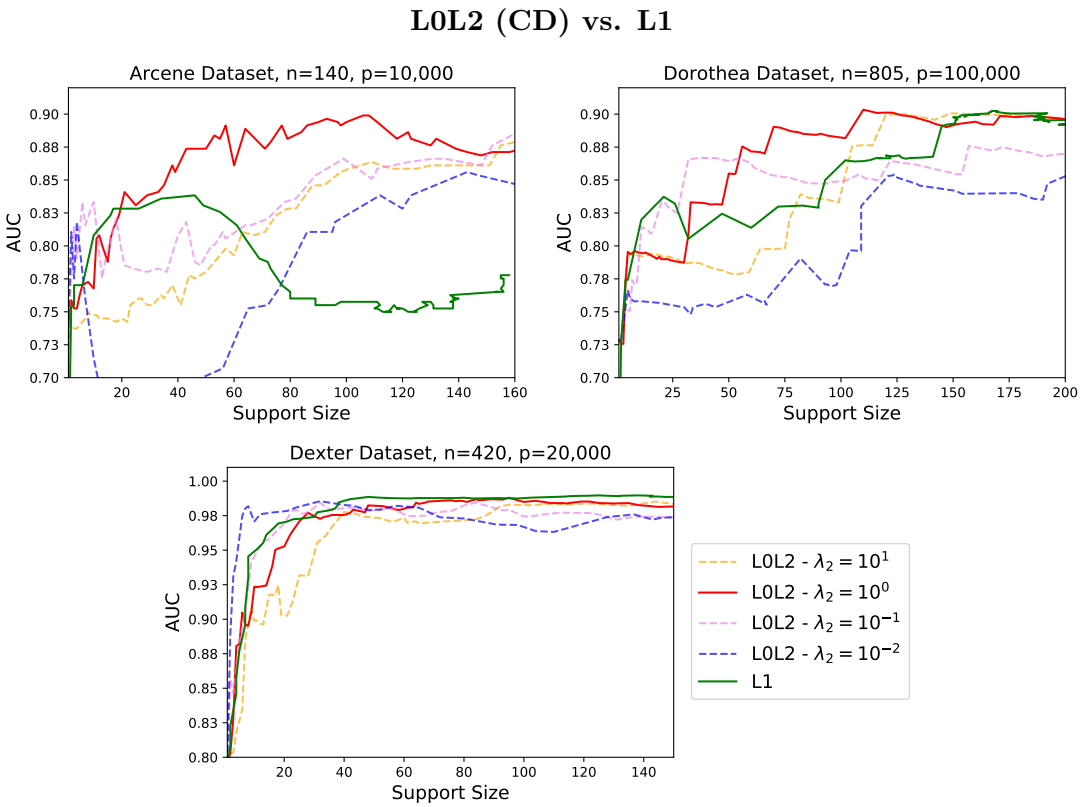

Figure 5: Plots of AUC versus corresponding support sizes for the Arcene, Dorothea, and Dexter data sets. The green curves correspond to logistic regression with ℓ_1 regularization. The other curves correspond to logistic regression with ℓ_0-ℓ_2 regularization using Algorithm 1 for different values of λ_2 (see legend).

We finally conclude:

$$\mathbb{E}\|\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}^*\|_2^2 \lesssim L^2 \lambda(k) \widetilde{\kappa}^2 \frac{k \log(p/k)}{n}.$$

■

Appendix B. Additional Experiments

Here we present additional experimental results complementing the real data set experiments presented in Section 5.4. We compare the performance of our proposed algorithm with ℓ_1-regularization. In particular, Figure 5 considers the ℓ_0-ℓ_2 penalty with CD (Algorithm 1); Figure 6 considers the ℓ_0-ℓ_1 penalty with local search (Algorithm 2 with $m = 1$); and Figure 7 considers the ℓ_0-ℓ_1 penalty with CD (Algorithm 1).

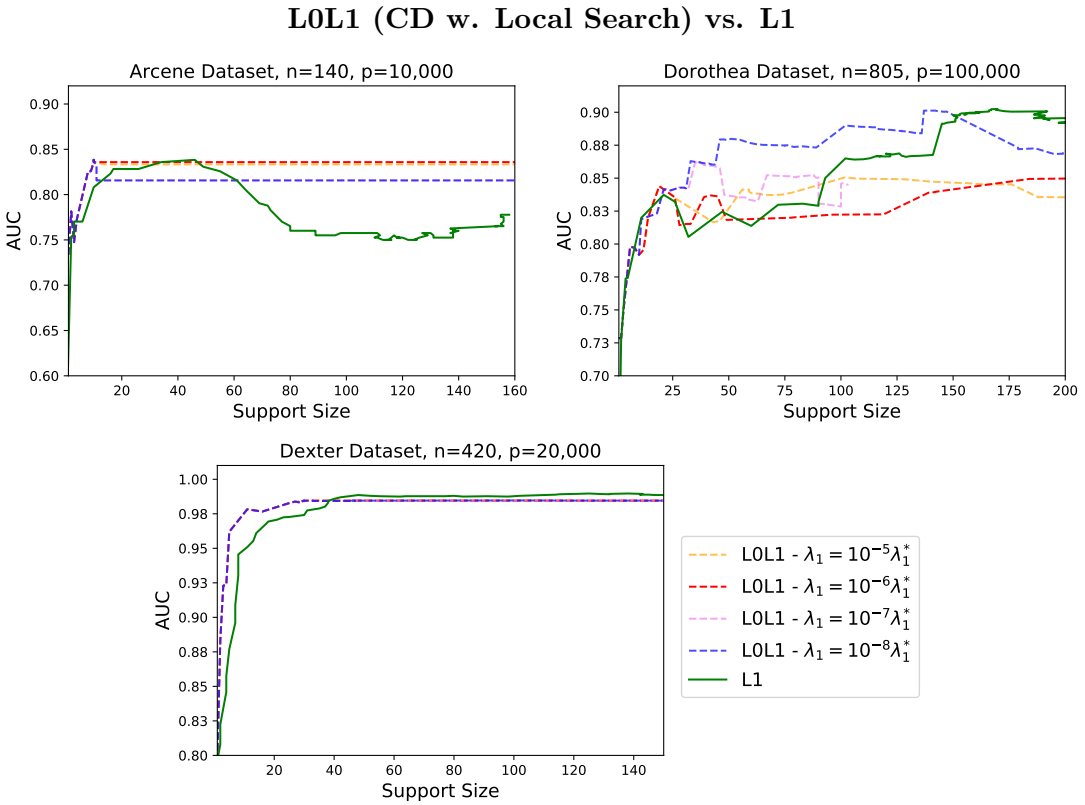

Figure 6: Plots of the AUC versus the support size for the Arcene, Dorothea, and Dexter data sets. The green curves correspond to logistic regression with ℓ_1 regularization. The other curves correspond to logistic regression with ℓ_0-ℓ_1 regularization using Algorithm 2 (with $m = 1$) for different values of λ_2 (see legend).

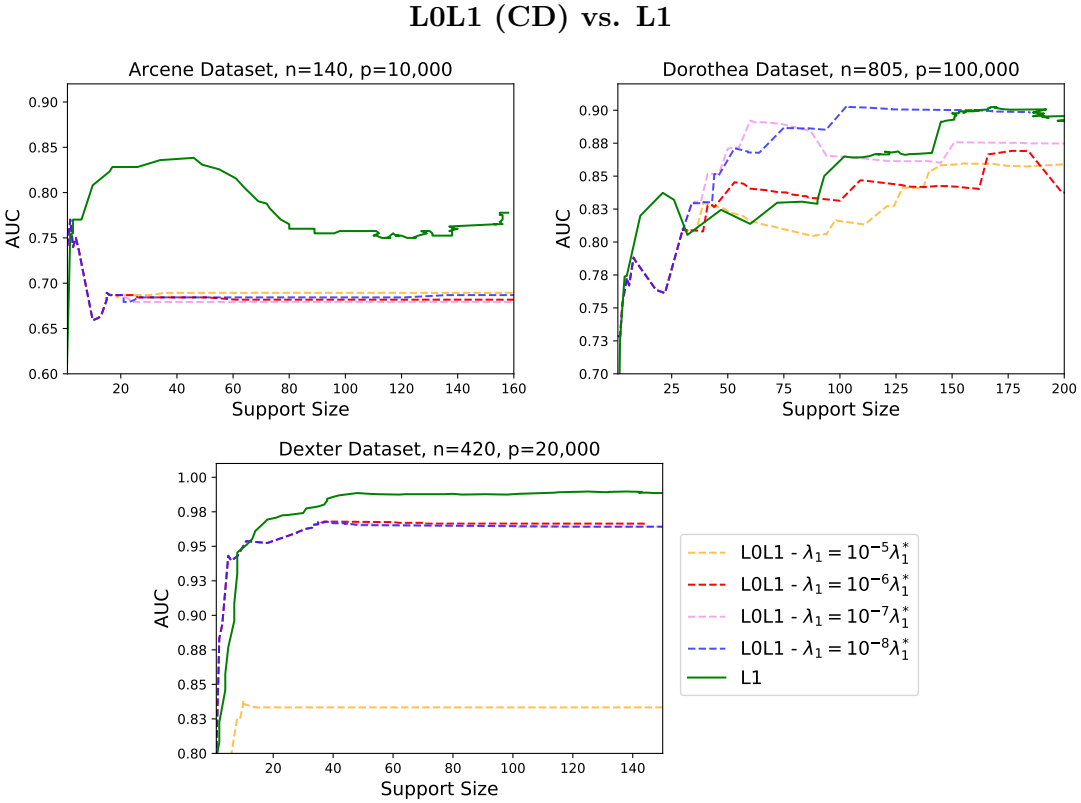

Figure 7: Plots of the AUC versus the support size for the Arcene, Dorothea, and Dexter data sets. The green curves correspond to logistic regression with ℓ_1 regularization. The other curves correspond to logistic regression with ℓ_0-ℓ_1 regularization using Algorithm 1 for different values of λ_2 (see legend).

References

Sohail Bahmani, Bhiksha Raj, and Petros T Boufounos. Greedy sparsity-constrained optimization. *Journal of Machine Learning Research*, 14(Mar):807–841, 2013.

Amir Beck and Yonina C. Eldar. Sparsity constrained nonlinear optimization: Optimality conditions and algorithms. *SIAM Journal on Optimization*, 23(3):1480–1509, 2013. doi: 10.1137/120869778. URL `https://doi.org/10.1137/120869778`.

Amir Beck and Luba Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013. doi: 10.1137/120887679. URL `http://dx.doi.org/10.1137/120887679`.

Alexandre Belloni and Victor Chernozhukov. l1-penalized quantile regression in high-dimensional sparse models. *The Annals of Statistics*, 39(1):82–130, 2011.

Pietro Belotti, Christian Kirches, Sven Leyffer, Jeff Linderoth, James Luedtke, and Ashutosh Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013.

D.P. Bertsekas. *Nonlinear Programming*. Athena scientific optimization and computation series. Athena Scientific, 2016. ISBN 9781886529052. URL `https://books.google.com/books?id=TwOujgEACAAJ`.

Dimitris Bertsimas and Angela King. Logistic regression: From art to science. *Statistical Science*, 32(3):367–384, 2017.

Dimitris Bertsimas, Angela King, and Rahul Mazumder. Best subset selection via a modern optimization lens. *The Annals of Statistics*, 44(2):813–852, 2016.

Dimitris Bertsimas, Jean Pauphilet, and Bart Van Parys. Sparse classification and phase transitions: A discrete optimization perspective. *arXiv preprint arXiv:1710.01352*, 2017.

Dimitris Bertsimas, Bart Van Parys, et al. Sparse high-dimensional regression: Exact scalable algorithms and phase transitions. *The Annals of Statistics*, 48(1):300–323, 2020.

Peter J Bickel, Ya'acov Ritov, and Alexandre B Tsybakov. Simultaneous analysis of lasso and dantzig selector. *The Annals of Statistics*, pages 1705–1732, 2009.

Thomas Blumensath and Mike E Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.

Patrick Breheny and Jian Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *The annals of applied statistics*, 5(1):232, 2011.

Peter Bühlmann and Sara Van De Geer. *Statistics for High-dimensional Data: Methods, Theory and Applications*. Springer Science & Business Media, 2011.

Florentina Bunea, Alexandre B Tsybakov, and Marten H Wegkamp. Aggregation for gaussian regression. *The Annals of Statistics*, 35(4):1674–1697, 2007.

Emmanuel Candes and Mark A Davenport. How well can we estimate a sparse vector? *Applied and Computational Harmonic Analysis*, 34(2):317–323, 2013.

Emmanuel J Candes and Terence Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, pages 2313–2351, 2007.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*. Springer series in statistics New York, NY, USA:, 2001.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010. URL `http://www.jstatsoft.org/v33/i01/`.

Pinghua Gong, Changshui Zhang, Zhaosong Lu, Jianhua Huang, and Jieping Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *International Conference on Machine Learning*, pages 37–45, 2013.

Eitan Greenshtein. Best subset selection, persistence in high-dimensional statistical learning and optimization under l1 constraint. *The Annals of Statistics*, 34(5):2367–2386, 2006.

Mert Gurbuzbalaban, Asuman Ozdaglar, Pablo A Parrilo, and Nuri Vanli. When cyclic coordinate descent outperforms randomized coordinate descent. In *Advances in Neural Information Processing Systems*, pages 6999–7007, 2017.

Isabelle Guyon, Steve Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems*, pages 545–552, 2005.

Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press, FL, 2015.

Trevor Hastie, Robert Tibshirani, and Ryan Tibshirani. Best subset, forward stepwise or lasso? analysis and recommendations based on extensive comparisons. *Statist. Sci.*, 35(4):579–592, 11 2020. doi: 10.1214/19-STS733. URL `https://doi.org/10.1214/19-STS733`.

Hussein Hazimeh and Rahul Mazumder. Fast best subset selection: Coordinate descent and local combinatorial optimization algorithms. *Operations Research*, 68(5):1517–1537, 2020. doi: 10.1287/opre.2019.1919. URL `https://doi.org/10.1287/opre.2019.1919`.

Hussein Hazimeh, Rahul Mazumder, and Ali Saab. Sparse regression at scale: Branch-and-bound rooted in first-order optimization. *arXiv preprint arXiv:2004.06152*, 2020.

Hussein Hazimeh, Rahul Mazumder, and Peter Radchenko. Grouped variable selection with discrete optimization: Computational and statistical perspectives. *arXiv preprint arXiv:2104.07084*, 2021.

Ja-Yong Koo, Yoonkyung Lee, Yuwon Kim, and Changyi Park. A Bahadur representation of the linear support vector machine. *Journal of Machine Learning Research*, 9(Jul): 1343–1368, 2008.

Jon Lee and Sven Leyffer. *Mixed Integer Nonlinear Programming*. Springer Publishing Company, Incorporated, 2011. ISBN 1461419263.

Huan Li and Zhouchen Lin. Accelerated proximal gradient methods for nonconvex programming. *Advances in Neural Information Processing Systems*, 28:379–387, 2015.

Zhaosong Lu. Iterative hard thresholding methods for l0 regularized convex cone programming. *Mathematical Programming*, 147(1):125–154, Oct 2014. ISSN 1436-4646. doi: 10.1007/s10107-013-0714-4. URL `https://doi.org/10.1007/s10107-013-0714-4`.

R. Mazumder, P. Radchenko, and A. Dedieu. Subset Selection with Shrinkage: Sparse Linear Modeling when the SNR is low. *ArXiv e-prints*, August 2017.

Rahul Mazumder, Jerome H. Friedman, and Trevor Hastie. Sparsenet: Coordinate descent with nonconvex penalties. *Journal of the American Statistical Association*, 106(495): 1125–1138, 2011. doi: 10.1198/jasa.2011.tm09738. URL `https://doi.org/10.1198/jasa.2011.tm09738`. PMID: 25580042.

Alan Miller. *Subset selection in regression*. CRC Press Washington, 2002.

Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

Sahand Negahban, Bin Yu, Martin J Wainwright, and Pradeep K Ravikumar. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. In *Advances in Neural Information Processing Systems*, pages 1348–1356, 2009.

Yu Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

Neal Parikh and Stephen Boyd. *Proximal Algorithms*. Now Foundations and Trends, 2014. ISBN 9781601987167. doi: 10.1561/2400000003. URL `http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=8187362`.

A. Patrascu and I. Necoara. Random coordinate descent methods for ℓ_0 regularized convex optimization. *IEEE Transactions on Automatic Control*, 60(7):1811–1824, July 2015. ISSN 0018-9286. doi: 10.1109/TAC.2015.2390551.

Bo Peng, Lan Wang, and Yichao Wu. An error bound for l1-norm support vector machine coefficients in ultra-high dimension. *Journal of Machine Learning Research*, 17:1–26, 2016.

Yaniv Plan and Roman Vershynin. Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. *IEEE Transactions on Information Theory*, 59(1):482–494, 2013.

Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Minimax rates of estimation for high-dimensional linear regression over l_q-balls. *IEEE transactions on information theory*, 57 (10):6976–6994, 2011.

Pradeep Ravikumar, Martin J Wainwright, and John D Lafferty. High-dimensional ising model selection using l1-regularized logistic regression. *The Annals of Statistics*, 38(3): 1287–1319, 2010.

Philippe Rigollet. 18.s997: High dimensional statistics. *Lecture Notes, Cambridge, MA, USA: MIT OpenCourseWare*, 2015.

Toshiki Sato, Yuichi Takano, Ryuhei Miyashiro, and Akiko Yoshise. Feature subset selection for logistic regression via mixed integer optimization. *Computational Optimization and Applications*, 64(3):865–880, 2016.

Shai Shalev-Shwartz and Tong Zhang. Proximal stochastic dual coordinate ascent. *arXiv preprint arXiv:1211.2717*, 2012.

Bernadetta Tarigan and Sara A Van De Geer. Classifiers of support vector machine type with l1 complexity regularization. *Bernoulli*, 12(6):1045–1076, 2006.

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

P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109(3):475–494, 2001. ISSN 1573-2878. doi: 10.1023/A:1017501703105. URL `http://dx.doi.org/10.1023/A:1017501703105`.

Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016.

Sara A Van de Geer. High-dimensional generalized linear models and the lasso. *The Annals of Statistics*, pages 614–645, 2008.

Rui Wang, Naihua Xiu, and Shenglong Zhou. Fast newton method for sparse logistic regression. *arXiv preprint arXiv:1901.02768*, 2019.

Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.

Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1): 3–34, 2015.

X. Yuan and Q. Liu. Newton-type greedy selection methods for ℓ_0 -constrained minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12): 2437–2450, 2017. doi: 10.1109/TPAMI.2017.2651813.

Cun-Hui Zhang. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.

Xiang Zhang, Yichao Wu, Lan Wang, and Runze Li. Variable selection for support vector machines in moderately high dimensions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(1):53–76, 2016.

Shenglong Zhou, Naihua Xiu, and Hou-Duo Qi. Global and quadratic convergence of newton hard-thresholding pursuit. *Journal of Machine Learning Research*, 22(12):1–45, 2021.