arXiv
Manuscript No.1502.04269

Supersparse Linear Integer Models for Optimized Medical Scoring Systems

Berk Ustun · Cynthia Rudin

Abstract Scoring systems are linear classification models that only require users to add, subtract and multiply a few small numbers in order to make a prediction. These models are in widespread use by the medical community, but are difficult to learn from data because they need to be accurate and sparse, have coprime integer coefficients, and satisfy multiple operational constraints. We present a new method for creating data-driven scoring systems called a Supersparse Linear Integer Model (SLIM). SLIM scoring systems are built by solving an integer program that directly encodes measures of accuracy (the 0–1 loss) and sparsity (the ℓ_0-seminorm) while restricting coefficients to coprime integers. SLIM can seamlessly incorporate a wide range of operational constraints related to accuracy and sparsity, and can produce highly tailored models without parameter tuning. We provide bounds on the testing and training accuracy of SLIM scoring systems, and present a new data reduction technique that can improve scalability by eliminating a portion of the training data beforehand. Our paper includes results from a collaboration with the Massachusetts General Hospital Sleep Laboratory, where SLIM was used to create a highly tailored scoring system for sleep apnea screening.

1 Introduction

Scoring systems are linear classification models that only require users to add, subtract and multiply a few small numbers in order to make a prediction. These models are used to assess the risk of numerous serious medical conditions since they allow physicians to make quick predictions, without extensive training, and without the use of a computer (see e.g., Knaus et al., 1991, Bone et al., 1992, Moreno et al., 2005). Many medical scoring systems that are currently in use were hand-crafted by physicians, whereby a panel of experts simply agreed on a model (see e.g., the CHADS_2 score of Gage et al., 2001). Some medical scoring systems are data-driven in the sense that they were created by rounding logistic regression coefficients (see e.g., the SAPS II score of Le Gall et al., 1993). Despite the widespread use of medical scoring systems in high-stakes applications, there has been little to no work that has focused on a direct method to learn these models from data.

Scoring systems are difficult to create using traditional machine learning methods because they need to be accurate, sparse, and use small coprime integer coefficients. This task is exceptionally challenging in

Berk Ustun
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
E-mail: ustunb@mit.edu

Cynthia Rudin
Sloan School of Management and CSAIL
Massachusetts Institute of Technology
E-mail: rudin@mit.edu

medical applications because models also need to satisfy explicit constraints on operational quantities such as the false positive rate or the number of features before they can be deployed. The sum of these requirements represent serious challenges for machine learning. Current methods for sparse linear classification such as the Lasso (Tibshirani, 1996) and Elastic Net (Zou and Hastie, 2005) control the accuracy and sparsity of models via convex surrogate functions to speed up computation, and require rounding to yield models with coprime integer coefficients. Approximations such as convex surrogate loss functions, ℓ_1-regularization, and rounding not only degrade predictive performance but make it difficult to address operational constraints imposed by physicians. To train a model that satisfies a hard constraint on the false positive rate, for instance, we must compute its value explicitly, which is impossible when we control accuracy by means of a surrogate loss function. In practice, traditional methods are therefore only able to address operational constraints through a parameter tuning process that involves high-dimensional grid search. As we show, this approach often fails to produce a model that satisfies operational constraints, let alone a model that is optimized for predictive accuracy.

In this paper, we present a new method to create data-driven scoring systems called a Supersparse Linear Integer Model (SLIM). SLIM is a integer programming problem that optimizes direct measures of accuracy (the 0–1 loss) and sparsity (the ℓ_0-seminorm) while restricting coefficients to a small set of coprime integers. In comparison to current methods for sparse linear classification, SLIM can produce scoring systems that are fully optimized for accuracy and sparsity, and that satisfy a wide range of complicated operational constraints without any parameter tuning.

The main contributions of our paper are as follows.

- We present a principled machine learning approach to learn scoring systems from data. This approach can produce tailored scoring systems that satisfy multiple operational constraints without any parameter tuning. Further, it has a unique advantage for imbalanced classification problems, where constraints on class-based accuracy can be explicitly enforced.
- We derive new bounds on the accuracy of discrete linear classification models. In particular, we present discretization bounds that guarantee that we will not lose training accuracy when the size of the coefficient set is sufficiently large. In addition, we present generalization bounds that relate the size of the coefficient set to a uniform guarantee on testing accuracy.
- We develop a novel data reduction technique that can improve the scalability of supervised classification algorithms by removing a portion of the training data beforehand. Further, we show how data reduction can be applied directly to SLIM.
- We present results from a collaboration with the Massachusetts General Hospital (MGH) Sleep Laboratory where SLIM was used to create a highly tailored scoring system for sleep apnea screening. Screening for sleep apnea is important: the condition is difficult to diagnose, has significant costs, and affects over 12 million people in the United States alone (Kapur, 2010).
- We provide a detailed experimental comparison between SLIM and eight popular classification methods on publicly available datasets. Our results suggest that SLIM can produce scoring systems that are accurate and sparse in a matter of minutes.
- We provide software to create SLIM scoring systems using MATLAB and the CPLEX API.

The remainder of our paper is structured as follows. In the rest of Section 1, we discuss related work. In Section 2, we introduce SLIM and discuss its special properties. In Section 2.2, we explain how SLIM can easily enforce operational constraints that are important for medical scoring systems to be used in practice. In Section 3, we present theoretical bounds on the accuracy of SLIM scoring systems and other discrete linear classification models. In Section 4, we present a data reduction technique to decrease the computation associated with SLIM and other supervised classification methods. In Section 5, we discuss a collaboration with the MGH Sleep Laboratory where we used SLIM to create a highly tailored scoring system for sleep apnea screening. In Section 6, we present experimental results to show that SLIM can create high-quality scoring systems in minutes. In Section 7, we present specialized extensions of SLIM.

1.1 Related Work

In what follows, we briefly discuss related work in medical scoring systems and linear classification.

Medical Scoring Systems

Medical scoring systems are sparse linear models with small coprime coefficients. Some popular examples include: SAPS I, II and III (Le Gall et al., 1993, Moreno et al., 2005) and APACHE I, II and III to assess ICU mortality risk (Knaus et al., 1981, 1985, 1991); CHADS_2 to assess the risk of stroke in patients with atrial fibrillation (Gage et al., 2001); and TIMI, to assess the risk of death and ischemic events (Antman et al., 2000).Most of the scoring systems that are in widespread use today were built without optimizing for predictive accuracy. In some cases, physicians built scoring systems by combining existing methods and heuristics. The SAPS II score, for instance, was built by rounding logistic regression coefficients as Le Gall et al. (1993) write, "*the general rule was to multiply the β for each range by 10 and round off to the nearest integer.*" This approach is at odds with the fact that rounding is known to produce suboptimal solutions in the field of integer programming. In other cases, scoring systems were hand-crafted by a panel of physicians, and not learned from data at all. This was the case for CHADS_2 as explained by Gage et al. (2001): "*We calculated CHADS_2, by adding 1 point each for each of the following – recent CHF, hypertension, age 75 years or older, and DM – and 2 points for a history of stroke or TIA.*" Methods that can learn tailored predictive models from data, such as SLIM, should eliminate the need for physicians to build scoring systems by hand.

To date, SLIM has already been used to create medical scoring systems for the purposes of diagnosing cognitive impairments using features derived from a clock-drawing test (see Souillard-Mandar et al., 2015), and for screening sleep apnea from electronic health records (see Ustun et al., 2015).

Sparse Linear Classification Models

In comparison to SLIM, current methods for sparse linear classification are designed to fit models with real coefficients, and need to be paired with a rounding procedure to create the same kinds of scoring systems used by physicians. In practice, rounding the coefficients of a linear model may significantly alter its accuracy and sparsity, and may result in a scoring system that violates operational constraints on these quantities. Current methods are also ill-suited to create scoring systems because they control accuracy and sparsity by means of convex surrogate functions to preserve scalability (see e.g., Tibshirani (1996), Efron et al. (2004)). As we show in Sections 5 and 6, surrogate functions result in a poor trade-off between accuracy and sparsity. Convex surrogate loss functions, for instance, produce models that are not robust to outliers (Nguyen and Sanner, 2013). Similarly, ℓ_1-regularization is only guaranteed to recover the correct sparse solution (i.e., the one that minimizes the ℓ_0-norm) under restrictive conditions that are rarely satisfied in practice (Zhao and Yu, 2007). In fact, ℓ_1-regularization may recover a solution with significantly less predictive accuracy relative to the correct sparse solution (see Lin et al. (2008) for a discussion).

SLIM is also related to a recent body of work on methods for discrete linear classification. Specifically, Chevaleyre et al. (2013) consider training linear classifiers with binary coefficients by rounding the coefficients of linear classifiers. In addition, Carrizosa et al. (2016) consider training linear classifiers with small integer coefficients using a MIP formulation. SLIM can reproduce both of these models. The converse, however, is not true because the methods of Chevaleyre et al. (2013) and Carrizosa et al. (2016): (i) optimize the hinge loss as opposed to the 0–1 loss; and (ii) do not include a mechanism to control sparsity. These differences may result in better scalability compared to SLIM. However, they also prevent these methods to create scoring systems that are sparse, that satisfy operational constraints on accuracy and/or sparsity, and that can be trained without parameter tuning. In addition to these differences, we note that the discretization bounds and generalization bounds in Section 3 are a novel contribution to this body of work and applicable to all linear models with discrete coefficients.

2 Methodology

We start with a dataset of N i.i.d. training examples $\mathcal{D}_N = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ where $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^{P+1}$ denotes a vector of features $[1, x_{i,1}, \ldots, x_{i,P}]^T$ and $y_i \in \mathcal{Y} = \{-1, 1\}$ denotes a class label. We consider linear models of the form $\hat{y} = \text{sign}(\boldsymbol{\lambda}^T \boldsymbol{x})$, where $\boldsymbol{\lambda} = [\lambda_0, \lambda_1, \ldots, \lambda_P]^T$ represents a vector of coefficients and λ_0 represents an intercept term. We learn the coefficients by solving an optimization problem of the form:

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \quad & \text{Loss}\left(\boldsymbol{\lambda}; \mathcal{D}_N\right) + C \cdot \Phi(\boldsymbol{\lambda}) \\ \text{s.t.} \quad & \boldsymbol{\lambda} \in \mathcal{L}. \end{aligned} \tag{1}$$

Here: the *loss function* $\text{Loss}\left(\boldsymbol{\lambda}; \mathcal{D}_N\right) : \mathbb{R}^{P+1} \times (\mathcal{X} \times \mathcal{Y})^N \to \mathbb{R}$ penalizes misclassifications; the *coefficient penalty* $\Phi(\boldsymbol{\lambda}) : \mathbb{R}^{P+1} \to \mathbb{R}$ induces soft qualities that are desirable but may be sacrificed for greater accuracy; the *coefficient set* $\mathcal{L}$ encodes hard qualities must be satisfied; and the *trade-off parameter* C controls the balance between accuracy and soft qualities. We assume: (i) the coefficient set contains the null vector, $\mathbf{0} \in \mathcal{L}$; (ii) the penalty is additively separable, $\Phi(\boldsymbol{\lambda}) = \sum_{j=0}^P \Phi_j(\lambda_j)$; (iii) the intercept is never penalized, $\Phi_0(\lambda_0) = 0$.

A *Supersparse Linear Integer Model* (SLIM) is a special case of the optimization in (1):

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \quad & \frac{1}{N} \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + C_0 \left\|\boldsymbol{\lambda}\right\|_0 + \epsilon \left\|\boldsymbol{\lambda}\right\|_1 \\ \text{s.t.} \quad & \boldsymbol{\lambda} \in \mathcal{L}. \end{aligned} \tag{2}$$

SLIM directly optimizes accuracy and sparsity by minimizing the 0–1 loss $\frac{1}{N}\sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right]$ and ℓ_0-norm $\left\|\boldsymbol{\lambda}\right\|_0 := \sum_{j=1}^P \mathbb{1}\left[\lambda_j \neq 0\right]$ respectively. The constraints usually restrict coefficients to a finite set of discrete values such as $\mathcal{L} = \{-10, \ldots, 10\}^{P+1}$, and may include additional operational constraints such as $\left\|\boldsymbol{\lambda}\right\|_0 \leq 10$. SLIM includes a *tiny* ℓ_1-penalty $\epsilon \left\|\boldsymbol{\lambda}\right\|_1$ in the objective for the sole purpose of restricting coefficients to coprime values.[1] To be clear, the ℓ_1-penalty parameter ϵ is always set to a value that is small enough to avoid ℓ_1-regularization (that is, ϵ is small enough to guarantee that SLIM never sacrifices accuracy or sparsity to attain a smaller ℓ_1-penalty).

SLIM is designed to produce scoring systems that attain a pareto-optimal trade-off between accuracy and sparsity: when we minimize 0–1 loss and the ℓ_0-penalty, we only sacrifice classification accuracy to attain higher sparsity, and vice versa. Minimizing the 0–1 loss produces scoring systems that are completely robust to outliers and attain the best learning-theoretic guarantee on predictive accuracy (see e.g. Brooks, 2011, Nguyen and Sanner, 2013). Similarly, controlling for sparsity via ℓ_0-regularization prevents the additional loss in accuracy due to ℓ_1-regularization (see Lin et al., 2008, for a discussion). In addition to these performance benefits, minimizing an approximation-free object function over a finite set of discrete coefficients means that the free parameters in SLIM's object have special properties.

Remark 1 If $\epsilon < \frac{\min\left(1/N, C_0\right)}{\max_{\boldsymbol{\lambda} \in \mathcal{L}} \left\|\boldsymbol{\lambda}\right\|_1}$ and $\mathcal{L}$ is a finite subset of $\mathbb{Z}^{P+1}$ then the optimization of (2) will produce a scoring system with coprime coefficients without affecting accuracy or sparsity:

$$\operatorname*{argmin}_{\boldsymbol{\lambda} \in \mathcal{L}} \frac{1}{N} \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + C_0 \left\|\boldsymbol{\lambda}\right\|_0 + \epsilon \left\|\boldsymbol{\lambda}\right\|_1 \subseteq \operatorname*{argmin}_{\boldsymbol{\lambda} \in \mathcal{L}} \frac{1}{N} \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + C_0 \left\|\boldsymbol{\lambda}\right\|_0$$

$$\text{and } \gcd(\{\lambda_j^*\}_{j=0}^P) = 1 \text{ for all } \boldsymbol{\lambda}^* \in \operatorname*{argmin}_{\boldsymbol{\lambda} \in \mathcal{L}} \frac{1}{N} \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + C_0 \left\|\boldsymbol{\lambda}\right\|_0 + \epsilon \left\|\boldsymbol{\lambda}\right\|_1 .$$

[1] To illustrate the use of the ℓ_1-penalty, consider a classifier such as $\hat{y} = \text{sign}\left(x_1 + x_2\right)$. If the objective in (2) only minimized the 0–1 loss and an ℓ_0-penalty, then $\hat{y} = \text{sign}\left(2x_1 + 2x_2\right)$ would have the same objective value as $\hat{y} = \text{sign}\left(x_1 + x_2\right)$ because it makes the same predictions and has the same number of non-zero coefficients. Since coefficients are restricted to a finite discrete set, we add a *tiny* ℓ_1-penalty in the objective of (2) so that SLIM chooses the classifier with the smallest (i.e. coprime) coefficients, $\hat{y} = \text{sign}\left(x_1 + x_2\right)$.

Remark 2 The trade-off parameter C_0 represents the maximum accuracy that SLIM will sacrifice to remove a feature from the optimal scoring system.

Remark 3 If $C_0 < \frac{1}{NP}$ and $\epsilon < \frac{\min(1/N, C_0)}{\max_{\boldsymbol{\lambda}\in\mathcal{L}}\|\boldsymbol{\lambda}\|_1} = \frac{C_0}{\max_{\boldsymbol{\lambda}\in\mathcal{L}}\|\boldsymbol{\lambda}\|_1}$ then the optimization of (2) will produce a scoring system with coefficients $\boldsymbol{\lambda} \in \mathcal{L}$ with the highest possible training accuracy:

$$\operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathcal{L}} \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right] + C_0\|\boldsymbol{\lambda}\|_0 + \epsilon\|\boldsymbol{\lambda}\|_1 \subseteq \operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathcal{L}} \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right].$$

Remark 4 If $C_0 > 1 - \frac{1}{N}$ and $\epsilon < \frac{\min(1/N, C_0)}{\max_{\boldsymbol{\lambda}\in\mathcal{L}}\|\boldsymbol{\lambda}\|_1} = \frac{1/N}{\max_{\boldsymbol{\lambda}\in\mathcal{L}}\|\boldsymbol{\lambda}\|_1}$ then the optimization of (2) will produce a scoring system with coefficients $\boldsymbol{\lambda} \in \mathcal{L}$ with the highest possible sparsity:

$$\operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathcal{L}} \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right] + C_0\|\boldsymbol{\lambda}\|_0 + \epsilon\|\boldsymbol{\lambda}\|_1 \subseteq \operatorname*{argmin}_{\boldsymbol{\lambda}\in\mathcal{L}} C_0\|\boldsymbol{\lambda}\|_0.$$

Note that these properties are only possible using the formulation in (2). In particular, Remarks 2-4 require that we control accuracy using the 0–1 loss and control sparsity using an ℓ_0-penalty, and Remark 1 requires that we restrict coefficients to a finite discrete set.

2.1 SLIM IP Formulation

We train SLIM scoring systems using the following IP formulation:

$$\min_{\boldsymbol{\lambda},\psi,\boldsymbol{\Phi},\alpha,\beta} \frac{1}{N}\sum_{i=1}^{N}\psi_i + \sum_{j=1}^{P}\Phi_j$$

$$\begin{aligned}
\text{s.t.} \quad & M_i\psi_i \geq \gamma - \sum_{j=0}^{P} y_i\lambda_j x_{i,j} && i = 1,...,N && \textit{0–1 loss} && \text{(3a)}\\
& \Phi_j = C_0\alpha_j + \epsilon\beta_j && j = 1,...,P && \textit{int. penalty} && \text{(3b)}\\
& -\Lambda_j\alpha_j \leq \lambda_j \leq \Lambda_j\alpha_j && j = 1,...,P && \ell_0\textit{-norm} && \text{(3c)}\\
& -\beta_j \leq \lambda_j \leq \beta_j && j = 1,...,P && \ell_1\textit{-norm} && \text{(3d)}\\
& \lambda_j \in \mathcal{L}_j && j = 0,...,P && \textit{coefficient set} &&\\
& \psi_i \in \{0,1\} && i = 1,...,N && \textit{loss variables} &&\\
& \Phi_j \in \mathbb{R}_+ && j = 1,...,P && \textit{penalty variables} &&\\
& \alpha_j \in \{0,1\} && j = 1,...,P && \ell_0\textit{ variables} &&\\
& \beta_j \in \mathbb{R}_+ && j = 1,...,P && \ell_1\textit{ variables} &&
\end{aligned}$$

Here, the constraints in (3a) set the loss variables $\psi_i = \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right]$ to 1 if a linear classifier with coefficients $\boldsymbol{\lambda}$ misclassifies example i. This is a Big-M constraint for the 0–1 loss that depends on scalar parameters γ and M_i (see e.g., Rubin, 2009). The value of M_i represents the maximum score when example i is misclassified, and can be set as $M_i = \max_{\boldsymbol{\lambda}\in\mathcal{L}}(\gamma - y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i)$ which is easy to compute since $\mathcal{L}$ is finite. The value of γ represents the "margin" and should be set as a lower bound on $y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i$. When the features are binary, γ can be set to any value between 0 and 1. In other cases, the lower bound is difficult to calculate exactly so we set $\gamma = 0.1$, which makes an implicit assumption on the values of the features. The constraints in (3b) set the total penalty for each coefficient to $\Phi_j = C_0\alpha_j + \epsilon\beta_j$, where $\alpha_j := \mathbb{1}[\lambda_j \neq 0]$ is defined by Big-M constraints in (3c), and $\beta_j := |\lambda_j|$ is defined by the constraints in (3d). We denote the largest absolute value of each coefficient as $\Lambda_j := \max_{\lambda_j\in\mathcal{L}_j}|\lambda_j|$.

Restricting coefficients to a finite set results in significant practical benefits for the SLIM IP formulation, especially in comparison to other IP formulations that minimize the 0–1-loss and/or penalize the ℓ_0-norm.

Many IP formulations compute the 0–1 loss and ℓ_0-norm by means of Big-M constraints that use require users to specify Big-M constants (see e.g., Wolsey, 1998). Restricting the coefficients to a finite set allows us to bound Big-M constants in the SLIM IP formulation. Specifically, the Big-M constant for computing the 0–1 loss in constraints (3a) is bounded as $M_i \leq \max_{\boldsymbol{\lambda} \in \mathcal{L}}(\gamma - y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i)$ and the Big-M constant used to compute the ℓ_0-norm in constraints (3c) is bounded as $\Lambda_j \leq \max_{\lambda_j \in \mathcal{L}_j} |\lambda_j|$ (compare with Brooks (2011), Guan et al. (2009) where the same parameters have to be approximated by a "sufficiently large" constants). Bounding these constants lead to a tighter LP relaxation, which narrows the integrality gap, and improves the ability of commercial IP solvers to quickly obtain a proof of optimality.

2.2 Operational Constraints

SLIM provides users with an unprecedented amount of flexibility over their models by allowing them to directly encode a wide range of operational constraints into its IP formulation. In what follows, we provide a few examples to illustrate this process. We note that these techniques are possible because: (i) the variables used to encode the 0–1 loss and ℓ_0-penalty in the SLIM IP formulation can also encode operational constraints related to accuracy and sparsity; (i) the free parameters in the SLIM objective can be set without tuning (see Remarks 2–4).

Loss Constraints for Imbalanced Data

The majority of classification problems in the medical domain are imbalanced. Handling imbalanced data is incredibly difficult for most classification methods since maximizing classification accuracy often produces a trivial model (i.e., if the probability of heart attack is 1%, a model that never predicts a heart attack is still 99% accurate). SLIM has a unique advantage on such problems as it not only avoid producing a trivial model, but can produce a model at any user-specified point on the ROC curve without parameter tuning. That is, when physicians specify hard constraints on sensitivity (or specificity), we can encode these as *loss constraints* into the IP formulation, and solve a single IP to obtain the least specific (or most sensitive) model. To train the most sensitive scoring system with a maximum error of $\gamma \in [0, 1]$ on negatively-labeled examples we solve an IP with the form:

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \quad & \frac{W^+}{N} \sum_{i \in \mathcal{I}^+} \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + \frac{W^-}{N} \sum_{i \in \mathcal{I}^-} \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + C_0 \left\|\boldsymbol{\lambda}\right\|_0 + \epsilon \left\|\boldsymbol{\lambda}\right\|_1 \\ \text{s.t.} \quad & \frac{1}{N^-} \sum_{i \in \mathcal{I}^-} \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \geq 0\right] \leq \gamma \\ & \boldsymbol{\lambda} \in \mathcal{L}. \end{aligned} \tag{4}$$

This formulation optimizes a *weighted* 0–1 loss function where W^+ and W^- are user-defined weights that control the accuracy on the N^+ positive examples from the set $\mathcal{I}^+ = \{i : y_i = +1\}$, and N^- negative examples from the set $\mathcal{I}^- = \{i : y_i = -1\}$, respectively. Assuming that $W^+ + W^- = 1$, we set $W^+ > \frac{N^-}{1+N^-}$ so that SLIM weighs the accuracy on each positive example as much as all of the negative examples. In a typical setting, this would return a scoring system that classifies all positive examples correctly at the expense of misclassify all of the negative examples in order to classify an additional positive example correctly. In this case, however, the loss constraint (4) explicitly limits the error on negative examples to γ. Thus, SLIM returns a scoring system that attains the highest sensitivity among models with a maximum error of γ on negative examples.

Feature-Based Constraints for Input Variables

SLIM provides fine-grained control over the composition of input variables in a scoring system by formulating feature-based constraints. Specifically, we can use the indicator variables that encode the ℓ_0-norm $\alpha_j := \mathbb{1}[\lambda_j \neq 0]$ to formulate many logical constraint between features such as "either-or" conditions and "if-then" conditions (see (Wolsey, 1998) for an overview). This presents a practical alternative to create classification models that obey structured sparsity constraints (Jenatton et al., 2011) or hierarchical constraints (Bien et al., 2013).

The indicator variables α_j can be used to limit the number of input variables to at most Θ by adding the constraint, $\sum_{j=1}^{P} \alpha_j \leq \Theta$. More complicated feature-based constraints include "if-then" constraints to ensure that a scoring system will only include *hypertension* and *heart_attack* if it also includes *stroke*: $\alpha_{heart\_attack} + \alpha_{hypertension} \leq 2\alpha_{stroke}$, or hierarchical constraints to ensure that an input variable in the leaves can only be used when all features above it in the hierarchy are also used: $\alpha_{leaf} \leq \alpha_{node}$ for all nodes above the leaf.

2.3 Feature-Based Preferences

Physicians often have soft preferences between different input variables. SLIM allows practitioners to encode these preferences by specifying a distinct trade-off parameter for each coefficient $C_{0,j}$.

Explicitly, when our model should use feature j instead of feature k, we set $C_{0,k} = C_{0,j} + \delta$, where $\delta > 0$ represents the maximum additional training accuracy that we are willing to sacrifice in order to use feature j instead of feature k. Thus, setting $C_{0,k} = C_{0,j} + 0.02$ would ensure that we would only be willing to use feature k instead of feature j if it yields an additional 2% gain in training accuracy over feature k.

This approach can also be used to handle problems with missing data. Consider training a model where feature j contains $M < N$ missing points. Instead of dropping these points, we can impute the values of the M missing examples, and adjust the trade-off parameter $C_{0,j}$ so that our model only uses feature j if it yields an additional gain in accuracy of more than M examples:

$$C_{0,j} = C_0 + \frac{M}{N}.$$

The adjustment factor is chosen so that: if $M = 0$ then $C_{0,j} = C_0$ and if $M = N$ then $C_{0,j} = 1$ and the coefficient is dropped entirely (see Remark 4). This ensures that features with lots of imputed values are more heavily penalized than features with fewer imputed values.

3 Bounds on Training and Testing Accuracy

In this section, we present bounds on the training and testing accuracy of SLIM scoring systems.

3.1 Discretization Bounds on Training Accuracy

Our first result shows that we can always craft a finite discrete set of coefficients $\mathcal{L}$ so that the training accuracy of a linear classifier with discrete coefficients $\boldsymbol{\lambda} \in \mathcal{L}$ (e.g. SLIM) is no worse than the training accuracy of a baseline linear classifier with real-valued coefficients $\boldsymbol{\rho} \in \mathbb{R}^P$ (e.g. SVM).

Theorem 1 (Minimum Margin Resolution Bound) *Let $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_P]^T \in \mathbb{R}^P$ denote the coefficients of a baseline linear classifier trained using data $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$. Let $X_{\max} = \max_i \|\boldsymbol{x}_i\|_2$ and $\gamma_{\min} = \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2}$ denote the largest magnitude and minimum margin achieved by any training example, respectively.*

Consider training a linear classifier with coefficients $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_P]^T$ from the set $\mathcal{L} = \{-\Lambda, \ldots, \Lambda\}^P$. If we choose a resolution parameter Λ such that:

$$\Lambda > \frac{X_{\max}\sqrt{P}}{2\gamma_{\min}}, \tag{5}$$

then there exists $\boldsymbol{\lambda} \in \mathcal{L}$ such that the 0–1 loss of $\boldsymbol{\lambda}$ is less than or equal to the 0–1 loss of $\boldsymbol{\rho}$:

$$\sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] \leq \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\rho}^T \boldsymbol{x}_i \leq 0\right].$$

Proof See Appendix A.

The proof of Theorem 1 uses a rounding procedure to choose a resolution parameter Λ so that the coefficient set $\mathcal{L}$ contains a classifier with discrete coefficients $\boldsymbol{\lambda}$ that attains the same the 0–1 loss as the baseline classifier with real coefficients $\boldsymbol{\rho}$. If the baseline classifier $\boldsymbol{\rho}$ is obtained by minimizing a convex surrogate loss, then the optimal SLIM classifier trained with the coefficient set from Theorem 1 may attain a lower 0–1 loss than $\mathbb{1}\left[y_i \boldsymbol{\rho}^T \boldsymbol{x}_i \leq 0\right]$ because SLIM directly minimizes the 0–1 loss.

The next corollary yields additional bounds on the training accuracy by considering progressively larger values of the margin. These bounds can be used to relate the resolution parameter Λ to a worst-case guarantee on training accuracy.

Corollary 1 (k^{th} Margin Resolution Bound) *Let $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_P]^T \in \mathbb{R}^P$ denote the coefficients of a linear classifier trained with data $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$. Let $\gamma_{(k)}$ denote the value of the k^{th} smallest margin, $\mathcal{I}_{(k)}$ denote the set of training examples with $\frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2} \leq \gamma_{(k)}$, and $X_{(k)} = \max_{i \notin \mathcal{I}_{(k)}} \|\boldsymbol{x}_i\|_2$ denote the largest magnitude of any training example $\boldsymbol{x}_i \in \mathcal{D}_N$ for $i \notin \mathcal{I}_{(k)}$.*

Consider training a linear classifier with coefficients $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_P]^T$ from the set $\mathcal{L} = \{-\Lambda, \ldots, \Lambda\}^P$. If we choose a resolution parameter Λ such that:

$$\Lambda > \frac{X_{(k)}\sqrt{P}}{2\gamma_{(k)}},$$

then there exists $\boldsymbol{\lambda} \in \mathcal{L}$ such that the 0–1 loss of $\boldsymbol{\lambda}$ and the 0–1 loss of $\boldsymbol{\rho}$ differ by at most $k-1$:

$$\sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] - \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\rho}^T \boldsymbol{x}_i \leq 0\right] \leq k - 1.$$

Proof The proof follows by applying Theorem 1 to the reduced dataset $\mathcal{D}_N \backslash \mathcal{I}_{(k)}$.

We have now shown that good discretized solutions exist and can be constructed easily. This motivates that optimal discretized solutions, which by definition are better than rounded solutions, will also be good relative to the best non-discretized solution.

3.2 Generalization Bounds on Testing Accuracy

According to the principle of structural risk minimization (Vapnik, 1998), fitting a classifier from a simpler class of models may lead to an improved guarantee on predictive accuracy. Consider training a classifier $f : \mathcal{X} \to \mathcal{Y}$ with data $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$, where $\boldsymbol{x}_i \in \mathcal{X} \subseteq \mathbb{R}^P$ and $y_i \in \mathcal{Y} = \{-1, 1\}$. In what follows, we provide uniform generalization guarantees on the predictive accuracy of all functions, $f \in \mathcal{F}$. These guarantees bound the true risk $R^{\text{true}}(f) = \mathbb{E}_{\mathcal{X},\mathcal{Y}} \mathbb{1}[f(\boldsymbol{x}) \neq y]$ by the empirical risk $R^{\text{emp}}(f) = \frac{1}{N} \sum_{i=1}^N \mathbb{1}[f(\boldsymbol{x}_i) \neq y_i]$ and other quantities important to the learning process.

Theorem 2 (Occam's Razor Bound for Discrete Linear Classifiers) *Let $\mathcal{F}$ denote the set of linear classifiers with coefficients $\boldsymbol{\lambda} \in \mathcal{L}$:*

$$\mathcal{F} = \left\{ f : \mathcal{X} \to \mathcal{Y} \;\middle|\; f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{\lambda}^T \boldsymbol{x}\right) \text{ and } \boldsymbol{\lambda} \in \mathcal{L} \right\}.$$

For every $\delta > 0$, with probability at least $1 - \delta$, every classifier $f \in \mathcal{F}$ obeys:

$$R^{\text{true}}(f) \leq R^{\text{emp}}(f) + \sqrt{\frac{\log(|\mathcal{L}|) - \log(\delta)}{2N}}.$$

A proof of Theorem 2 can be found in Section 3.4 of Bousquet et al. 2004. The result that more restrictive hypothesis spaces can lead to better generalization provides motivation for using discrete models without necessarily expecting a loss in predictive accuracy. The bound indicates that we include more coefficients in the set $\mathcal{L}$ as the amount of data N increases.

In Theorem 3, we improve the generalization bound from Theorem 2 by excluding models that are provably suboptimal from the hypothesis space. Here, we exploit the fact that we can bound the number of non-zero coefficients in a SLIM scoring system based on the value of C_0.

Theorem 3 (Generalization of Sparse Discrete Linear Classifiers) *Let $\mathcal{F}$ denote the set of linear classifiers with coefficients $\boldsymbol{\lambda}$ from a finite set $\mathcal{L}$ such that:*

$$\mathcal{F} = \left\{ f : \mathcal{X} \to \mathcal{Y} \;\middle|\; f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{\lambda}^T \boldsymbol{x}\right) \right\}$$

$$\boldsymbol{\lambda} \in \underset{\lambda \in \mathcal{L}}{\text{argmin}} \frac{1}{N} \sum_{i=1}^N \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] + C_0 \|\boldsymbol{\lambda}\|_0$$

For every $\delta > 0$, with probability at least $1 - \delta$, every classifier $f \in \mathcal{F}$ obeys:

$$R^{\text{true}}(f) \leq R^{\text{emp}}(f) + \sqrt{\frac{\log(|\mathcal{H}_{P,C_0}|) - \log(\delta)}{2N}}.$$

where

$$\mathcal{H}_{P,C_0} = \left\{ \boldsymbol{\lambda} \in \mathcal{L} \;\middle|\; \|\boldsymbol{\lambda}\|_0 \leq \left\lfloor \frac{1}{C_0} \right\rfloor \right\}.$$

Proof See Appendix A.

This theorem relates the trade-off parameter C_0 in the SLIM objective to the generalization of SLIM scoring systems. It indicates that increasing the value of the C_0 parameter will produce a model with better generalization properties.

In Theorem 4, we produce a better generalization bound by exploiting the fact that SLIM scoring systems use coprime integer coefficients (see Remark 1). In particular, we express the generalization bound from Theorem 2 using the P-dimensional Farey points of level Λ (see Marklof, 2012, for a definition).

Theorem 4 (Generalization of Discrete Linear Classifiers with Coprime Coefficients) *Let $\mathcal{F}$ denote the set of linear classifiers with coprime integer coefficients, $\boldsymbol{\lambda}$, bounded by Λ:*

$$\mathcal{F} = \left\{ f : \mathcal{X} \to \mathcal{Y} \;\middle|\; f(\boldsymbol{x}) = \text{sign}\left(\boldsymbol{\lambda}^T \boldsymbol{x}\right) \text{ and } \boldsymbol{\lambda} \in \mathcal{L} \right\},$$
$$\mathcal{L} = \left\{ \boldsymbol{\lambda} \in \hat{\mathbb{Z}}^P \;\middle|\; |\lambda_j| \leq \Lambda \text{ for } j = 1, \ldots, P \right\},$$
$$\hat{\mathbb{Z}}^P = \left\{ \boldsymbol{z} \in \mathbb{Z}^P \;\middle|\; \text{gcd}(\boldsymbol{z}) = 1 \right\}.$$

For every $\delta > 0$, with probability at least $1 - \delta$, every classifier $f \in \mathcal{F}$ obeys:

$$R^{\text{true}}(f) \leq R^{\text{emp}}(f) + \sqrt{\frac{\log(|\mathcal{C}_{P,\Lambda}|) - \log(\delta)}{2N}},$$

where $\mathcal{C}_{P,\Lambda}$ denotes the set of Farey points of level Λ:

$$\mathcal{C}_{P,\Lambda} = \left\{ \frac{\boldsymbol{\lambda}}{q} \in [0,1)^P : (\boldsymbol{\lambda}, q) \in \hat{\mathbb{Z}}^{P+1} \text{ and } 1 \leq q \leq \Lambda \right\}.$$

The proof involves a counting argument over coprime integer vectors, using the definition of Farey points from number theory.

In Figure 1, we plot the relative density of coprime integer vectors bounded by Λ (i.e., $|\mathcal{C}_{P,\Lambda}|/(2\Lambda+1)^P$), and the relative improvement in the generalization bound due to the use of coprime coefficients. We see that the use of coprime coefficients can significantly reduce the number of classifiers based on the dimensionality of the data and the value of Λ. The corresponding improvement in the generalization bound may be significant when the data are high dimensional and Λ is small.

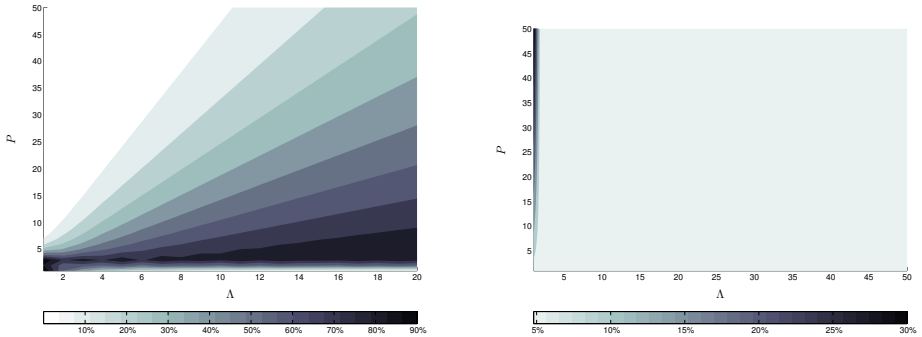

Fig. 1: Relative density of coprime integer vectors in $\mathbb{Z}^P$ (left), and the relative improvement in the generalization bound due to the use of coprime coefficients for $\delta = 0.01$ (right).

4 Data Reduction

Data reduction is a technique that can decrease the computation associated with training a supervised classification model by discarding redundant training data. This technique can be applied to any supervised classification method where the training procedure is carried out by solving an optimization problem. However, it is best suited for methods such as SLIM, where the underlying optimization problem may be difficult to solve for large instances. In this section, we first describe how data reduction works in a general setting, and then show how it can be applied to SLIM.

4.1 Data Reduction for Optimization-Based Supervised Classification

Consider training a classifier $f : \mathcal{X} \to \mathcal{Y}$ by solving a computationally challenging optimization problem,

$$\min_{f} \; Z(f; \mathcal{D}_N) \text{ s.t. } f \in \mathcal{F}. \tag{6}$$

We refer to the optimization problem in (6) as the *original problem*. Here, $\mathcal{F}$ represents the set of feasible classifiers and $Z : \mathcal{F} \times (\mathcal{X} \times \mathcal{Y})^N \to \mathbb{R}$ represents its objective function.

Data reduction aims to decrease the computation associated with solving the original problem by removing redundant examples from $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$ (i.e., data points that can be safely discarded without changing the optimal solution to (6)). The technique requires users to specify a *surrogate problem* that is considerably easier to solve. Given the initial training data $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$, and the surrogate problem, data reduction solves $N + 1$ variants of the surrogate problem to identify redundant examples. These examples are then removed from the initial training data to leave behind a subset of reduced training data $\mathcal{D}_M \subseteq \mathcal{D}_N$ that is guaranteed to yield the same optimal classifier as $\mathcal{D}_N$. Thus, the computational gain from data reduction comes from training a model with $\mathcal{D}_M$ (i.e., solving an instance of the original problem with $N - M$ fewer examples).

We provide an overview of data reduction in Algorithm 1. To explain how the algorithm works, let us denote the surrogate problem as:

$$\min_{f} \; \tilde{Z}(f; \mathcal{D}_N) \text{ s.t } f \in \tilde{\mathcal{F}}. \tag{7}$$

Here $\tilde{Z} : \tilde{\mathcal{F}} \times (\mathcal{X} \times \mathcal{Y})^N \to \mathbb{R}$ denotes the objective function of the surrogate problem, and $\tilde{\mathcal{F}}$ denotes its set of feasible classifiers. Data reduction can be used with any surrogate problem so long as the ε-level set of the surrogate problem contains all optimizers to the original problem. That is, we can use any feasible set $\tilde{\mathcal{F}}$ and any objective function $\tilde{Z}(.)$ as long as we can specify a value of ε such that

$$\tilde{Z}(f^*) \leq \tilde{Z}(\tilde{f}^*) + \varepsilon \qquad \forall f^* \in \mathcal{F}^* \text{ and } \tilde{f}^* \in \tilde{\mathcal{F}}^*. \tag{8}$$

Here, f^* denotes an optimal classifier to the original problem from the set $\mathcal{F}^* = \operatorname{argmin}_{f \in \mathcal{F}} Z(f)$, and $\tilde{f}^*$ denotes an optimal classifier to the surrogate problem from the set $\tilde{\mathcal{F}}^* = \operatorname{argmin}_{f \in \tilde{\mathcal{F}}} \tilde{Z}(f)$. The width of the the surrogate level set ε is related to the amount of data that will be removed. If ε is too large, the method will not remove very many examples and will be less helpful for reducing computation (see Figure 3).

In the first stage of data reduction, we solve the surrogate problem to: (i) compute the upper bound on the objective value of classifiers in the surrogate level set $\tilde{Z}(\tilde{f}^*) + \varepsilon$; and (ii) to identify a baseline label $\tilde{y}_i := \operatorname{sign}\left(\tilde{f}^*(\boldsymbol{x}_i)\right)$ for each example $i = 1, \ldots, N$. In the second stage of data reduction, we solve a variant of the surrogate problem for each example $i = 1, \ldots, N$. The i^{th} variant of the surrogate problem includes an additional constraint that forces example i to be classified as $-\tilde{y}_i$:

$$\min_{f} \; \tilde{Z}(f; \mathcal{D}_N) \text{ s.t } f \in \tilde{\mathcal{F}} \text{ and } \tilde{y}_i f(\boldsymbol{x}_i) < 0 \tag{9}$$

We denote the optimal classifier to the i^{th} variant as $\tilde{f}^*_{-i}$. If $\tilde{f}^*_{-i}$ lies outside of the surrogate level set (i.e., $\tilde{Z}(\tilde{f}^*_{-i}) > \tilde{Z}(\tilde{f}^*) + \varepsilon$) then no classifier in the surrogate level set will label point i as $-\tilde{y}_i$. In other words, all classifiers in the surrogate level set must label this point as $\tilde{y}_i$. Since the surrogate level set contains the optimal classifiers to the original problem by the assumption in (8), we can therefore remove example i from the reduced dataset $\mathcal{D}_M$ because we know that an optimal classifier to the original problem will label this point as $\tilde{y}_i$. We illustrate this situation in Figure 2.

In Theorem 5, we prove that we obtain the same set of optimal classifiers if we train a model with the initial data $\mathcal{D}_N$ or the reduced data $\mathcal{D}_M$. In Theorem 6, we provide sufficient conditions for a surrogate loss function to satisfy the level set condition in (8).

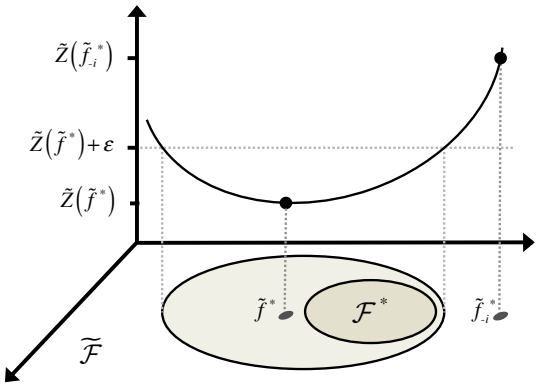

Fig. 2: We initialize data reduction with ε large enough so that $\tilde{Z}(f^*) < \tilde{Z}(\tilde{f}^*) + \varepsilon$ for all $f^* \in \mathcal{F}^*$ and all $\tilde{f}^* \in \tilde{\mathcal{F}}^*$. Here, f^* is the optimal classifer to the original problem from the set of optimal classifiers $\mathcal{F}^*$, and $\tilde{f}^*$ is the optimal classifier to the surrogate problem from the set of optimal classifiers $\tilde{\mathcal{F}}^*$. Data reduction fits a classifier $\tilde{f}_{-i}^*$ for each example in the initial training data $\mathcal{D}_N$ by solving a variant of the surrogate problem with an additional constraint that forces $\tilde{f}_{-i}^*$ to classify i in a different way than $\tilde{f}^*$. If $\tilde{Z}(\tilde{f}_{-i}^*) > \tilde{Z}(\tilde{f}^*) + \varepsilon$, then we know the predicted class of example i under f^* and can remove it from the reduced training data $\mathcal{D}_M$.

Algorithm 1 Data Reduction from $\mathcal{D}_N$ to $\mathcal{D}_M$

Input: initial training data, $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$
Input: surrogate problem, $\min \tilde{Z}(f; \mathcal{D}_N)$ s.t. $f \in \tilde{\mathcal{F}}$
Input: width of the surrogate level set, ε
  $\mathcal{D}_M \longleftarrow \emptyset$
  $\tilde{f}^* \longleftarrow \operatorname{argmin}_f \tilde{Z}(f; \mathcal{D}_N)$
  for $i = 1, \ldots, N$ **do**
    $\tilde{y}_i \longleftarrow \operatorname{sign}\left(\tilde{f}^*(\boldsymbol{x}_i)\right)$
    $\tilde{f}_{-i}^* \longleftarrow \operatorname{argmin} \tilde{Z}(f; \mathcal{D}_N)$ s.t. $f \in \tilde{\mathcal{F}}$ and $\tilde{y}_i f(\boldsymbol{x}_i) < 0$
    if $\tilde{Z}(\tilde{f}_{-i}^*; \mathcal{D}_N) \leq \tilde{Z}(\tilde{f}^*; \mathcal{D}_N) + \varepsilon$ **then**
      $\mathcal{D}_M \longleftarrow \mathcal{D}_M \cup (\boldsymbol{x}_i, y_i)$
    end if
  end for
Output: $\mathcal{D}_M$, reduced training data

Theorem 5 (Equivalence of the Reduced Data) *Consider an optimization problem to train a classifier $f \in \mathcal{F}$ with data $\mathcal{D}_N$,*

$$\min_f \; Z(f; \mathcal{D}_N) \; s.t \; f \in \mathcal{F},$$

as well as a surrogate optimization problem to train a classifier $f \in \tilde{\mathcal{F}}$ with data $\mathcal{D}_N$,

$$\min_f \; \tilde{Z}(f; \mathcal{D}_N) \; s.t. \; f \in \tilde{\mathcal{F}}.$$

Let $f^* \in \mathcal{F}^* := \operatorname{argmin}_{f \in \mathcal{F}} Z(f; \mathcal{D}_N)$ and $\tilde{f} \in \tilde{\mathcal{F}}^* := \operatorname{argmin}_{f \in \tilde{\mathcal{F}}} \tilde{Z}(f; \mathcal{D}_N)$. If we choose a value of ε so that

$$\tilde{Z}(f^*; \mathcal{D}_N) \leq \tilde{Z}(\tilde{f}^*; \mathcal{D}_N) + \varepsilon \quad \forall f^* \in \mathcal{F}^* \text{ and } \tilde{f}^* \in \tilde{\mathcal{F}}^*, \tag{10}$$

then Algorithm 1 will output a reduced dataset $\mathcal{D}_M \subseteq \mathcal{D}_N$ such that

$$\operatorname*{argmin}_{f \in \mathcal{F}} Z(f; \mathcal{D}_N) = \operatorname*{argmin}_{f \in \mathcal{F}} Z(f; \mathcal{D}_M). \tag{11}$$

Proof See Appendix A.

Theorem 6 (Sufficient Conditions to Satisfy the Level Set Condition) *Consider an optimization problem where the objective minimizes the 0–1 loss function* $Z_{01} : \mathbb{R}^P \rightarrow \mathbb{R}$,

$$\min_{\boldsymbol{\lambda} \in \mathbb{R}^P} Z_{01}(\boldsymbol{\lambda}),$$

as well as a surrogate optimization problem where the objective minimizes a surrogate loss function $\psi : \mathbb{R}^P \rightarrow \mathbb{R}$,

$$\min_{\boldsymbol{\lambda} \in \mathbb{R}^P} Z_{\psi}(\boldsymbol{\lambda}).$$

If the surrogate loss function ψ *satisfies the following properties for all* $\boldsymbol{\lambda} \in \mathbb{R}^P$, $\boldsymbol{\lambda}_{01}^* \in \operatorname{argmin}_{\boldsymbol{\lambda} \in \mathbb{R}^P} Z_{01}(\boldsymbol{\lambda})$, *and* $\boldsymbol{\lambda}_{\psi}^* \in \operatorname{argmin}_{\boldsymbol{\lambda} \in \mathbb{R}^P} Z_{\psi}(\boldsymbol{\lambda})$*:*

I. *Upper bound on the 0–1 loss:* $Z_{01}(\boldsymbol{\lambda}) \leq Z_{\psi}(\boldsymbol{\lambda})$

II. *Lipschitz near* $\boldsymbol{\lambda}_{01}^*$*:* $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{\psi}^*\| < A \implies Z_{\psi}(\boldsymbol{\lambda}) - Z_{\psi}(\boldsymbol{\lambda}_{\psi}^*) < L\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{\psi}^*\|$

III. *Curvature near* $\boldsymbol{\lambda}_{\psi}^*$*:* $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_{\psi}^*\| > C_{\boldsymbol{\lambda}} \implies Z_{\psi}(\boldsymbol{\lambda}) - Z_{\psi}(\boldsymbol{\lambda}_{\psi}^*) > C_{\psi}$

IV. *Closeness of loss near* $\boldsymbol{\lambda}_{01}^*$*:* $|Z_{\psi}(\boldsymbol{\lambda}_{01}^*) - Z_{01}(\boldsymbol{\lambda}_{01}^*)| < \varepsilon$

then it will also satisfy a level-set condition required for data reduction,

$$Z_{\psi}(\boldsymbol{\lambda}_{01}^*) \leq Z_{\psi}(\boldsymbol{\lambda}_{\psi}^*) + \varepsilon \qquad \forall \boldsymbol{\lambda}_{01}^* \text{ and } \boldsymbol{\lambda}_{\psi}^*,$$

whenever $\varepsilon = LC_{\boldsymbol{\lambda}}$ *obeys* $C_{\psi} > 2\varepsilon$.

Proof See Appendix A.

4.2 Off-The-Shelf Data Reduction for SLIM

Data reduction can easily be applied to SLIM by using an off-the-shelf approach where we use the LP relaxation of the SLIM IP as the surrogate problem. The off-the-shelf approach may be used as a preliminary procedure before the training process, or as an iterative procedure that is called by the IP solver during the training process as feasible solutions are found.

When we use the LP relaxation to the SLIM IP as the surrogate problem, we can determine a suitable width for the surrogate level set ε by using a feasible solution to the SLIM IP. To see this, let us denote the SLIM IP as $\min_f Z(f)$ s.t. $f \in \mathcal{F}$, and denote its LP relaxation as $\min_f Z(f)$ s.t. $f \in \tilde{\mathcal{F}}$. In addition, let us denote the optimal solution to the SLIM IP as f^* and the optimal solution to the LP relaxation as $\tilde{f}^*$. Since $\mathcal{F} \subseteq \tilde{\mathcal{F}}$, we have that $Z(\tilde{f}^*) \leq Z(f^*)$. For any feasible solution to the SLIM IP $\hat{f} \in \mathcal{F}$, we also have that $Z(f^*) \leq Z(\hat{f})$. Combining both inequalities, we see that,

$$Z(\tilde{f}^*) \leq Z(f^*) \leq Z(\hat{f}).$$

Thus, we can satisfy the level set condition (8) using a feasible solution to the SLIM IP $\hat{f} \in \mathcal{F}$ by setting the width of the surrogate level set as

$$\varepsilon(\hat{f}) := Z(\hat{f}) - Z(\tilde{f}^*).$$

In Figure 3, we show much training data can be discarded using off-the-shelf data reduction when we train a SLIM scoring system on the `bankruptcy` dataset (see Table 4). Specifically, we plot the percentage of data removed by Algorithm 1 for values of $\varepsilon \in [\varepsilon_{\min}, \varepsilon_{\max}]$ where $\varepsilon_{\min}$ and $\varepsilon_{\max}$ represent the smallest

and largest widths of the surrogate level set that could be used in practice. In particular, $\varepsilon_{\min}$ is computed using the optimal solution to the IP as:

$$\varepsilon_{\min} := Z(f^*) - Z(\tilde{f}^*),$$

and $\varepsilon_{\max}$ is computed using a feasible solution to the IP that can be guessed without any computation (i.e., a linear classifier with coefficients $\boldsymbol{\lambda} = \mathbf{0}$):

$$\varepsilon_{\max} := Z(\mathbf{0}) - Z(\tilde{f}^*).$$

In this case, we can discard over 40% of the training data by using the trivial solution $\boldsymbol{\lambda} = 0$, and discard over 80% of the training data by using a higher quality feasible solution.

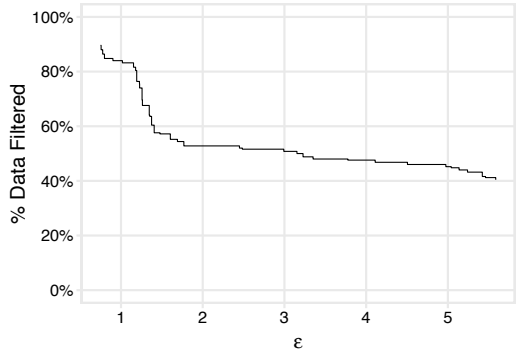

Fig. 3: Proportion of training data filtered as a function of the width of the level set, ε for the `bankruptcy` dataset. Here, the original problem is an instance of the SLIM IP with $C_0 = 0.01$ and $\mathcal{L} = \{-10, \ldots, 10\}^{P+1}$.

5 Application to Sleep Apnea Screening

In this section, we discuss a collaboration with the MGH Sleep Laboratory where we used SLIM to create a scoring system for sleep apnea screening (see also Ustun et al., 2015, for a far more detailed treatment). Our goal is to highlight the flexibility and performance of our approach on a real-world problem that requires a tailored prediction model.

5.1 Data and Operational Constraints

The dataset for this application contains $N = 1922$ records of patients and $P = 33$ binary features related to their health and sleep habits. Here, $y_i = +1$ if patient i has obstructive sleep apnea (OSA) and $y_i = -1$ otherwise. There is significant class imbalance as $\Pr(y_i = +1) = 76.9\%$.

To ensure that the scoring system we produced would be used and accepted by physicians, our collaborators specified three simple operational constraints:

1. *Limited FPR:* The model had to achieve the highest possible true positive rate (TPR) while maintaining a maximum false positive rate (FPR) of 20%. This would ensure that the model could diagnose as many cases of sleep apnea as possible but limit the number of faulty diagnoses.

2. *Limited Model Size:* The model had to be transparent and use at most 5 features. This would ensure that the model was could be explained and understood by other physicians in a short period of time.
3. *Sign Constraints:* The model had to obey established relationships between well-known risk factors and the incidence of sleep apnea (e.g. it could not suggest that a patient with hypertension had a lower risk of sleep apnea since hypertension is a positive risk factor for sleep apnea).

5.2 Training Setup and Model Selection

We trained a SLIM scoring system with integer coefficients between -10 and 10. We addressed all three operational constraints without parameter tuning or model selection, as follows:

- We added a loss constraint using the loss variables to limit the maximum FPR at 20%. We then set $W^+ = N^-/(1+N^-)$ to guarantee that the optimization would yield a classifier with the highest possible TPR with a maximum FPR less than 20% (see Section 2.2).
- We added a feature-based constraint using the loss variables to limit the maximum number of features to 5 (see Section 2.2). We then set $C_0 = 0.9W^-/NP$ so that the optimization would yield a classifier that did not sacrifice accuracy for sparsity (see Remark 3).
- We added sign constraints to the coefficients to ensure that our model would not violate established relationships between features and the predicted outcome (i.e., we set $\lambda_j \geq 0$ if there had to be a positive relationship, and $\lambda_j \leq 0$ if there had to be a negative relationship).

With this setup, we trained 10 models with subsets of the data to assess predictive accuracy via 10-fold cross validation (10-CV), and 1 final model with all of data to hand over to our collaborators. We set up each IP using the slim_for_matlab toolbox (Ustun, 2015) and solved each IP for 1 hour, in parallel, on 12-core 2.7GHZ machine with 48GB RAM. Thus, the training process for SLIM required 1 hour of computing time.

As a comparison, we trained models with 8 baseline classification methods shown in Table 1. We dealt with the class imbalance by using a cost-sensitive approach, where we used a weighted loss function and varied its sensitivity parameter W^+ across a large range. When possible, we addressed the remaining operational constraints by searching over a fine grid of free parameters. Model selection was difficult for baseline methods because they could not accomodate operational constraints in the same way as SLIM. For each baseline method, we chose the best model that satisfied all operational constraints by: (i) dropping any instance of the free parameters where operational constraints were violated; (ii) choosing the instance that maximized the 10-CV mean test TPR. We ruled that an instance of the free parameters violated an operational constraint if any of the following conditions were met: (1) the 10-CV mean test FPR of the model produced with the instance was greater than the 10-CV mean test FPR of the SLIM model (20.9%); (2) the model size[2] of the final model produced with the instance was greater than 5; (3) the final model produced did not obey sign constraints. This model selection procedure may have biased the results in favor of the baseline methods because we mixed testing and training data by looking at the final model to ensure that operational constraints were satisfied.

5.3 Results and Observations

In what follows, we report our observations related to operational constraints, predictive performance and interpretability. We show the performance of the best model we trained using each method in Table 2, and summarize the operational constraints they were able to satisfy in Table 3.

[2] Model size represents the number of coefficients for linear models (Lasso, Ridge, Elastic Net, SLIM, SVM Lin.), the number of leaves for decision tree models (C5.0T, CART), and the number of rules for rule-based models (C5.0R). For completeness, we set the model size for black-box models (SVM RBF) to the number of features in each dataset.

Method	Controls	# Instances	Settings and Free Parameters
CART	Max FPR Model Size	39	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$
C5.0T	Max FPR	39	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$
C5.0R	Max FPR Model Size	39	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$
Lasso	Max FPR Model Size Signs	39000	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$ $\times$ 1000 values of λ chosen by **glmnet**
Ridge	Max FPR Signs	39000	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$ $\times$ 1000 values of λ chosen by **glmnet**
Elastic Net	Max FPR Model Size Signs	975000	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$ $\times$ 1000 values of λ chosen by **glmnet** $\times$ 19 values of $\alpha \in \{0.05, 0.10, \ldots, 0.95\}$
SVM Lin.	Max FPR	975	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$ $\times$ 25 values of $C \in \{10^{-3}, 10^{-2.75}, \ldots, 10^{3}\}$
SVM RBF	Max FPR	975	39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$ $\times$ 25 values of $C \in \{10^{-3}, 10^{-2.75}, \ldots, 10^{3}\}$
SLIM	Max FPR Model Size Signs	1	$W^+ = N^-/(1+N^-)$, $C_0 = 0.9W^-/NP$, $\lambda_0 \in \{-100, \ldots, 100\}$, $\lambda_j \in \{-10, \ldots, 10\}$

Table 1: Training setup for all methods. An instance is a unique combination of free parameters. Controls refer to operational constraints that we expect each method to handle. We include further details on methods and software packages in Table 5.

On the Difficulties of Handling Operational Constraints

Among the 9 classification methods that we used, only SLIM, Lasso and Elastic Net could produce a model that satisfied all of operational constraints given to us by physicians. Tree and rule-based methods such as CART, C5.0 Tree and C5.0 Rule were unable to produce a model with a maximum FPR of 20% (see Figure 4). Methods that used ℓ_2-regularization such as Ridge, SVM Lin. and SVM RBF were unable to produce a model with the required level of sparsity. While we did not expect all methods to satisfy all of the operational constraints, we included them to emphasize the following important points. Namely, state-of-the-art methods for applied predictive modeling do not:

- Handle simple operational constraints that are crucial for models to be used and accepted. Implementations of popular classification methods do not have a mechanism to adjust important model qualities. That is, there is no mechanism to control sparsity in C5.0T (Kuhn et al. 2012) and no mechanism to incorporate sign constraints in SVM (Meyer et al. 2012). Finding a method with suitable controls is especially difficult when a model has to satisfy multiple operational constraints.
- Have controls that are easy-to-use and/or that work correctly. When a method has suitable controls to handle operational constraints, producing a model often requires a tuning process over a high-dimensional free parameter grid. Even after extensive tuning, however, it is possible to never find a model that satisfies all operational constraints (e.g. CART, C5.0R, C5.0T for the Max FPR constraint in Figure 4).
- Allow tuning to be portable when the training set changes. Consider a standard model selection procedure where we choose free parameters to maximize predictive accuracy. In this case, we would train models on several folds for each instance of the free parameters, choose an instance of the free parameters that maximized our estimate of predictive accuracy among the instances that met all operational constraints, and then train a final model using these values of the free parameters. Unfortunately, there is no guarantee that the final model will obey all operational constraints.

		OBJECTIVE	CONSTRAINTS		OTHER INFORMATION				
Method	**Constraints Satisfied**	**Test TPR**	**Test FPR**	**Final Model Size**	**Model Size**	**Train TPR**	**Train FPR**	**Final Train TPR**	**Final Train FPR**
SLIM	All	61.4% 55.5 - 68.8%	20.9% 15.0 - 30.4%	5 -	5 5 - 5	62.4% 61.0 - 64.2%	19.7% 19.3 - 20.0%	62.0% -	19.6% -
Lasso	All	29.3% 19.2 - 60.0%	8.6% 0.0 - 33.3%	3 -	3 3 - 6	28.7% 21.4 - 54.6%	7.2% 3.5 - 20.5%	22.1% -	3.8% -
Elastic Net	All	44.2% 0.0 - 64.1%	18.8% 0.0 - 37.0%	3 -	3 3 - 6	45.6% 0.0 - 66.5%	17.4% 0.0 - 36.4%	54.3% -	20.7% -
Ridge	Max FPR	66.0% 60.5 - 68.5%	20.6% 8.6 - 32.6%	30 -	30 30 - 30	66.4% 64.0 - 68.9%	18.9% 17.3 - 21.5%	66.0% -	18.9% -
SVM RBF	Max FPR	64.3% 59.2 - 71.1%	20.8% 10.0 - 30.4%	33 -	33 33 - 33	67.9% 66.5 - 70.0%	12.2% 11.1 - 13.3%	67.8% -	12.4% -
SVM Lin.	Max FPR	62.7% 57.9 - 69.0%	19.8% 7.5 - 28.6%	31 -	31 31 - 31	63.7% 61.5 - 66.1%	17.0% 15.6 - 18.5%	63.1% -	17.1% -
C5.0R	None	84.0% 78.9 - 87.7%	43.0% 32.6 - 54.2%	26 -	23 18 - 30	86.1% 84.2 - 88.5%	33.8% 30.9 - 38.2%	85.5% -	32.9% -
C5.0T	None	81.3% 77.4 - 84.8%	42.9% 29.6 - 62.5%	39 -	42 39 - 50	85.3% 82.6 - 88.6%	29.5% 24.6 - 33.7%	84.5% -	28.4% -
CART	None	93.0% 88.8 - 96.1%	70.4% 61.1 - 83.3%	8 -	9 4 - 12	95.2% 93.1 - 97.2%	66.8% 55.0 - 76.0%	95.9% -	73.9% -

Table 2: TPR, FPR and model size for all methods. We report the 10-CV mean TPR and FPR, and the 10-CV median for the model size. The ranges in each cell represent the 10-CV minimum and maximum.

	% of Instances that Satisfied		
Method	**Max FPR**	**Max FPR & Model Size**	**Max FPR, Model Size & Signs**
SLIM	100.0%	100.0%	100.0%
Lasso	19.6%	4.8%	4.8%
Elastic Net	18.3%	1.0%	1.0%
Ridge	20.9%	0.0%	0.0%
SVM Lin	18.7%	0.0%	0.0%
SVM RBF	15.8%	0.0%	0.0%
C5.0R	0.0%	0.0%	0.0%
C5.0T	0.0%	0.0%	0.0%
CART	0.0%	0.0%	0.0%

Table 3: Percentage of instances that fulfilled operational constraints. Each instance is a unique combination of free parameters for a given method.

On the Sensitivity of Acceptable Models

Among the three methods that produced acceptable models, the scoring system produced by SLIM had significantly higher sensitivity than the models produced by Lasso and Elastic Net – a result that we expected given that SLIM minimizes the 0–1 loss and an ℓ_0-penalty while Lasso and Elastic Net minimize convex surrogates of these quantities. This result held true even when we relaxed various operational constraints. In Figure 5, for instance, we plot the sensitivity and sparsity of models that satisfied the max FPR and sign constraints. Here, we see that Lasso and Elastic Net need at least 8 coefficients to produce a model with the same degree of sensitivity as SLIM. In Figure 6, we plot the TPR and FPR of models

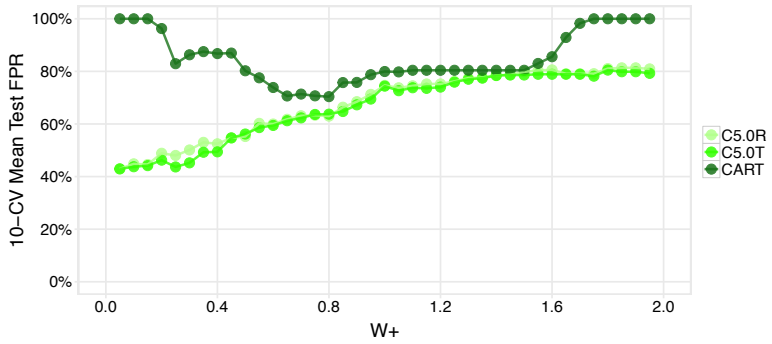

Fig. 4: 10-CV mean test FPR for models trained with CART, C5.0, C5.0T across the full range of W^+. These methods cannot produce a model that satisfies the max FPR $\leq 20\%$ constraint.

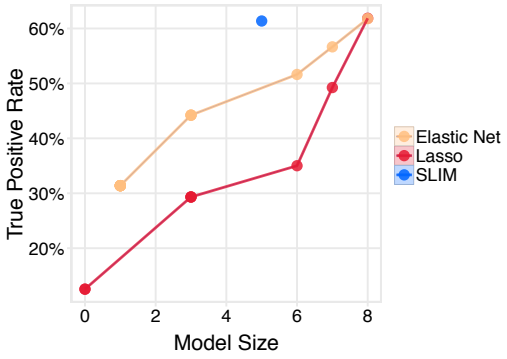

Fig. 5: Sensitivity and model size of Lasso and Elastic Net models that satisfy the sign and FPR constraints. For each method, we plot the instance that attains the highest 10-CV mean test TPR at model sizes between 0 and 8. Lasso and Elastic Net need at least 8 coefficients to produce a model with the same sensitivity as SLIM.

that satisfied the sign and model size constraints. As shown, SLIM scoring systems dominate Lasso and Elastic Net models across the entire ROC curve. These sensitivity advantages are also evident in Table 2: in particular, SLIM yields a model with a similar level of sensitivity and specificity as Ridge and SVM Lin. even as it is fitting models from a far smaller hypothesis space (i.e. linear classifiers with 5 features, sign constraints and integer coefficients vs. linear classifiers with real coefficients).

On the Usability and Interpretability of Acceptable Models

To discuss interpretability, we compare the best models that satisfied all operational constraints in Figure 7, and present the SLIM model as a scoring system in Figure 8.

In this case, our collaborators found that all three models were aligned with domain knowledge as they obeyed sign constraints and had large coefficients for well-known risk factors such as *bmi*, *female*, *age*, *snoring* and/or *hypertension*. Unfortunately, the Lasso and Elastic Net models could not be deployed as screening tools due to their poor sensitivity (29.3% for Lasso and 44.2% for Elastic Net). This was not the case for the SLIM model, which had a much higher sensitivity (61.4%).

Our results highlight some of the unique *interpretability* benefits of SLIM scoring systems – that is, their ability to provide "a *qualitative understanding* of the relationship between *joint* values of the input variables and the resulting predicted response value" (Hastie et al., 2009). SLIM scoring systems are well-suited to

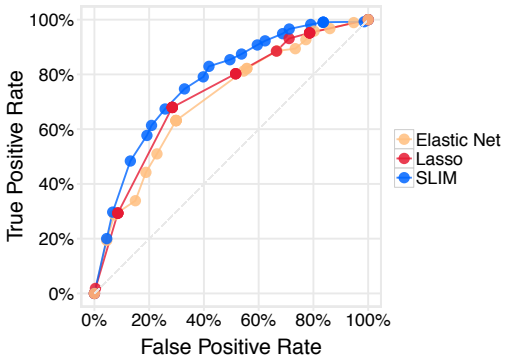

Fig. 6: ROC curve for SLIM, Lasso and Elastic Net instances that satisfy the sign and model size constraints. For each method, we plot the instance that attains the highest 10-CV mean test TPR for 10-CV mean FPR values of $5\%, 10\%, \ldots, 95\%$. Note that we had to train 19 additional instances of SLIM to create this plot.

SLIM	4 $age \geq 60$	+	4 *hypertension*	+	2 $bmi \geq 30$	+	2 $bmi \geq 40$	−	6 *female*	− 1
Lasso	0.13 *snoring*	+	0.12 *hypertension*	−	0.26 *female*	−	0.17			
Elastic Net	0.03 *snoring*	+	0.02 *hypertension*	−	0.09 *female*	−	0.02			

Fig. 7: Score functions of the most sensitive predictive models that satisfied all three operational constraints. The baseline models have very poor sensitivity as shown in Table 2.

PREDICT PATIENT HAS OBSTRUCTIVE SLEEP APNEA IF SCORE > 1

1.	$age \geq 60$	4 points		.....
2.	*hypertension*	4 points	+	.....
3.	*body mass index* ≥ 30	2 points	+	.....
4.	*body mass index* ≥ 40	2 points	+	.....
5.	*female*	-6 points	+	.....
	ADD POINTS FROM ROWS 1 – 5	**SCORE**	=	.....

Fig. 8: SLIM scoring system for sleep apnea screening. This model achieves a 10-CV mean test TPR/FPR of 61.4/20.9%, obeys all operational constraints, and was trained without parameter tuning. It also generalizes well due to the simplicity of the hypothesis space: here the training TPR/FPR of the final model is 62.0/19.6%.

provide this kind of qualitative understanding due to their high level of sparsity and small integer coefficients. These qualities help users gauge the influence of each input variable with respect to the others, which is especially important because humans can only handle a few cognitive entities at once (7 ± 2 according to Miller 1984), and are seriously limited in estimating the association between three or more variables (Jennings et al., 1982). Sparsity and small integer coefficients also allow users to make quick predictions without a computer or a calculator, which may help them understand how the model works by actively using it to classify prototypical examples. Here, this process helped our collaborators come up with the following simple rule-based explanation for our model predicted that a patient has OSA (i.e., when SCORE > 1): "*if the patient is male, predict OSA if age* $\geq$ *60 OR hypertension OR bmi* $\geq$ *30; if the patient is female, predict OSA if bmi* $\geq$ *40 AND (age* $\geq$ *60 OR hypertension).*"

6 Numerical Experiments

In this section, we present numerical experiments to compare the accuracy and sparsity of SLIM scoring systems to other popular classification models. Our goal is to illustrate the off-the-shelf performance of SLIM and show that we can train accurate scoring systems for real-sized datasets in minutes.

6.1 Experimental Setup

Datasets: We ran numerical experiments on 8 datasets from the UCI Machine Learning Repository (Bache and Lichman, 2013) summarized in Table 4. We chose these datasets to explore the performance of each method as we varied the size and nature of the training data. We processed each dataset by binarizing all categorical features and some real-valued features. For the purposes of reproducibility, we include all processed datasets in Online Resource 1.

Dataset	Source	N	P	Classification Task
`adult`	Kohavi (1996)	32561	36	predict if a U.S. resident earns more than $50 000
`breastcancer`	Mangasarian et al. (1995)	683	9	detect breast cancer using a biopsy
`bankruptcy`	Kim and Han (2003)	250	6	predict if a firm will go bankrupt
`haberman`	Haberman (1976)	306	3	predict 5-year survival after breast cancer surgery
`heart`	Detrano et al. (1989)	303	32	identify patients a high risk of heart disease
`mammo`	Elter et al. (2007)	961	12	detect breast cancer using a mammogram
`mushroom`	Schlimmer (1987)	8124	113	predict if a mushroom is poisonous
`spambase`	Cranor and LaMacchia (1998)	4601	57	predict if an e-mail is spam

Table 4: Datasets used in the numerical experiments.

Methods: We summarize the training setup for each method in Table 5. We trained SLIM scoring systems using slim_for_matlab toolbox paired with the CPLEX 12.6.0 API and models with baseline methods using publicly available packages in R 3.1.1 (R Core Team, 2014). For each method, each dataset, and each unique combination of free parameters, we trained 10 models using subsets of the data to estimate predictive accuracy via 10-fold cross-validation (10-CV), and 1 final model using all of the data to assess sparsity and interpretability. We ran all baseline methods without time constraints over a large grid of free parameters. We produced an ℓ_0-regularization path for SLIM by solving 6×11 IPs for each dataset (6 values of $C_0 \times 11$ training runs per C_0). We allocated at most 10 minutes to solve each IP, and solved 12 IPs in parallel on a 12-core 2.7 GHZ machine with 48 GB RAM. Thus, it took at most 1 hour to train SLIM scoring systems for each dataset. Since the `adult` and `haberman` datasets were imbalanced, we trained all methods on these datasets with a weighted loss function where we set $W^+ = N^-/N$ and $W^- = N^+/N$.

6.2 Results and Observations

We summarize the results of our experiments in Table 6 and Figures 13–14. We report the sparsity of models using a metric that we call *model size*. Model size represents the number of coefficients for linear models (Lasso, Ridge, Elastic Net, SLIM, SVM Lin.), the number of leaves for decision tree models (C5.0T, CART), and the number of rules for rule-based models (C5.0R). For completeness, we set the model size for black-box models (SVM RBF) to the number of features in each dataset.

Method	Acronym	Software	Settings and Free Parameters
CART Decision Trees	CART	**rpart** (Therneau et al., 2012)	default settings
C5.0 Decision Trees	C5.0T	**c50** (Kuhn et al., 2012)	default settings
C5.0 Rule List	C5.0R	**c50** (Kuhn et al., 2012)	default settings
Log. Reg. + ℓ_1 penalty	Lasso	**glmnet** (Friedman et al., 2010)	1000 values of λ chosen by **glmnet**
Log. Reg. + ℓ_2 penalty	Ridge	**glmnet** (Friedman et al., 2010)	1000 values of λ chosen by **glmnet**
Log. Reg. + ℓ_1/ℓ_2 penalty	Elastic Net	**glmnet** (Friedman et al., 2010)	1000 values of λ chosen by **glmnet** $\times$ 19 values of $\alpha \in \{0.05, 0.10, \ldots, 0.95\}$
SVM + Linear Kernel	SVM Lin.	**e1071** (Meyer et al., 2012)	25 values of $C \in \{10^{-3}, 10^{-2.75}, \ldots, 10^{3}\}$
SVM + RBF Kernel	SVM RBF	**e1071** (Meyer et al., 2012)	25 values of $C \in \{10^{-3}, 10^{-2.75}, \ldots, 10^{3}\}$
SLIM Scoring Systems	SLIM	**slim_for_matlab** (Ustun, 2015)	6 values of $C_0 \in \{0.01, 0.075, 0.05, 0.025, 0.001, 0.9/NP\}$ with $\lambda_j \in \{-10, \ldots, 10\}$; $\lambda_0 \in \{-100, \ldots, 100\}$

Table 5: Training setup for classification methods used for the numerical experiments.

We show the accuracy and sparsity of all methods on all dataset in Figures 13–14. For each dataset, and each method, we plot a point at the 10-CV mean test error and final model size, and surround this point with an error bar whose height corresponds to the 10-CV standard deviation in test error. In addition, we include ℓ_0-regularization paths for SLIM and Lasso on the right side of Figures 13–14 to show how the test error varies at different levels of sparsity for sparse linear models.

Dataset	Details		Metric	SLIM	Lasso	Ridge	Elastic Net	C5.0R	C5.0T	CART	SVM Lin.	SVM RBF
adult	N	32561	**test error**	17.4 ± 1.4%	17.3 ± 0.9%	17.6 ± 0.9%	17.4 ± 0.9%	26.4 ± 1.8%	26.3 ± 1.4%	75.9 ± 0.0%	16.8 ± 0.8%	16.3 ± 0.5%
	P	37	**train error**	17.5 ± 1.2%	17.2 ± 0.1%	17.6 ± 0.1%	17.4 ± 0.1%	25.3 ± 0.4%	24.9 ± 0.4%	75.9 ± 0.0%	16.7 ± 0.1%	16.3 ± 0.1%
	Pr(y=+1)	24%	**model size**	18	14	36	17	41	87	4	36	36
	Pr(y=−1)	76%	**model range**	7 - 26	13 - 14	36 - 36	16 - 18	38 - 46	78 - 99	4 - 4	36 - 36	36 - 36
breastcancer	N	683	**test error**	3.4 ± 2.0%	3.4 ± 2.2%	3.4 ± 2.0%	3.1 ± 2.1%	4.3 ± 3.3%	5.3 ± 3.4%	5.6 ± 1.9%	3.1 ± 2.0%	3.5 ± 2.5%
	P	10	**train error**	3.2 ± 0.2%	2.9 ± 0.3%	3.0 ± 0.3%	2.8 ± 0.3%	2.1 ± 0.3%	1.6 ± 0.4%	3.6 ± 0.3%	2.7 ± 0.2%	0.3 ± 0.1%
	Pr(y=+1)	35%	**model size**	2	9	9	9	8	13	7	9	9
	Pr(y=−1)	65%	**model range**	2 - 2	8 - 9	9 - 9	9 - 9	6 - 9	7 - 16	3 - 7	9 - 9	9 - 9
bankruptcy	N	250	**test error**	0.8 ± 1.7%	0.0 ± 0.0%	0.4 ± 1.3%	0.0 ± 0.0%	0.8 ± 1.7%	0.8 ± 1.7%	1.6 ± 2.8%	0.4 ± 1.3%	0.4 ± 1.3%
	P	7	**train error**	0.0 ± 0.0%	0.0 ± 0.0%	0.4 ± 0.1%	0.4 ± 0.7%	0.4 ± 0.2%	0.4 ± 0.2%	1.6 ± 0.3%	0.4 ± 0.1%	0.4 ± 0.1%
	Pr(y=+1)	57%	**model size**	3	3	6	3	4	4	2	6	6
	Pr(y=−1)	43%	**model range**	2 - 3	3 - 3	6 - 6	3 - 3	4 - 4	4 - 4	2 - 2	6 - 6	6 - 6
haberman	N	306	**test error**	29.2 ± 14.0%	42.5 ± 11.3%	36.9 ± 15.0%	40.9 ± 14.0%	42.7 ± 9.4%	42.7 ± 9.4%	43.1 ± 8.0%	45.3 ± 14.7%	47.5 ± 6.2%
	P	4	**train error**	29.7 ± 1.5%	40.6 ± 1.9%	41.0 ± 9.7%	45.1 ± 12.0%	40.4 ± 8.5%	40.4 ± 8.5%	34.3 ± 2.8%	46.0 ± 3.6%	5.4 ± 1.5%
	Pr(y=+1)	74%	**model size**	3	2	3	1	3	3	9	3	4
	Pr(y=−1)	26%	**model range**	2 - 3	2 - 2	3 - 3	1 - 1	0 - 3	1 - 3	4 - 9	3 - 3	4 - 4
mammo	N	961	**test error**	19.5 ± 3.0%	19.0 ± 3.1%	19.2 ± 3.0%	19.0 ± 3.1%	20.5 ± 3.3%	20.3 ± 3.5%	20.7 ± 3.9%	20.3 ± 3.0%	19.1 ± 3.1%
	P	15	**train error**	18.3 ± 0.3%	19.3 ± 0.3%	19.2 ± 0.4%	19.2 ± 0.3%	19.8 ± 0.3%	19.9 ± 0.3%	20.0 ± 0.6%	20.3 ± 0.4%	18.2 ± 0.4%
	Pr(y=+1)	46%	**model size**	9	13	14	14	5	5	5	14	14
	Pr(y=−1)	54%	**model range**	9 - 11	12 - 13	14 - 14	13 - 14	3 - 5	4 - 6	3 - 5	14 - 14	14 - 14
heart	N	303	**test error**	16.5 ± 7.8%	15.2 ± 6.3%	14.9 ± 5.9%	14.5 ± 5.9%	21.2 ± 7.5%	23.2 ± 6.8%	19.8 ± 6.5%	15.5 ± 6.5%	15.2 ± 6.0%
	P	33	**train error**	14.9 ± 1.1%	14.0 ± 1.0%	13.1 ± 0.8%	13.2 ± 0.6%	10.0 ± 1.8%	8.5 ± 2.0%	14.3 ± 0.9%	13.6 ± 0.5%	10.4 ± 0.8%
	Pr(y=+1)	46%	**model size**	3	12	32	24	7	16	6	31	32
	Pr(y=−1)	54%	**model range**	3 - 3	10 - 13	30 - 32	22 - 27	9 - 17	12 - 27	6 - 8	28 - 32	32 - 32
mushroom	N	8124	**test error**	0.0 ± 0.0%	0.0 ± 0.0%	1.7 ± 0.3%	0.0 ± 0.0%	0.0 ± 0.0%	0.0 ± 0.0%	1.2 ± 0.6%	0.0 ± 0.0%	0.0 ± 0.0%
	P	114	**train error**	0.0 ± 0.0%	0.0 ± 0.0%	1.7 ± 0.0%	0.0 ± 0.0%	0.0 ± 0.0%	0.0 ± 0.0%	1.1 ± 0.3%	0.0 ± 0.0%	0.0 ± 0.0%
	Pr(y=+1)	48%	**model size**	7	21	113	108	8	9	7	98	113
	Pr(y=−1)	52%	**model range**	7 - 7	19 - 23	113 - 113	106 - 108	8 - 8	9 - 9	6 - 8	98 - 108	113 - 113
spambase	N	4601	**test error**	6.3 ± 1.2%	10.0 ± 1.7%	26.3 ± 1.7%	10.0 ± 1.7%	6.6 ± 1.3%	7.3 ± 1.0%	11.1 ± 1.4%	7.8 ± 1.5%	13.7 ± 1.4%
	P	58	**train error**	5.7 ± 0.3%	9.5 ± 0.3%	26.1 ± 0.2%	9.6 ± 0.2%	4.2 ± 0.3%	3.9 ± 0.3%	9.8 ± 0.3%	8.1 ± 0.8%	1.3 ± 0.1%
	Pr(y=+1)	39%	**model size**	34	28	57	28	29	73	7	57	57
	Pr(y=−1)	61%	**model range**	28 - 40	28 - 29	57 - 57	28 - 29	23 - 31	56 - 78	6 - 10	57 - 57	57 - 57

Table 6: Accuracy and sparsity of all methods on all datasets. Here: test error refers to the 10-CV mean test error ± the 10-CV standard deviation in test error; train error refers to the 10-CV mean training error ± the 10-CV standard deviation in training error; model size refers to the final model size; and model range refers to the 10-CV minimum and maximum model size. The results reflect the models produced by each method when free parameters are chosen to minimize the 10-CV mean test error. We report the 10-CV weighted test and training error for `adult` and `haberman`.

We wish to make the following observations regarding our results:

On the Accuracy, Sparsity and Computation

Our results show that many methods are unable to produce models that attain the same levels of accuracy and sparsity as SLIM. As shown in Figures 13–14, SLIM always produces a model that is more accurate than Lasso at some level of sparsity, and sometimes more accurate at all levels of sparsity (e.g., `spambase`, `haberman`, `mushroom`, `breastcancer`). Although optimization problems to train SLIM scoring systems were $\mathcal{NP}$-hard, we did not find any evidence that computational issues hurt the performance of SLIM on any of the datasets. We obtained accurate and sparse models for all datasets in 10 minutes using CPLEX 12.6. Further, the solver provided a proof of optimality (i.e., a MIPGAP of 0.0%) for all models we trained for `mammo`, `mushroom`, `bankruptcy`, `breastcancer`. We attribute these benefits to SLIM's tighter MIP formulation (see Section 2.1).

On the Regularization Effect of Discrete Coefficients

We expect that methods that directly optimize accuracy and sparsity will achieve the best possible accuracy at every level of sparsity (i.e. the best possible trade-off between accuracy and sparsity). SLIM directly optimizes accuracy and sparsity. However, it may not necessarily achieve the best possible accuracy at each level of sparsity because it restricts coefficients to a finite discrete set $\mathcal{L}$.

By comparing SLIM to Lasso, we can identify a baseline regularization effect due to this $\mathcal{L}$ set restriction. In particular, we know that when Lasso's performance dominates that of SLIM, it is very arguably due to the use of a small set of discrete coefficients. Our results show that this tends to happen mainly at large model sizes (see e.g., the regularization path for `breastcancer`, `heart`, `mammo`). This suggests that the $\mathcal{L}$ set restriction has a more noticeable impact on accuracy at larger model sizes.

One interesting effect of the $\mathcal{L}$ set restriction is that the most accurate SLIM scoring system may not use all of the features in the dataset. In our experiments, we always trained SLIM with $C_0 = 0.9/NP$ to obtain a scoring system with the highest training accuracy among linear models with coefficients in $\boldsymbol{\lambda} \in \mathcal{L}$ (see Remark 3). In the `bankruptcy` dataset, for example, we find that this model only uses 3 out of 6 features. This is due to the $\mathcal{L}$ set restriction: if the $\mathcal{L}$ restriction were relaxed, then the method would use all features to improve its training accuracy (as is the case with Ridge or SVM Lin.).

On the Interpretability of Models

To discuss interpretability, we focus on the `mushroom` dataset, which provides a nice basis for comparison as many methods produce a model that attains perfect predictive accuracy. In Figures 9–12, we show the sparsest models that achieve perfect predictive accuracy. We omit models from some methods because they do not attain perfect accuracy (CART), or use far more features (Ridge, SVM Lin, SVM RBF).

Here, the SLIM scoring system uses 7 integer coefficients. However, it can be simplified into a 5 line scoring system since *odor=none*, *odor=almond*, and *odor=anise* are mutually exclusive variables with the same coefficient. The model lets users make predictions by hand, and uses a linear form that helps users gauge the influence of each input variable with respect to the others. Note that only some of these qualities are found in the other models. The Lasso model, for instance, has a linear form but uses far more features. In contrast, the C5.0 models let users to make predictions by hand, but have a hierarchical structure that makes it difficult to gauge the influence of each input variable with respect to the others.

We note that these qualities represent "baseline" interpretability benefits. In practice, interpretability is a subjective and multifaceted notion (i.e., it depends on who will be using the model, and on many model qualities, as discussed in Kodratoff (1994), Pazzani (2000), Freitas (2014)). In light of this, SLIM has a additional interpretability benefit because it allows practitioners to work closely with their target audience and encode all interpretability-related requirements into their model by means of operational constraints.

PREDICT MUSHROOM IS POISONOUS IF SCORE $>$ 3

1.	*spore_print_color* = *green*	4 points		·····
2.	*stalk_surface_above_ring* = *grooves*	2 points	+	·····
3.	*population* = *clustered*	2 points	+	·····
4.	*gill_size* = *broad*	-2 points	+	·····
5.	*odor* $\in$ {*none*, *almond*, *anise*}	-4 points	+	·····
	ADD POINTS FROM ROWS 1–5	**SCORE**	=	·····

Fig. 9: SLIM scoring system for `mushroom`. This model has a 10-CV mean test error of 0.0 ± 0.0%.

	10.86	*spore_print_color* = *green*	+	4.49	*gill_size* = *narrow*	+	4.29	*odor* = *foul*
+	2.73	*stalk_surface_below_ring* = *scaly*	+	2.60	*stalk_surface_above_ring* = *grooves*	+	2.38	*population* = *clustered*
+	0.85	*spore_print_color* = *white*	+	0.44	*stalk_root* = *bulbous*	+	0.43	*gill_spacing* = *close*
+	0.38	*cap_color* = *white*	+	0.01	*stalk_color_below_ring* = *yellow*	−	8.61	*odor* = *anise*
−	8.61	*odor* = *almond*	−	8.51	*odor* = *none*	−	0.53	*cap_surface* = *fibrous*
−	0.25	*population* = *solitary*	−	0.21	*stalk_surface_below_ring* = *fibrous*	−	0.09	*spore_print_color* = *brown*
−	0.00	*cap_shape* = *convex*	−	0.00	*gill_spacing* = *crowded*	−	0.00	*gill_size* = *broad*
+	0.25							

Fig. 10: Lasso score function for `mushroom`. This model has a 10-CV mean test error of 0.0 ± 0.0%.

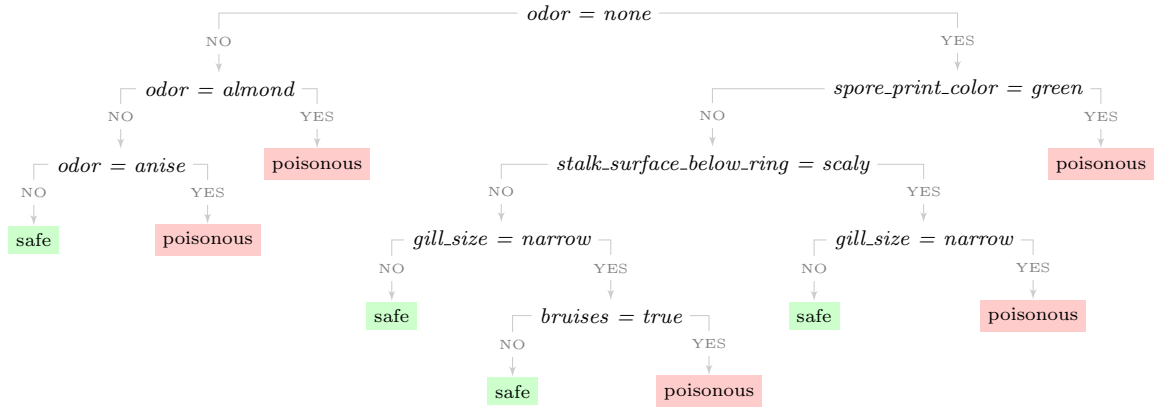

Fig. 11: C5.0 decision tree for `mushroom`. This model has a 10-CV mean test error of 0.0±0.0%.

Rule			**Confidence**	**Support**	**Lift**
odor = *none* ∧ *gill_size* ≠ *narrow* ∧ *spore_print_color* ≠ *green*	⟹	safe	1.000	3216	1.9
bruises = *false* ∧ *odor* = *none* ∧ *stalk_surface_below_ring* ≠ *scaly*	⟹	safe	0.999	1440	1.9
odor = *almond*	⟹	safe	0.998	400	1.9
odor = *anise*	⟹	safe	0.998	400	1.9
odor ≠ *almond* ∧ *odor* ≠ *anise* ∧ *odor* ≠ *none*	⟹	poisonous	1.000	3796	2.1
spore_print_color = *green*	⟹	poisonous	0.986	72	2.9
gill_size = *narrow* ∧ *stalk_surface_below_ring* = *scaly*	⟹	poisonous	0.976	40	2.0

Fig. 12: C5.0 rule list for `mushroom`. This model has a 10-CV mean test error of 0.0 ± 0.0%.

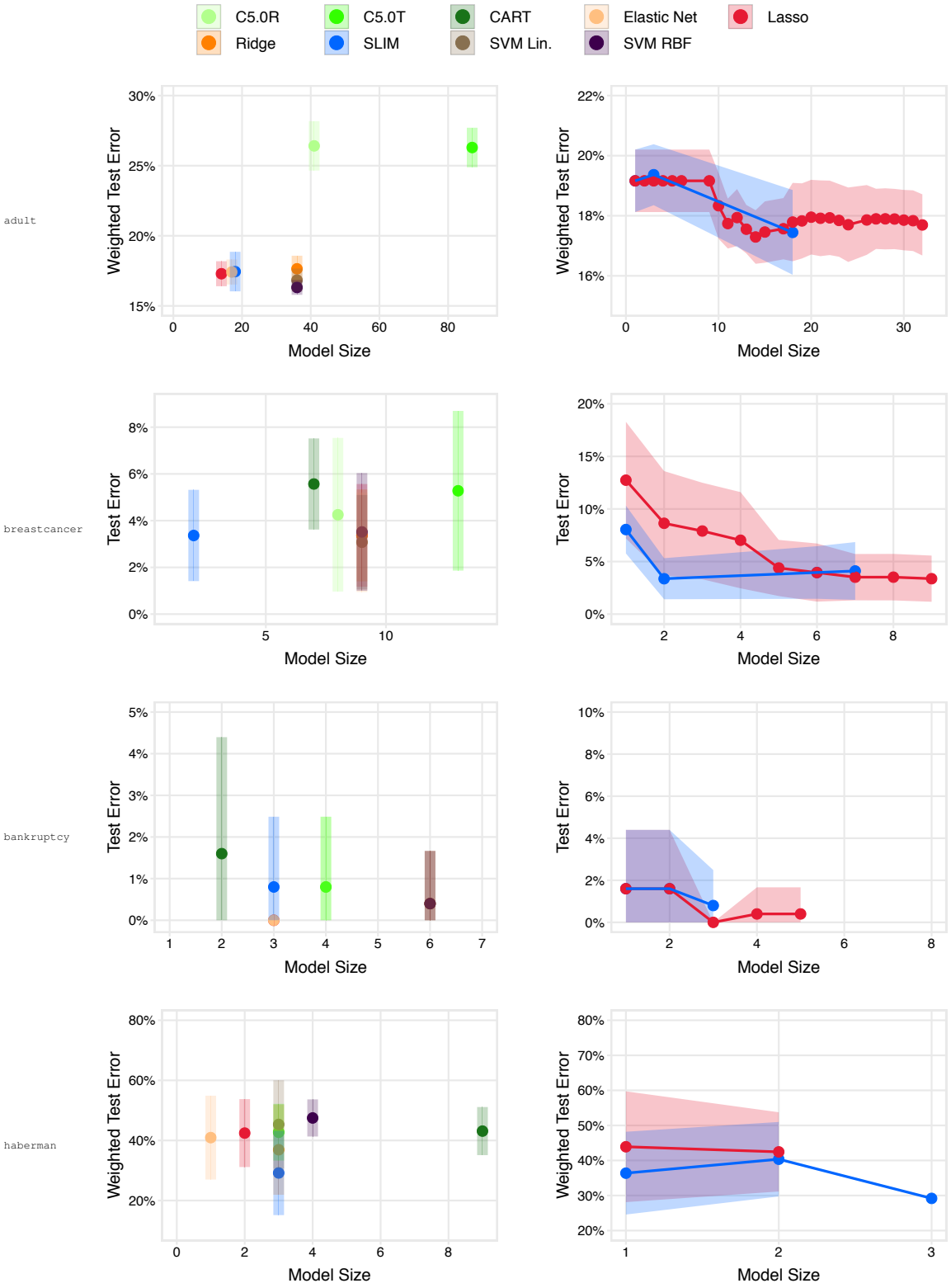

Fig. 13: Accuracy and sparsity of all classification methods on all datasets. For each dataset, we plot the performance of models when free parameters are set to values that minimize the 10-CV mean test error (left), and plot the performance of SLIM and Lasso across the full ℓ_0-regularization path (right).

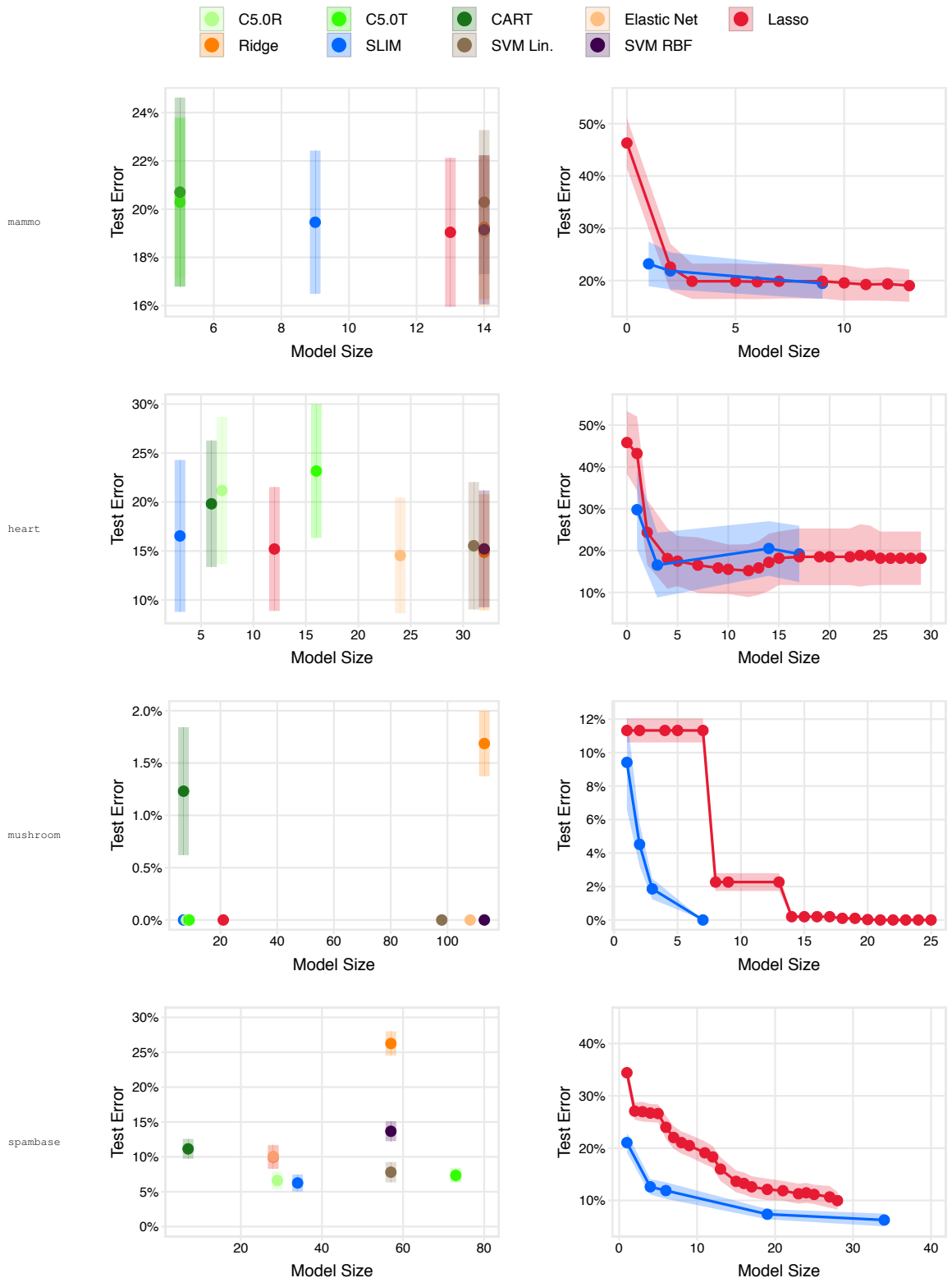

Fig. 14: Accuracy and sparsity of all classification methods on all datasets. For each dataset, we plot the performance of models when free parameters are set to values that minimize the 10-CV mean test error (left)m and plot the performance of SLIM and Lasso across the full ℓ_0-regularization path (right).

7 Specialized Models

In this section, we present three specialized models related to SLIM. These models are all special instances of the optimization problem in (1).

7.1 Personalized Models

A *Personalized Integer Linear Model* (PILM) is a generalization of SLIM that provides soft control over the coefficients in a scoring system. To use this model, users define $R+1$ interpretability sets,

$$\mathcal{L}_r = \{l_{r,1}, \ldots, l_{r,K_r}\} \text{ for } r = 0, \ldots, R,$$

as well as a "personalized" interpretability penalty,

$$\Phi_j(\lambda_j) = \begin{cases} C_0 & \text{if } \lambda_j \in \mathcal{L}_0 \\ \vdots & \\ C_R & \text{if } \lambda_j \in \mathcal{L}_R. \end{cases}$$

In order to penalize coefficients from less interpretable sets more heavily, we need that: (i) $\mathcal{L}_1, \ldots, \mathcal{L}_R$ are mutually exclusive; (ii) $\mathcal{L}_r$ is more interpretable than $\mathcal{L}_{r+1}$; (iii) the trade-off parameters are monotonically increasing in r, so that $C_0 < \ldots < C_R$. The values of the parameters C_r can be set as the minimum gain in training accuracy required for the optimal classifier to use a coefficient from $\mathcal{L}_r$.

As an example, consider training a PILM scoring system with the penalty:

$$\Phi_j(\lambda_j) = \begin{cases} C_0 = 0.00 & \text{if} \quad \lambda_j \in 0 \\ C_1 = 0.01 & \text{if} \quad \lambda_j \in \pm\{1, \ldots, 10\} \\ C_2 = 0.05 & \text{if} \quad \lambda_j \in \pm\{11, \ldots, 100\}. \end{cases}$$

Here, the optimal classifier will use a coefficient from $\mathcal{L}_1$ if it yields at least a 1% gain in training accuracy, and a coefficient from $\mathcal{L}_2$ if it yields at least a 5% gain in training accuracy.

We can train a PILM scoring system by solving the following IP:

$$\min_{\boldsymbol{\lambda},\psi,\boldsymbol{\Phi},\mathbf{u}} \; \frac{1}{N}\sum_{i=1}^{N} \psi_i + \sum_{j=1}^{P} \Phi_j$$

$$\text{s.t.} \quad M_i\psi_i \geq \gamma - \sum_{j=0}^{P} y_i\lambda_j x_{i,j} \quad i = 1,...,N \qquad \textit{0–1 loss} \quad (12a)$$

$$\Phi_j = \sum_{r=0}^{R}\sum_{k=1}^{K_r} C_r u_{j,k,r} \quad j = 1,...,P \qquad \textit{int. penalty} \quad (12b)$$

$$\lambda_j = \sum_{r=0}^{R}\sum_{k=1}^{K_r} l_{r,k} u_{j,k,r} \quad j = 0,...,P \qquad \textit{coefficient values} \quad (12c)$$

$$1 = \sum_{r=0}^{R}\sum_{k=1}^{K_r} u_{j,k,r} \quad j = 0,...,P \qquad \textit{1 int. set per coef.} \quad (12d)$$

$$\psi_i \in \{0,1\} \quad i = 1,...,N \qquad \textit{loss variables}$$

$$\Phi_j \in \mathbb{R}_+ \quad j = 1,...,P \qquad \textit{int. penalty variables}$$

$$u_{j,r,k} \in \{0,1\} \quad j = 0,...,P \quad r = 0,...,R \quad k = 1,...,K_r \qquad \textit{coef. value variables}$$

Here, the loss constraints and Big-M parameters in (12a) are identical to those from the SLIM IP formulation in Section 2. The $u_{j,k,r}$ are binary indicator variables that are set to 1 if λ_j is equal to $l_{k,r}$. Constraints (12d) ensure that each coefficient uses exactly one value from one interpretability set. Constraints (12c) ensure that each coefficient λ_j is assigned a value from the appropriate interpretability set $\mathcal{L}_r$. Constraints (12b) ensure that each coefficient λ_j is assigned the value specified by the personalized interpretability penalty.

7.2 Rule-Based Models

SLIM can be extended to produce specialized "rule-based" models when the training data are composed of *binary rules*. In general, any real-valued feature can be converted into a binary rule by setting a threshold (e.g., we can convert *age* into the feature $age \geq 25 := \mathbb{1}[age \geq 25]$). The values of the thresholds can be set using domain expertise, rule mining, or discretization techniques (Liu et al., 2002).

In what follows, we assume that we train models with a binarized dataset that contains T_j binary rules $\boldsymbol{h}_{j,t} \in \{0,1\}^N$ for each feature $\boldsymbol{x}_j \in \mathbb{R}^N$ in the original dataset. Thus, we consider models with the form:

$$\hat{y} = \text{sign}\left(\lambda_0 + \sum_{j=1}^{P}\sum_{t=1}^{T_j}\lambda_{j,t}h_{j,t}\right).$$

We make the following assumptions about the binarization process. If $\boldsymbol{x}_j$ is a binary variable, then it is left unchanged so that $T_j = 1$ and $\boldsymbol{h}_{j,T_j} := \boldsymbol{x}_j$. If $\boldsymbol{x}_j$ is a categorical variable $\boldsymbol{x}_j \in \{1,\ldots,K\}$, the binarization yields a binary rule for each category so that $T_j = K$ and $\boldsymbol{h}_{j,t} := \mathbb{1}[\boldsymbol{x}_j = k]$ for $t = 1,\ldots,K$. If $\boldsymbol{x}_j$ is a real variable, then the binarization yields T_j binary rules[3] of the form $\boldsymbol{h}_{j,t} := \mathbb{1}[\boldsymbol{x}_j \geq v_{j,t}]$ where $v_{j,t}$ denotes the t^{th} threshold for feature j.

7.2.1 M-of-N Rule Tables

M-of-N rule tables are simple rule-based models that, given a set of N rules, predict $\hat{y} = +1$ if at least M of them are true (see e.g., Figure 15). These models have the major benefit that they do not require the user to compute a mathematical expression. M-of-N rule tables were originally proposed as auxiliary models that could be extracted from neural nets (Towell and Shavlik, 1993) but can also be trained as stand-alone discrete linear classification models as suggested by Chevaleyre et al. (2013).

We can produce a fully optimized M-of-N rule table by solving an optimization problem of the form:

$$\begin{aligned}
\min_{\boldsymbol{\lambda}} \quad & \sum_{i=1}^{N}\mathbb{1}[y_i\hat{y}_i \leq 0] + C_0\|\boldsymbol{\lambda}\|_0 \\
\text{s.t.} \quad & \lambda_0 \in \{-P,\ldots,0\} \\
& \lambda_{j,t} \in \{0,1\} \qquad\qquad j = 1,...,P \quad t = 1,...,T_j.
\end{aligned}$$

The coefficients from this optimization problem yield an M-of-N rule table with $M = \lambda_0 + 1$ and $N = \sum_{j=1}^{P}\sum_{t=1}^{T_j}\lambda_{j,t}$. Here, we can achieve exact ℓ_0-regularization using an ℓ_1-penalty since $\|\lambda_{j,t}\|_0 = \|\lambda_{j,t}\|_1$ for $\lambda_{j,t} \in \{0,1\}$. Since we use the 0–1 loss, the trade-off parameter C_0 can be set as minimum gain in training accuracy required to include a rule in the optimal table.

[3] While there exists an infinite number of thresholds for a real-valued feature, we only need consider at most $N-1$ thresholds (i.e. one threshold placed each pair of adjacent values, $x_{(i),j} < v_{j,t} < x_{(i+1),j}$). Using additional thresholds will produce the same set of binary rules and the same rule-based model.

PREDICT TUMOR IS BENIGN
IF AT LEAST 5 OF THE FOLLOWING 8 RULES ARE TRUE

$$UniformityOfCellSize \geq 3$$
$$UniformityOfCellShape \geq 3$$
$$MarginalAdhesion \geq 3$$
$$SingleEpithelialCellSize \geq 3$$
$$BareNuclei \geq 3$$
$$NormalNucleoli \geq 3$$
$$BlandChromatin \geq 3$$
$$Mitoses \geq 3$$

Fig. 15: M-of-N rule table for the `breastcancer` dataset for $C_0 = 0.9/NP$. This model has 8 rules and a 10-CV mean test error of $4.8 \pm 2.5\%$. We trained this model with binary rules $h_{i,j} := \mathbb{1}[x_{i,j} \geq 3]$.

We can train an M-of-N rule table by solving the following IP:

$$\min_{\boldsymbol{\lambda},\psi,\boldsymbol{\Phi}} \ \frac{1}{N}\sum_{i=1}^{N}\psi_i + \sum_{j=1}^{P}\Phi_j$$

$$\text{s.t.} \quad M_i\psi_i \geq \gamma - \sum_{j=0}^{P}\sum_{t=1}^{T_j} y_i\lambda_{j,t}h_{i,j,t} \quad i = 1,...,N \qquad \textit{0–1 loss} \tag{13a}$$

$$\begin{aligned}
\Phi_{j,t} &= C_0\lambda_{j,t} && j = 1,...,P \quad t = 1,...,T_j && \textit{int. penalty} \\
\lambda_0 &\in \{-P,\ldots,0\} && && \textit{intercept value} \\
\lambda_{j,t} &\in \{0,1\} && j = 1,...,P \quad t = 1,...,T_j && \textit{coefficient values} \\
\psi_i &\in \{0,1\} && i = 1,...,N && \textit{0–1 loss indicators} \\
\Phi_{j,t} &\in \mathbb{R}_+ && j = 1,...,P \quad t = 1,...,T_j && \textit{int. penalty values}
\end{aligned} \tag{13b}$$

Here, the loss constraints and Big-M parameters in (13a) are identical to those from the SLIM IP formulation in Section 2. Constraints (13b) define the penalty variables $\Phi_{j,t}$ as the value of the ℓ_0-penalty.

7.2.2 Threshold-Rule Models

A *Threshold-Rule Integer Linear Model* (TILM) is a scoring system where the input variables are thresholded versions of the original feature set (i.e. decision stumps). These models are well-suited to problems where the outcome has a non-linear relationship with real-valued features. As an example, consider the SAPS II scoring system of Le Gall et al. (1993), which assesses the mortality of patients in intensive care using thresholds on real-valued features such as $blood\_pressure > 200$ and $heart\_rate < 40$. TILM optimizes the binarization of real-valued features by using feature selection on a large (potentially exhaustive) pool of binary rules for each real-valued feature. Carrizosa et al. (2010), Van Belle et al. (2013) and Goh and Rudin (2014) take different but related approaches for constructing classifiers with binary threshold rules.

We train TILM scoring systems using an optimization problem of the form:

$$\begin{aligned}
\min_{\boldsymbol{\lambda}} \quad & \frac{1}{N}\sum_{i=1}^{N}\mathbb{1}[y_i\hat{y}_i \leq 0] + C_f \cdot \text{Features} + C_t \cdot \text{Rules per Feature} + \epsilon\|\boldsymbol{\lambda}\|_1 \\
\text{s.t.} \quad & \boldsymbol{\lambda} \in \mathcal{L}, \\
& \sum_{t=1}^{T_j}\mathbb{1}[\lambda_{j,t} \neq 0] \leq R_{max} \text{ for } j = 1,\ldots,P, \\
& \text{sign}(\lambda_{j,1}) = \ldots = \text{sign}\left(\lambda_{j,T_j}\right) \text{ for } j = 1,\ldots,P.
\end{aligned}$$

TILM uses an interpretability penalty that penalizes the number of rules used in the classifier as well as the number of features associated with these rules. The small ℓ_1-penalty in the objective restricts coefficients to coprime values as in SLIM. Here, C_f tunes the number of features used in the model, C_t tunes the number of rules per feature, and ϵ is set to a small value to produce coprime coefficients. TILM includes additional hard constraints to limit the number of rules per feature to R_{max} (e.g., $R_{max} = 3$), and to ensure that the coefficients for binary rules from a single feature agree in sign (this ensures that each feature maintains a strictly monotonically increasing or decreasing relationship with the outcome).

We can train a TILM scoring system by solving the following IP:

$$
\begin{aligned}
\min_{\boldsymbol{\lambda},\psi,\boldsymbol{\Phi},\tau,\nu,\delta} \quad & \frac{1}{N}\sum_{i=1}^{N}\psi_i + \sum_{j=1}^{P}\Phi_j & & & & \\
\text{s.t.} \quad & M_i\psi_i \geq \gamma - \sum_{j=0}^{P}\sum_{t=1}^{T_j} y_i\lambda_{j,t}h_{i,j,t} & i = 1,...,N & & \textit{0–1 loss} & \text{(14a)} \\
& \Phi_j = C_f\nu_j + C_t\tau_j + \epsilon\sum_{t=1}^{T_j}\beta_{j,t} & j = 1,...,P & & \textit{int. penalty} & \text{(14b)} \\
& T_j\nu_j = \sum_{t=1}^{T_j}\alpha_{j,t} & j = 1,...,P & & \textit{feature use} & \text{(14c)} \\
& \tau_j = \sum_{t=1}^{T_j}\alpha_{j,t} - 1 & j = 1,...,P & & \textit{threshold/feature} & \text{(14d)} \\
& \tau_j \leq R_{max} + 1 & j = 1,...,P & & \textit{max thresholds} & \text{(14e)} \\
& -\Lambda_j\alpha_{j,t} \leq \lambda_{j,t} \leq \Lambda_j\alpha_{j,t} & j = 1,...,P & t = 1,...,T_j & \ell_0 \textit{ norm} & \\
& -\beta_{j,t} \leq \lambda_{j,t} \leq \beta_{j,t} & j = 1,...,P & t = 1,...,T_j & \ell_1 \textit{ norm} & \\
& -\Lambda_j(1-\delta_j) \leq \lambda_{j,t} \leq \Lambda_j\delta_j & j = 1,...,P & t = 1,...,T_j & \textit{agree in sign} & \text{(14f)} \\
& \lambda_{j,t} \in \mathcal{L}_j & j = 0,...,P & t = 1,...,T_j & \textit{coefficient values} & \\
& \psi_i \in \{0,1\} & i = 1,...,N & & \textit{0–1 loss indicators} & \\
& \Phi_j \in \mathbb{R}_+ & j = 1,...,P & & \textit{int. penalty variables} & \\
& \alpha_j \in \{0,1\} & j = 1,...,P & & \ell_0 \textit{ variables} & \\
& \beta_j \in \mathbb{R}_+ & j = 1,...,P & & \ell_1 \textit{ variables} & \\
& \nu_j \in \{0,1\} & j = 1,...,P & & \textit{feature use indicators} & \\
& \tau_j \in \mathbb{Z}_+ & j = 1,...,P & & \textit{threshold/feature variables} & \\
& \delta_j \in \{0,1\} & j = 1,...,P & & \textit{sign indicators} &
\end{aligned}
$$

Here, the loss constraints and Big-M parameters in (14a) are identical to those from the SLIM IP formulation in Section 2. Constraints (14b) set the interpretability penalty for each coefficient as $\Phi_j = C_f\nu_j + C_t\tau_j + \epsilon\sum\beta_{j,t}$. The variables in the interpretability penalty include: ν_j, which indicate that we use at least one threshold rule from feature j; τ_j, which count the number of additional binary rules derived from feature j; and $\beta_{j,t} := |\lambda_{j,t}|$. The values of ν_j and τ_j are set using the indicator variables $\alpha_{j,t} := \mathbb{1}[\lambda_{j,t} \neq 0]$ in constraints (14c) and (14d). Constraints (14e) limit the number of binary rules from feature j to $\mathbb{R}_{max}$. Constraints (14f) ensure that the coefficients of binary rules derived from feature j agree in sign; these constraints are encoded using the variables $\delta_j := \mathbb{1}[\lambda_{j,t} \geq 0]$.

8 Conclusion

In this paper, we introduced a new method for creating data-driven medical scoring systems which we refer to as a Supersparse Linear Integer Model (SLIM). We showed how SLIM can produce scoring systems that are fully optimized for accuracy and sparsity, that can accomodate multiple operational constraints, and that can be trained without parameter tuning.

The major benefits of our approach over existing methods come from the fact that we avoid approximations that are designed to achieve faster computation. Approximations such as surrogate loss functions and ℓ_1-regularization hinder the accuracy and sparsity of models as well as the ability of practitioners to control these qualities. Such approximations are no longer needed for many datasets, since using current integer programming software, we can now train scoring systems for many real-world problems. Integer programming software also caters to practitioners in other ways, by allowing them to choose from a pool of models by mining feasible solutions and to seamlessly benefit from periodic computational improvements without revising their code.

Acknowledgments

We thank the editors and reviewers for valuable comments that helped improve this paper. In addition, we thank Dr. Matt Bianchi and Dr. Brandon Westover at the Massachusetts General Hospital Sleep Clinic for providing us with data used in Section 5. We gratefully acknowledge support from Siemens and Wistron.

References

Antman, Elliott M, Marc Cohen, Peter JLM Bernink, Carolyn H McCabe, Thomas Horacek, Gary Papuchis, Branco Mautner, Ramon Corbalan, David Radley, and Eugene Braunwald. The TIMI risk score for unstable angina/non–ST elevation MI. *The Journal of the American Medical Association*, 284(7):835–842, 2000.

Bache, K. and M. Lichman. UCI machine learning repository, 2013.

Bien, Jacob, Jonathan Taylor, Robert Tibshirani, and others. A lasso for hierarchical interactions. *The Annals of Statistics*, 41(3):1111–1141, 2013.

Bone, RC, RA Balk, FB Cerra, RP Dellinger, AM Fein, WA Knaus, RM Schein, WJ Sibbald, JH Abrams, GR Bernard, and others. American college of chest physicians/society of critical care medicine consensus conference: Definitions for sepsis and organ failure and guidelines for the use of innovative therapies in sepsis. *Critical Care Medicine*, 20(6):864–874, 1992.

Bousquet, Olivier, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.

Brooks, J Paul. Support vector machines with the ramp loss and the hard margin loss. *Operations Research*, 59(2):467–479, 2011.

Carrizosa, Emilio, Belen Martín-Barragán, and Dolores Romero Morales. Binarized support vector machines. *INFORMS Journal on Computing*, 22(1):154–167, 2010.

Carrizosa, Emilio, Amaya Nogales-Gómez, and Dolores Romero Morales. Strongly agree or strongly disagree?: Rating features in support vector machines. *Information Sciences*, 329:256–273, 2016.

Chevaleyre, Yann, Frederic Koriche, and Jean-Daniel Zucker. Rounding methods for discrete linear classification. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 651–659, 2013.

Cranor, Lorrie Faith and Brian A LaMacchia. Spam! *Communications of the ACM*, 41(8):74–83, 1998.

Detrano, Robert, Andras Janosi, Walter Steinbrunn, Matthias Pfisterer, Johann-Jakob Schmid, Sarbjit Sandhu, Kern H Guppy, Stella Lee, and Victor Froelicher. International application of a new probability algorithm for the diagnosis of coronary artery disease. *The American journal of cardiology*, 64(5):304–310, 1989.

Efron, Bradley, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

Elter, M, R Schulz-Wendtland, and T Wittenberg. The prediction of breast cancer biopsy outcomes using two cad approaches that both emphasize an intelligible decision process. *Medical Physics*, 34(11):4164–4172, 2007.

Freitas, Alex A. Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10, March 2014.

Friedman, Jerome, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010.

Gage, Brian F, Amy D Waterman, William Shannon, Michael Boechler, Michael W Rich, and Martha J Radford. Validation of clinical classification schemes for predicting stroke. *The Journal of the American Medical Association*, 285(22):2864–2870, 2001.

Goh, Siong Thye and Cynthia Rudin. Box drawings for learning with imbalanced data. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 333–342. ACM, 2014.

Guan, Wei, Alex Gray, and Sven Leyffer. Mixed-integer support vector machine. In *NIPS Workshop on Optimization for Machine Learning*, 2009.

Haberman, Shelby J. Generalized residuals for log-linear models. In *Proceedings of the 9th international biometrics conference, Boston*, pages 104–122, 1976.

Hastie, Trevor, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

Jenatton, Rodolphe, Jean-Yves Audibert, and Francis Bach. Structured variable selection with sparsity-inducing norms. *The Journal of Machine Learning Research*, 12:2777–2824, 2011.

Jennings, D, TM Amabile, and L Ross. Informal covariation assessment: Data-based vs. theory-based judgments. *Judgment under uncertainty: Heuristics and biases*, pages 211–230, 1982.

Kapur, Vishesh K. Obstructive sleep apnea: diagnosis, epidemiology, and economics. *Respiratory care*, 55 (9):1155–1167, 2010.

Kim, Myoung-Jong and Ingoo Han. The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms. *Expert Systems with Applications*, 25(4):637–646, 2003.

Knaus, William A, Jack E Zimmerman, Douglas P Wagner, Elizabeth A Draper, and Diane E Lawrence. APACHE-acute physiology and chronic health evaluation: a physiologically based classification system. *Critical Care Medicine*, 9(8):591–597, 1981.

Knaus, William A, Elizabeth A Draper, Douglas P Wagner, and Jack E Zimmerman. APACHE II: a severity of disease classification system. *Critical Care Medicine*, 13(10):818–829, 1985.

Knaus, William A, DP Wagner, EA Draper, JE Zimmerman, Marilyn Bergner, PG Bastos, CA Sirio, DJ Murphy, T Lotring, and A Damiano. The APACHE III prognostic system. risk prediction of hospital mortality for critically ill hospitalized adults. *Chest Journal*, 100(6):1619–1636, 1991.

Kodratoff, Y. The comprehensibility manifesto. *KDD Nugget Newsletter*, 94(9), 1994.

Kohavi, Ron. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *KDD*, pages 202–207, 1996.

Kuhn, Max, Steve Weston, and Nathan Coulter. *C50: C5.0 Decision Trees and Rule-Based Models*, 2012. C code for C5.0 by R. Quinlan. R package version 0.1.0-013.

Le Gall, Jean-Roger, Stanley Lemeshow, and Fabienne Saulnier. A new simplified acute physiology score (SAPS II) based on a european/north american multicenter study. *The Journal of the American Medical Association*, 270(24):2957–2963, 1993.

Lin, Dongyu, Emily Pitler, Dean P Foster, and Lyle H Ungar. In defense of l0. In *Workshop on Feature Selection,(ICML 2008)*, 2008.

Liu, H, F Hussain, C L Tan, and M Dash. Discretization: An enabling technique. *Data mining and knowledge discovery*, 2002.

Mangasarian, Olvi L, W Nick Street, and William H Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, 1995.

Marklof, J. Fine-scale statistics for the multidimensional Farey sequence. *ArXiv e-prints*, July 2012.

Meyer, David, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, and Friedrich Leisch. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2012. R package version 1.6-1.

Miller, Alan J. Selection of subsets of regression variables. *Journal of the Royal Statistical Society. Series A (General)*, pages 389–425, 1984.

Moreno, Rui P, Philipp GH Metnitz, Eduardo Almeida, Barbara Jordan, Peter Bauer, Ricardo Abizanda Campos, Gaetano Iapichino, David Edbrooke, Maurizia Capuzzo, and Jean-Roger Le Gall. SAPS 3 - from evaluation of the patient to evaluation of the intensive care unit. part 2: Development of a prognostic model for hospital mortality at icu admission. *Intensive Care Medicine*, 31(10):1345–1355, 2005.

Nguyen, Tan and Scott Sanner. Algorithms for direct 0–1 loss optimization in binary classification. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1085–1093, 2013.

Pazzani, Michael J. Knowledge discovery from data? *Intelligent systems and their applications, IEEE*, 15(2): 10–12, 2000.

R Core Team, . *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. URL http://www.R-project.org/.

Rubin, Paul A. Mixed integer classification problems. In *Encyclopedia of Optimization*, pages 2210–2214. Springer, 2009.

Schlimmer, Jeffrey Curtis. Concept acquisition through representational adjustment. 1987.

Souillard-Mandar, William, Randall Davis, Cynthia Rudin, Rhoda Au, David J Libon, Rodney Swenson, Catherine C Price, Melissa Lamar, and Dana L Penney. Learning classification models of cognitive conditions from subtle behaviors in the digital clock drawing test. *Machine Learning*, pages 1–49, 2015.

Therneau, Terry, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning*, 2012. URL http://CRAN.R-project.org/package=rpart. R package version 4.1-0.

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

Towell, G G and J W Shavlik. Extracting refined rules from knowledge-based neural networks. *Machine Learning*, 1993.

Ustun, Berk. *slim_for_matlab: build optimized scoring systems using MATLAB and the CPLEX API*, 2015. URL http://github.com/ustunb/slim_for_matlab. GitHub Repository. Commit fb0b9222c59fe26307fa039b89ec95d619a5f577.

Ustun, Berk, Brandon M. Westover, Cynthia Rudin, and Matt T. Bianchi. Clinical prediction models for sleep apnea: superiority of medical history over symptoms. *Journal of Clinical Sleep Medicine (forthcoming)*, 2015.

Van Belle, Vanya, Patrick Neven, Vernon Harvey, Sabine Van Huffel, Johan AK Suykens, and Stephen Boyd. Risk group detection and survival function estimation for interval coded survival methods. *Neurocomputing*, 112:200–210, 2013.

Vapnik, Vladimir. *Statistical Learning Theory*. Wiley, New York, 1998.

Wolsey, Laurence A. *Integer programming*, volume 42. Wiley New York, 1998.

Zhao, Peng and Bin Yu. On model selection consistency of lasso. *Journal of Machine Learning Research*, 7 (2):25–41, 2007.

Zou, Hui and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

A Proofs of Theorems

Proof of Theorem 1 (Minimum Margin Resolution Bound)

Proof We use normalized versions of the vectors, $\boldsymbol{\rho}/\left\|\boldsymbol{\rho}\right\|_2$ and $\boldsymbol{\lambda}/\Lambda$ because the 0–1 loss is scale invariant:

$$\sum_{i=1}^{N} \mathbb{1}\left[y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i \leq 0\right] = \sum_{i=1}^{N} \mathbb{1}\left[y_i \frac{\boldsymbol{\lambda}^T \boldsymbol{x}_i}{\Lambda} \leq 0\right],$$
$$\sum_{i=1}^{N} \mathbb{1}\left[y_i \boldsymbol{\rho}^T \boldsymbol{x}_i \leq 0\right] = \sum_{i=1}^{N} \mathbb{1}\left[y_i \frac{\boldsymbol{\rho}^T \boldsymbol{x}_i}{\left\|\boldsymbol{\rho}\right\|_2} \leq 0\right].$$

We set $\Lambda > \frac{X_{\max}\sqrt{P}}{2\gamma_{\min}}$ as in (5). Using Λ, we then define $\boldsymbol{\lambda}/\Lambda$ element-wise so that λ_j/Λ is equal to $\rho_j/\left\|\boldsymbol{\rho}\right\|_2$ rounded to the nearest $1/\Lambda$ for $j = 1, \ldots, P$.

We first show that our choice of Λ and $\boldsymbol{\lambda}$ ensures that the difference between the margin of $\boldsymbol{\rho}/\left\|\boldsymbol{\rho}\right\|_2$ and the margin of $\boldsymbol{\lambda}/\Lambda$ on all training examples is always less than the minimum margin of $\boldsymbol{\rho}/\left\|\boldsymbol{\rho}\right\|_2$, defined as $\gamma_{\min} = \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2}$. This statement follows from the fact that, for all i:

$$\begin{aligned}
\left|\frac{\boldsymbol{\lambda}^T \boldsymbol{x}_i}{\Lambda} - \frac{\boldsymbol{\rho}^T \boldsymbol{x}_i}{\left\|\boldsymbol{\rho}\right\|_2}\right| &\leq \left\|\frac{\boldsymbol{\lambda}}{\Lambda} - \frac{\boldsymbol{\rho}}{\left\|\boldsymbol{\rho}\right\|_2}\right\|_2 \left\|\boldsymbol{x}_i\right\|_2 && (15)\\
&= \left(\sum_{j=1}^{P} \left|\frac{\lambda_j}{\Lambda} - \frac{\rho_j}{\left\|\boldsymbol{\rho}\right\|_2}\right|^2\right)^{1/2} \left\|\boldsymbol{x}_i\right\|_2 \\
&\leq \left(\sum_{j=1}^{P} \frac{1}{(2\Lambda)^2}\right)^{1/2} \left\|\boldsymbol{x}_i\right\|_2 && (16)\\
&= \frac{\sqrt{P}}{2\Lambda} X_{\max} \\
&< \frac{\sqrt{P} X_{\max}}{2\left(\frac{X_{\max}\sqrt{P}}{2\min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2}}\right)} && (17)\\
&= \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2}. && (18)
\end{aligned}$$

Here: the inequality in (15) uses the Cauchy-Schwarz inequality; the inequality in (16) is due to the fact that the distance between $\rho_j/\left\|\boldsymbol{\rho}\right\|_2$ and λ_j/Λ is at most $1/2\Lambda$; and the inequality in (17) is due to our choice of Λ.

Next, we show that our choice of Λ and $\boldsymbol{\lambda}$ ensures that $\boldsymbol{\rho}/\left\|\boldsymbol{\rho}\right\|_2$ and $\boldsymbol{\lambda}/\Lambda$ classify each point in the same way. We consider three cases: first, the case where $\boldsymbol{x}_i$ lies on the margin; second, the case where $\boldsymbol{\rho}$ has a positive margin on $\boldsymbol{x}_i$; and third, the case where $\boldsymbol{\rho}$ has a negative margin on $\boldsymbol{x}_i$. For the case when $\boldsymbol{x}_i$ lies on the margin, $\min_i |\boldsymbol{\rho}^T \boldsymbol{x}_i| = 0$ and the theorem holds trivially. For the case where $\boldsymbol{\rho}$ has positive margin, $\boldsymbol{\rho}^T \boldsymbol{x}_i > 0$, the following calculation using (18) is relevant:

$$\frac{\boldsymbol{\rho}^T \boldsymbol{x}_i}{\left\|\boldsymbol{\rho}\right\|_2} - \frac{\boldsymbol{\lambda}^T \boldsymbol{x}_i}{\Lambda} \leq \left|\frac{\boldsymbol{\lambda}^T \boldsymbol{x}_i}{\Lambda} - \frac{\boldsymbol{\rho}^T \boldsymbol{x}_i}{\left\|\boldsymbol{\rho}\right\|_2}\right| < \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2}.$$

We will use the fact that for any i', by definition of the minimum:

$$0 \leq \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_{i'}|}{\left\|\boldsymbol{\rho}\right\|_2} - \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2},$$

and combine this with a rearrangement of the previous expression to obtain:

$$0 \leq \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2} - \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2} = \frac{\boldsymbol{\rho}^T \boldsymbol{x}_i}{\left\|\boldsymbol{\rho}\right\|_2} - \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\left\|\boldsymbol{\rho}\right\|_2} < \frac{\boldsymbol{\lambda}^T \boldsymbol{x}_i}{\Lambda}.$$

Thus, we have shown that $\boldsymbol{\lambda}^T\boldsymbol{x}_i > 0$ whenever $\boldsymbol{\rho}^T\boldsymbol{x}_i > 0$.

For the case where $\boldsymbol{\rho}$ has a negative margin on $\boldsymbol{x}_i$, $\boldsymbol{\rho}^T\boldsymbol{x}_i < 0$, we perform an analogous calculation:

$$\frac{\boldsymbol{\lambda}^T\boldsymbol{x}_i}{\|\boldsymbol{\lambda}\|_2} - \frac{\boldsymbol{\rho}^T\boldsymbol{x}_i}{\|\boldsymbol{\rho}\|_2} \leq \left|\frac{\boldsymbol{\lambda}^T\boldsymbol{x}_i}{\Lambda} - \frac{\boldsymbol{\rho}^T\boldsymbol{x}_i}{\|\boldsymbol{\rho}\|_2}\right| < \min_i \frac{|\boldsymbol{\rho}^T\boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2}.$$

and then using that $\boldsymbol{\rho}^T\boldsymbol{x}_i < 0$,

$$0 \leq \frac{|\boldsymbol{\rho}^T\boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2} - \min_i \frac{|\boldsymbol{\rho}^T\boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2} = \frac{-\boldsymbol{\rho}^T\boldsymbol{x}_i}{\|\boldsymbol{\rho}\|_2} - \min_i \frac{|\boldsymbol{\rho}^T\boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2} < -\frac{\boldsymbol{\lambda}^T\boldsymbol{x}_i}{\Lambda}.$$

Thus, we have shown $\boldsymbol{\lambda}^T\boldsymbol{x}_i < 0$ whenever $\boldsymbol{\rho}^T\boldsymbol{x}_i < 0$.

Putting both the positive margin and negative margin cases together, we find that for all i,

$$\mathbb{1}\left[y_i\boldsymbol{\rho}^T\boldsymbol{x}_i \leq 0\right] = \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right].$$

Summing over i yields the statement of the theorem. □

Proof of Theorem 3 (Generalization of Sparse Discrete Linear Classifiers)

Proof Let $Z(\boldsymbol{\lambda};\mathcal{D}_N) = \frac{1}{N}\sum_{i=1}^N \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right] + C_0\|\boldsymbol{\lambda}\|_0$. Note that $\boldsymbol{\lambda} = 0$ is a feasible solution since we assume that $0 \in \mathcal{L}$. Since $\boldsymbol{\lambda} = 0$ achieves an objective value of $Z(0;\mathcal{D}_N) = 1$, any optimal solution, $\boldsymbol{\lambda} \in \operatorname{argmin}_{\lambda\in\mathcal{L}} Z(\boldsymbol{\lambda};\mathcal{D}_N)$, must attain an objective value $Z(\boldsymbol{\lambda};\mathcal{D}_N) \leq 1$. This implies

$$\begin{aligned} Z(\boldsymbol{\lambda};\mathcal{D}_N) &\leq 1, \\ C_0\|\boldsymbol{\lambda}\|_0 \leq \frac{1}{N}\sum_{i=1}^N \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \leq 0\right] + C_0\|\boldsymbol{\lambda}\|_0 &\leq 1, \\ \|\boldsymbol{\lambda}\|_0 &\leq \frac{1}{C_0}, \\ \|\boldsymbol{\lambda}\|_0 &\leq \left\lfloor\frac{1}{C_0}\right\rfloor. \end{aligned}$$

The last line uses that $\|\boldsymbol{\lambda}\|_0$ is an integer.

Thus, $\mathcal{H}_{P,C_0}$ is large enough to contain all minimizers of $Z(\cdot;\mathcal{D}_N)$ for any $\mathcal{D}_N$. The statement of the theorem follows from applying Theorem 2. □

Proof of Theorem 5 (Equivalence of the Reduced Data)

Proof Let us denote the set of classifiers whose objective value is less or equal to $\tilde{Z}(\tilde{f}^*;\mathcal{D}_N)$ as

$$\tilde{\mathcal{F}}^\varepsilon = \left\{f \in \tilde{\mathcal{F}} \;\middle|\; \tilde{Z}(f;\mathcal{D}_N) \leq \tilde{Z}(\tilde{f};\mathcal{D}_N) + \varepsilon\right\}.$$

In addition, let us denote the set of points that have been removed by the data reduction algorithm

$$\mathcal{S} = \mathcal{D}_N \setminus \mathcal{D}_M.$$

By definition, data reduction only removes an example if its sign is fixed. This means that $\operatorname{sign}\left(f(\boldsymbol{x}_i)\right) = \operatorname{sign}\left(\tilde{f}(\boldsymbol{x}_i)\right)$ for all $i \in \mathcal{S}$ and $f \in \tilde{\mathcal{F}}^\varepsilon$. Thus, we can see that for all classifiers $f \in \tilde{\mathcal{F}}^\varepsilon$,

$$Z(f;\mathcal{D}_N) = Z(f;\mathcal{D}_M) + \sum_{i\in\mathcal{S}} \mathbb{1}\left[y_i f(\boldsymbol{x}_i) \leq 0\right] = Z(f;\mathcal{D}_M) + \sum_{i\in\mathcal{S}} \mathbb{1}\left[y_i\tilde{f}(\boldsymbol{x}_i) \leq 0\right] = Z(f;\mathcal{D}_M) + C. \tag{19}$$

We now proceed to prove the statement in (11). When $\mathcal{S} = \emptyset$, then $\mathcal{D}_N = \mathcal{D}_M$, and (11) follows trivially. When, $\mathcal{S} \neq \emptyset$, we note that

$$\begin{aligned}\mathcal{F}^* = \operatorname*{argmin}_{f\in\mathcal{F}} Z(f;\mathcal{D}_N) &= \operatorname*{argmin}_{f\in\mathcal{F}} Z(f;\mathcal{D}_M \cup \mathcal{S}),\\ &= \operatorname*{argmin}_{f\in\mathcal{F}} Z(f;\mathcal{D}_M) + Z(f;\mathcal{S}),\\ &= \operatorname*{argmin}_{f\in\mathcal{F}} Z(f;\mathcal{D}_M) + C, \qquad (20)\\ &= \operatorname*{argmin}_{f\in\mathcal{F}} Z(f;\mathcal{D}_M).\end{aligned}$$

Here, the statement in (20) follows directly from (19). □

Proof of Theorem 6 (Sufficient Conditions to Satisfy the Level Set Condition)

Proof We assume that we have found a surrogate function, ψ, that satisfies conditions I–IV and choose $C_\psi > 2\varepsilon$.

Our proof uses the following result: if $\|\boldsymbol{\lambda}_{01}^* - \boldsymbol{\lambda}_\psi^*\| > C_{\boldsymbol{\lambda}}$ then $\boldsymbol{\lambda}_{01}^*$ cannot be a minimizer of $Z_{01}(\boldsymbol{\lambda})$ because this would lead to a contradiction with the definition of $\boldsymbol{\lambda}_{01}^*$. To see that this result holds, we use condition III with $\boldsymbol{\lambda} = \boldsymbol{\lambda}_{01}^*$ to see that $\|\boldsymbol{\lambda}_{01}^* - \boldsymbol{\lambda}_\psi^*\| > C_{\boldsymbol{\lambda}}$ implies $Z_\psi(\boldsymbol{\lambda}_{01}^*) - Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) > C_\psi$. Thus,

$$\begin{aligned}Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) + C_\psi &< Z_\psi(\boldsymbol{\lambda}_{01}^*)\\ Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) + C_\psi &< Z_{01}(\boldsymbol{\lambda}_{01}^*) + \varepsilon \qquad (21)\\ Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) + C_\psi - \varepsilon &< Z_{01}(\boldsymbol{\lambda}_{01}^*)\\ Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) + C_\psi - \varepsilon &< Z_\psi(\boldsymbol{\lambda}_{01}^*) \qquad (22)\\ Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) + \varepsilon &< Z_\psi(\boldsymbol{\lambda}_{01}^*). \qquad (23)\end{aligned}$$

Here the inequality in (21) follows from condition IV, the inequality in (22) follows from condition I, and the inequality in (23) follows from our choice that $C_\psi > 2\varepsilon$.

We proceed by looking at the LHS and RHS of (23) separately. Using condition I on the LHS of (23) we get that:

$$Z_{01}\left(\boldsymbol{\lambda}_\psi^*\right) + \varepsilon \leq Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) + \varepsilon. \qquad (24)$$

Using condition IV on the RHS of (23) we get that:

$$Z_\psi(\boldsymbol{\lambda}_{01}^*) \leq Z_{01}(\boldsymbol{\lambda}_{01}^*) + \varepsilon. \qquad (25)$$

Combining the inequalities in (23), (24) and (25), we get that:

$$Z_{01}\left(\boldsymbol{\lambda}_\psi^*\right) < Z_{01}(\boldsymbol{\lambda}_{01}^*). \qquad (26)$$

The statement in (26) is a contradiction of the definition of $\boldsymbol{\lambda}_{01}^*$. Thus, we know that our assumption was incorrect and thus $\|\boldsymbol{\lambda}_{01}^* - \boldsymbol{\lambda}_\psi^*\| \leq C_{\boldsymbol{\lambda}}$. We plug this into the Lipschitz condition II as follows:

$$\begin{aligned}Z_\psi(\boldsymbol{\lambda}_{01}^*) - Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) \leq L\|\boldsymbol{\lambda}_{01}^* - \boldsymbol{\lambda}_\psi^*\| < LC_{\boldsymbol{\lambda}},\\ Z_\psi(\boldsymbol{\lambda}_{01}^*) < LC_{\boldsymbol{\lambda}} + Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right).\end{aligned}$$

Thus, we have satisfied the level set condition with $\varepsilon = LC_{\boldsymbol{\lambda}}$. □