Mach Learn (2016) 102:349–391
DOI 10.1007/s10994-015-5528-6

# Supersparse linear integer models for optimized medical scoring systems

**Berk Ustun**[1] · **Cynthia Rudin**[2]

Received: 1 February 2015 / Accepted: 5 August 2015 / Published online: 5 November 2015


**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 using an integer programming problem that directly encodes measures of accuracy (the 0–1 loss) and sparsity (the $\ell_0$-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 acceptable models without parameter tuning because of the direct control provided over these quantities. 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 is being used to create a highly tailored scoring system for sleep apnea screening.

**Electronic supplementary material** The online version of this article (doi:10.1007/s10994-015-5528-6) contains supplementary material, which is available to authorized users.

Editors: Byron Wallace and Jenna Wiens.

✉ Berk Ustun
ustunb@mit.edu

Cynthia Rudin
rudin@mit.edu

[1] Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA

[2] Sloan School of Management and CSAIL, Massachusetts Institute of Technology, Cambridge, MA, USA

**Keywords** Medical scoring systems · Discrete linear classification · Integer programming · 0–1 Loss · Sparsity · Interpretability · Sleep apnea · Supervised Classification

## 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 $\text{CHADS}_2$ 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, there has been little to no work that has focused on machine learning methods 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 especially challenging in medical applications because models also need to satisfy explicit constraints on operational quantities such as the false positive rate or the number of features. Such 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 approximations to speed up computation, and require rounding to yield models with coprime integer coefficients. Approximations such as convex surrogate loss functions, $\ell_1$-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 explicitly calculate its value, which is impossible when we control accuracy by means of a surrogate loss function. In practice, traditional methods can only address multiple operational constraints through a 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 scoring system 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 an integer program that optimizes direct measures of accuracy (the 0–1 loss) and sparsity (the $\ell_0$-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 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 general-

ization 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 is being 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.

The remainder of our paper is structured as follows. In the remainder of Sect. 1, we discuss related work. In Sect. 2, we introduce SLIM and discuss its special properties. In Sect. 3, we explain how SLIM can easily accommodate operational constraints that are important for medical scoring systems to be used in practice. In Sect. 4, we present theoretical bounds on the accuracy of SLIM scoring systems and other discrete linear classification models. In Sect. 5, we present a data reduction technique to decrease the computation associated with training SLIM scoring systems and other supervised classification models. In Sect. 6, 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 Sect. 7, we report experimental results to show that SLIM can create scoring systems that are accurate and sparse in minutes. In Sect. 8, we present two specialized extensions of SLIM.

### 1.1 Related work

Our work is related to several streams of research, namely: medical scoring systems; sparse linear classification; discrete linear classification; and mixed-integer programming (MIP) approaches for classification. In what follows, we discuss related work in each area separately.

#### *1.1.1 Medical scoring systems*

Some popular medical scoring systems include:

- SAPS I, II and III, to assess ICU mortality risk (Le Gall et al. 1984, 1993; Moreno et al. 2005);
- 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);
- Wells Criteria for pulmonary embolisms (Wells et al. 2000); Wells Criteria for and deep vein thrombosis (Wells et al. 1997);
- TIMI, to assess the risk of death and ischemic events (Antman et al. 2000);
- SIRS, to detect system inflammatory response syndrome (Bone et al. 1992);

All of the scoring systems listed above are sparse linear models with small coprime coefficients. The $CHADS_2$ scoring system, for instance, uses 5 coefficients with values of 1 and 2.

Many medical scoring systems were constructed without optimizing for predictive accuracy. In some cases, physicians built scoring systems by combining existing linear classification methods with heuristics. The SAPS II score, for instance, was constructed by

rounding logistic regression coefficients: as Le Gall et al. (1993) write, "the general rule was to multiply the $\beta$ 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 appears to have been the case for $CHADS_2$ as suggested by Gage et al. (2001): "To create $CHADS_2$, we assigned 2 points to a history of prior cerebral ischemia and 1 point for the presence of other risk factors because a history of prior cerebral ischemia increases the relative risk (RR) of subsequent stroke commensurate to 2 other risk factors combined. 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 easily produce highly tailored prediction models, such as SLIM, should eliminate the need for physicians to create models by hand.

In addition to the sleep apnea application that we present in Sect. 6, SLIM has also been used to create medical scoring system for diagnosing cognitive impairments using features derived from a clock-drawing test (Souillard-Mandar et al. 2015), and to create scoring systems for recidivism prediction (Zeng et al. 2015).

### *1.1.2 Sparse linear classification models*

In comparison to SLIM, the majority of current methods for sparse linear classification are designed to fit models with real coefficients and would therefore require rounding to yield scoring systems. In practice, rounding the coefficients of a linear model may significantly alter its accuracy and sparsity, and may produce a scoring system that violates operational constraints on these quantities. Further, many current methods also 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 Sects. 6 and 7, surrogate functions provide a poor trade-off between accuracy and sparsity. Convex surrogate loss functions, for instance, produce models that do not attain the best learning-theoretic guarantee on predictive accuracy and are not robust to outliers (Li and Lin 2007; Brooks and Lee 2010; Nguyen and Sanner 2013). Similarly, $\ell_1$-regularization is only guaranteed to recover the correct sparse solution (i.e., the one that minimizes the $\ell_0$-norm) under restrictive conditions that are rarely satisfied in practice (Zhao and Bin 2007; Liu and Zhang 2009). In fact, $\ell_1$-regularization may recover a solution that attains a significant loss in predictive accuracy relative to the correct sparse solution (see e.g., Lin et al. 2008, for a discussion). Sparse linear classifiers can also be produced using feature selection algorithms (Guyon and Elisseeff 2003; Mao 2004), though these algorithms cannot guarantee an optimal balance between accuracy and sparsity as they typically rely on greedy optimization (with some exceptions, see e.g., Bradley et al. 1999).

### *1.1.3 Discrete linear classification models*

SLIM is part of a recent stream of research on creating linear classifiers with discrete coefficients. Specifically, Chevaleyre et al. (2013) have considered training linear classifiers with binary coefficients by rounding the coefficients of linear classifiers that minimize the hinge loss. In addition, Carrizosa et al. (2013) have considered training linear classifiers with small integer coefficients by using a MIP formulation that optimizes the hinge loss. The discretization bounds and generalization bounds in Sect. 4 are a novel contribution to this body of work and applicable to all linear models with discrete coefficients.

SLIM can reproduce the models proposed by Chevaleyre et al. (2013) and Carrizosa et al. (2013) (see e.g., our formulation to create M-of-N rule tables in Sect. 8.2.1). The converse,

however, is not necessarily true because the methods of Chevaleyre et al. (2013) and Carrizosa et al. (2013) have the following weaknesses: (i) they optimize the hinge loss as opposed to the 0–1 loss; and (ii) they do not include a mechanism to control sparsity. These differences may result in better scalability compared to SLIM. However, they also eliminate the ability of these methods to produce scoring systems that are sparse, that satisfy operational constraints on accuracy and/or sparsity, and that can be trained without parameter tuning.

#### *1.1.4 MIP approaches for classification*

SLIM uses integer programming (IP) to achieve three distinct goals: (i) minimize the 0–1 loss; (ii) penalize the $\ell_0$-norm for feature selection; and (iii) restrict coefficients to a small set of integers. MIP approaches have been used to tackle each of these goals, albeit separately.

MIP formulations to minimize the 0–1 loss, for instance, were first proposed in Liittschwager and Wang (1978) and Bajgier and Hill (1982), and later refined by Mangasarian (1994), Asparoukhov and Stam (1997) and Glen (1999). Similarly, MIP formulations that penalize the $\ell_0$-norm for feature selection were proposed in Goldberg and Eckstein (2010), Goldberg and Eckstein (2012) and Nguyen and Franke (2012). To our knowledge, the only MIP formulation to restrict coefficients to a small set of integers is proposed in Carrizosa et al. (2013).

SLIM has unique practical benefits in comparison to these MIP formulations since it handles all three of these goals simultaneously. As we discuss in Sect. 2, the simultaneous approach allows SLIM to train models parameter tuning, and make use of the variables that encode the 0–1 loss and $\ell_0$-norm to accommodate important operational constraints. Further, restricting coefficients to a discrete set that is finite leads to an IP formulation whose LP relaxation is significantly tighter than other MIP formulations designed to minimize the 0–1 loss and/or penalize the $\ell_0$-norm.

The problem of finding a linear classifier that minimizes the 0–1 loss function is sometimes referred to as misclassification minimization for the linear discriminant problem in the MIP community (see e.g., Rubin 2009; Lee and Wu 2009, for an overview). Seeing how early attempts at misclassification minimization were only feasible for tiny datasets with at most 200 examples (see e.g., Joachimsthaler and Stam 1990), a large body of work has focused on improving scalability by using heuristics (Rubin 1990; Yanev and Balev 1999), decomposition procedures (Rubin 1997), cutting planes (Brooks 2011) and specialized branch-and-bound algorithms (Nguyen and Sanner 2013). The data reduction technique we present in Sect. 5 is a novel contribution to this body of work, and addresses a need for general methods to remove redundant data put forth by Bradley et al. (1999). We compare and contrast data reduction to existing approaches in greater detail in Sect. 5.3.

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

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

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

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

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

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 $\ell_0$-norm $\|\boldsymbol{\lambda}\|_0 := \sum_{j=1}^{P} \mathbb{1}\left[\lambda_j \neq 0\right]$ 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 $\|\boldsymbol{\lambda}\|_0 \leq 10$. SLIM includes a *tiny* $\ell_1$-penalty $\epsilon \|\boldsymbol{\lambda}\|_1$ in the objective for the sole purpose of restricting coefficients to coprime values. [1] To be clear, the $\ell_1$-penalty parameter $\epsilon$ is always set to a value that is small enough to avoid $\ell_1$-regularization (that is, $\epsilon$ is small enough to guarantee that SLIM never sacrifices accuracy or sparsity to attain a smaller $\ell_1$-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 $\ell_0$-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 $\ell_0$-regularization prevents the additional loss in accuracy due to $\ell_1$-regularization (see Lin et al. 2008, for a discussion). We can make further use of the variables that encode the 0–1 loss and $\ell_0$-penalty to formulate operational constraints related to accuracy and sparsity (see Sect. 3).

One unique benefit in minimizing an approximation-free objective function over a finite set of discrete coefficients is that the free parameters in the objective of (2) have special properties.

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

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

[1] To illustrate the use of the $\ell_1$-penalty, consider a classifier such as $\hat{y} = \text{sign}(x_1 + x_2)$. If the objective in (2) only minimized the 0–1 loss and an $\ell_0$-penalty, then $\hat{y} = \text{sign}(2x_1 + 2x_2)$ would have the same objective value as $\hat{y} = \text{sign}(x_1 + x_2)$ 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* $\ell_1$-penalty in the objective of (2) so that SLIM chooses the classifier with the smallest (i.e. coprime) coefficients, $\hat{y} = \text{sign}(x_1 + x_2)$.

and,

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

*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 $\boldsymbol{\lambda} \in \mathcal{L}$ that attains the highest possible training accuracy. That is,

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

*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 $\boldsymbol{\lambda} \in \mathcal{L}$ that attains the highest possible sparsity. That is,

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

The aforementioned properties are only possible using the formulation in (2). In particular, the properties in Remarks 2–4 require that we control accuracy using the 0–1 loss and control sparsity using an $\ell_0$-penalty. In addition, the property in Remark 1 requires that we restrict coefficients to a finite set of discrete values.

### 2.1 SLIM IP Formulation

We train SLIM scoring systems using the following IP formulation:

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

Here, the constraints in (3a) set the loss variables $\psi_i = \mathbb{1}\left[y_i\boldsymbol{\lambda}^T\boldsymbol{x}_i \le 0\right]$ 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 $\gamma$ 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 $\boldsymbol{\lambda}$ is restricted to a finite discrete set. The value of $\gamma$ 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, $\gamma$ 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 $\alpha_j := \mathbb{1}\left[\lambda_j \neq 0\right]$ 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 discrete 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 $\ell_0$-norm. Many IP formulations compute the 0–1 loss and $\ell_0$-norm by means of Big-M constraints that use on Big-M constants (see e.g., Wolsey 1998). Restricting the coefficients to a finite discrete set allows us to bound Big-M constants that are typically set heuristically to a large value. Specifically, the Big-M constant for computing the 0–1 loss in constraints (3a) is bounded as $M_i \leq \max_{\boldsymbol{\lambda} \in \mathcal{L}}(\gamma - y_i \boldsymbol{\lambda}^T \boldsymbol{x}_i)$ (compare with Brooks 2011, where the same parameter has to be approximated by a "sufficiently large constant"). Similarly, the Big-M constant used to compute the $\ell_0$-norm in constraints (3c) is bounded at $\Lambda_j \leq \max_{\lambda_j \in \mathcal{L}_j} |\lambda_j|$ (compare with Guan et al. 2009, where this same parameter has to be approximated by a "sufficiently large constant"). These differences lead to a tighter LP relaxation, which narrows the integrality gap, and subsequently improves the ability of commercial IP solvers to obtain a proof of optimality.

## 3 Operational constraints

In this section, we discuss how SLIM can accommodate a wide range of operational constraints related to the accuracy and sparsity of predictive models. The following techniques provide users with a practical approach to create tailored prediction models. They are made possible by the facts that: (i) the variables that encode the 0–1 loss and $\ell_0$-penalty in the SLIM IP formulation can also be used to handle accuracy and sparsity; and (ii) the free parameters in the SLIM objective can be set without tuning (see Remarks 2–4).

### 3.1 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 unique benefits when training scoring systems on imbalanced problems. Specifically, it is the only method that can train a supervised classification model at any point on the ROC curve without any parameter tuning. When physicians specify hard constraints on sensitivity (or specificity), we can encode these as *loss constraints* into the IP formulation, and solve the IP to obtain the least specific (or most sensitive) model without parameter tuning. 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:

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

Here, we have used a *weighted* 0–1 loss function where $W^+$ and $W^-$ are weights that control the accuracy on the $N^+$ positively-labeled examples from the set $\mathcal{I}^+ = \{i : y_i = +1\}$, and $N^-$ negatively-labeled examples from the set $\mathcal{I}^- = \{i : y_i = -1\}$, respectively. Assuming that $W^+ + W^- = 1$, we can set $W^+ > \frac{N^-}{1+N^-}$ and $W^- = 1 - W^+$ so that the optimization aims to find a scoring system that classifies all of the positively-labeled examples correctly, at the expense of misclassifying all of the negatively-labeled examples. Constraint (4) prevents this from happening by limiting the error on negatively-labeled examples to $\gamma$. Thus, the optimal scoring system attains the highest sensitivity among models with a maximum error of $\gamma$ on negatively-labeled examples.

### 3.2 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 make use of the indicator variables that encode the $\ell_0$-norm $\alpha_j := \mathbb{1}\left[\lambda_j \neq 0\right]$ to formulate arbitrarily complicated logical constraints between features such as "either-or" conditions or "if-then" conditions (see e.g., Wolsey 1998). This presents a practical alternative to create classification models that obey structured sparsity constraints (see e.g., Jenatton et al. 2011) or hierarchical constraints (see e.g., Bien et al. 2013).

The indicator variables $\alpha_j$ can be used to limit the number of input variables to at most $\Theta$ by adding the constraint,

$$\sum_{j=1}^{P} \alpha_j \leq \Theta.$$

More complicated constraints include, for example, "if-then" constraints to ensure that a model will 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:

$$\alpha_{leaf} \leq \alpha_{node} \text{ for all nodes above the leaf.}$$

### 3.3 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.

## 4 Bounds on training and testing accuracy

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

### 4.1 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 $\Lambda$ 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. □

The proof of Theorem 1 uses a rounding procedure to choose a resolution parameter $\Lambda$ 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 $\Lambda$ 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 $\Lambda$ 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.

### 4.2 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) *Let $\mathcal{F}$ denote the set of linear classifiers with coefficients $\boldsymbol{\lambda} \in \mathcal{L}$:*

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

*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 exploiting the fact that we can bound the number of non-zero coefficients in a SLIM scoring system in terms of 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:*

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

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

*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*

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

*Proof* See Appendix. □

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 $\Lambda$ (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 $\Lambda$:*

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

$$\mathcal{L} = \left\{ \boldsymbol{\lambda} \in \hat{\mathbb{Z}}^P \;\middle|\; |\lambda_j| \leq \Lambda \quad \text{for} \quad j = 1, \ldots, P \right\},$$

$$\hat{\mathbb{Z}}^P = \left\{ \boldsymbol{z} \in \mathbb{Z}^P \;\middle|\; gcd(\boldsymbol{z}) = 1 \right\}.$$

*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 $\Lambda$:*

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

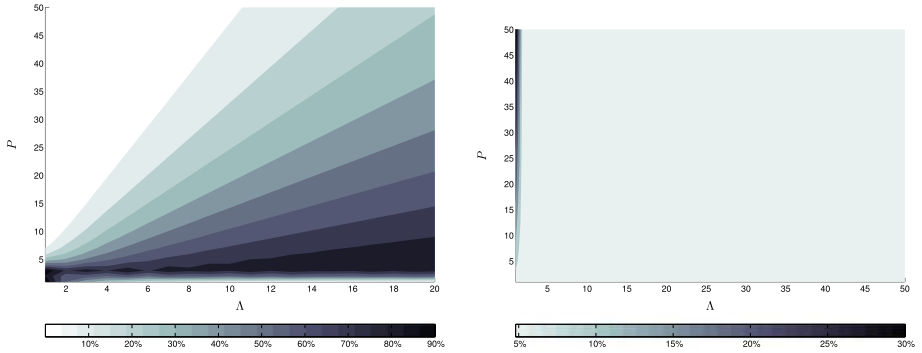



**Fig. 1** Relative density of coprime integer vectors in $\mathbb{Z}^P$ (*left*), and the relative improvement in the generalization bound due to the use of coprime coefficients for $\delta = 0.01$ (*right*)

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

In Fig. 1, we plot the relative density of coprime integer vectors in $\mathbb{Z}^P$ bounded by $\Lambda$ (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. This shows that using coprime coefficients can significantly reduce the number of classifiers based on the dimensionality of the data and the value of $\Lambda$. The corresponding improvement in the generalization bound may be significant when the data are high dimensional and $\Lambda$ is small.

## 5 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.

### 5.1 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 $\varepsilon$-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 $\varepsilon$ such that

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

Here, $f^*$ denotes an optimal classifier to the original problem from the set $\mathcal{F}^* = \text{argmin}_{f \in \mathcal{F}} Z(f)$, and $\tilde{f}^*$ denotes an optimal classifier to the surrogate problem from the set $\tilde{\mathcal{F}}^* = \text{argmin}_{f \in \tilde{\mathcal{F}}} \tilde{Z}(f)$. The width of the the surrogate level set $\varepsilon$ is related to the amount of data that will be removed. If $\varepsilon$ is too large, the method will not remove very many examples and will be less helpful for reducing computation (see Fig. 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 $\tilde{y}_i := \text{sign}\left(\tilde{f}^*(\boldsymbol{x}_i)\right)$ 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 example $i$ as $-\tilde{y}_i$. In other words, all classifiers in the surrogate level set must label this example 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 Fig. 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).

**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 $\mathcal{D}_N$,*

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

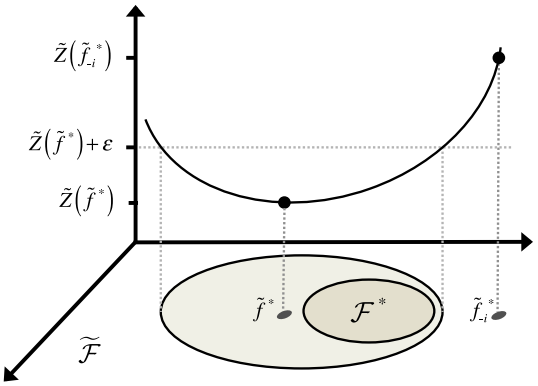



**Fig. 2** We initialize data reduction with $\varepsilon$ large enough so that $\tilde{Z}(f^*) < \tilde{Z}(\tilde{f}^*) + \varepsilon$ for all $f^* \in \mathcal{F}^*$ and all $\tilde{f}^* \in \tilde{\mathcal{F}}^*$. Here, $f^*$ is the optimal classifer to the original problem from the set of optimal 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 $\mathcal{D}_M$

**Require:** initial training data; $\mathcal{D}_N = (\boldsymbol{x}_i, y_i)_{i=1}^N$
**Require:** surrogate problem, $\min \tilde{Z}(f; \mathcal{D}_N)$ s.t. $f \in \tilde{\mathcal{F}}$
**Require:** width of the surrogate level set, $\varepsilon$
  **Initialize:** $\mathcal{D}_M \longleftarrow \emptyset$
  $\tilde{f}^* \longleftarrow \operatorname{argmin}_f \tilde{Z}(f; \mathcal{D}_N)$
  **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**
**Ensure:** $\mathcal{D}_M$, reduced training data

*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* $\varepsilon$ *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.

**Theorem 6** (Sufficient conditions to satisfy the level set condition) *Consider an optimization problem where the objective minimizes the 0–1 loss function* $Z_{01} : \mathbb{R}^P \to \mathbb{R}$,

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

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

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

*If the surrogate loss function* $\psi$ *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 $\boldsymbol{\lambda}_{01}^*$: $\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_\psi^*\| < A \implies Z_\psi(\boldsymbol{\lambda}) - Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) < L\|\boldsymbol{\lambda} - \boldsymbol{\lambda}_\psi^*\|$

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

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

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

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

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

*Proof* See Appendix. □

### 5.2 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.

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 $\varepsilon$ 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 Fig. 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 $\boldsymbol{\lambda} = \mathbf{0}$):

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

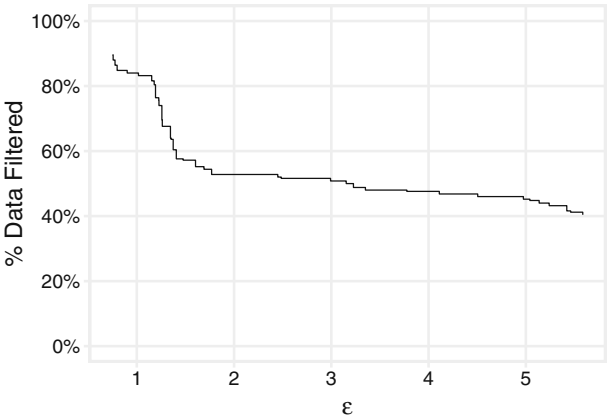



**Fig. 3** Proportion of training data filtered as a function of the width of the level set, $\varepsilon$ for the `bankruptcy` dataset. Here, the original problem is an instance of the SLIM IP with $C_0 = 0.01$ and $\mathcal{L} = \{-10, \ldots, 10\}^{P+1}$

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

### 5.3 Discussion and related work

Data reduction is a technique that can be applied to a wide range of supervised classification methods, including methods that minimize the 0–1 loss function.

Data reduction is fundamentally different from many techniques for improving the scalability of 0–1 loss minimization, such as oscillation heuristics (Rubin 1990), decomposition (Rubin 1997), cutting planes (Brooks 2011) and specialized branch-and-bound algorithms (Nguyen and Sanner 2013). In fact, data reduction is most similar to the scenario reduction methods in the stochastic programming literature (see e.g., Dupačová et al. 2000, 2003). In comparison to scenario reduction techniques, data reduction does not require the objective function to satisfy stability or convexity assumptions, and is designed to recover the true optimal solution as opposed to an $\varepsilon$-optimal solution.

Data reduction has the advantage that it easily be applied to SLIM by using SLIM's LP relaxation as the surrogate problem. This 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. Off-the-shelf data reduction makes use of the integrality gap to identify examples that have to be classified a certain way. Accordingly, the effectiveness of this approach may be improved using techniques that narrow the integrality gap—specifically by using higher quality feasible solutions and/or strengthening SLIM's LP relaxation (e.g., by using the cutting planes of Brooks 2011).

## 6 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). Our goal is to highlight the flexibility and performance of our approach on a real-world problem that requires a tailored prediction model.

### 6.1 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 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).

### 6.2 Training setup and model selection

We trained a SLIM scoring system with integer coefficients between $-10$ and 10. We addressed all 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 Sect. 3.1).
- We added a feature-based constraint using the loss variables to limit the maximum number of features to 5 (see Sect. 3.2). 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 tenfold cross validation (10-CV), and 1 final model with all of data to hand over to our collaborators. We solved each IP for 1 hour, in parallel, on 12-core 2.7 GHz 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 accommodate 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

**Table 1** Training setup for all methods

| Method | Controls | # Instances | Settings and free parameters |
|---|---|---|---|
| CART | Max FPR<br>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<br>Model Size | 39 | 39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}$ |
| Lasso | Max FPR<br>Model Size<br>Signs | 39,000 | 39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}\times$ 1000 values of $\lambda$ chosen by **glmnet** |
| Ridge | Max FPR<br>Signs | 39,000 | 39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}\times$ 1000 values of $\lambda$ chosen by **glmnet** |
| Elastic Net | Max FPR<br>Model Size<br>Signs | 975,000 | 39 values of $W^+ \in \{0.025, 0.05, \ldots, 0.975\}\times$ 1000 values of $\lambda$ 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<br>Model Size<br>Signs | 1 | $W^+ = N^-/(1+N^-)$, $C_0 = 0.9W^-/NP$, $\lambda_0 \in \{-100, \ldots, 100\}$, $\lambda_j \in \{-10, \ldots, 10\}$ |

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

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.

### 6.3 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 from each method in Table 2, and summarize the operational constraints they satisfied 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.

**Table 2** TPR, FPR and model size for all methods

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

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

**Table 3** Percentage of instances that fulfilled operational constraints

| Method | % of Instances | | |
|---|---|---|---|
| | 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 |

Each instance is a unique combination of free parameters

### *6.3.1 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 Fig. 4). Methods that used $\ell_2$-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 Fig. 4).
- Allow tuning to be portable when the training set changes. Consider a standard K-CV model selection procedure where we choose free parameters to maximize predictive accuracy. To adapt this procedure on a problem with operational constraints, we would train models on K validation folds for each instance of the free parameters, choose an instance of the free parameters that maximizes the mean K-CV test accuracy among the instances that satisfied all operational constraints, and train a "final model" for this instance. Unfortunately, there is no guarantee that the final model will obey all operational constraints.

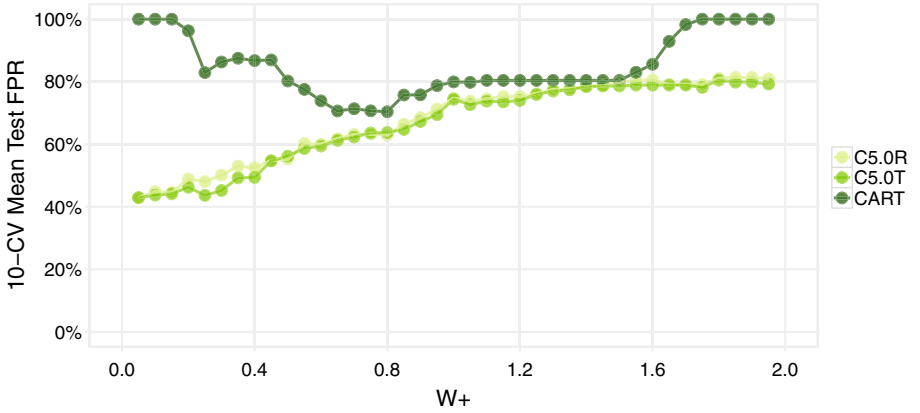
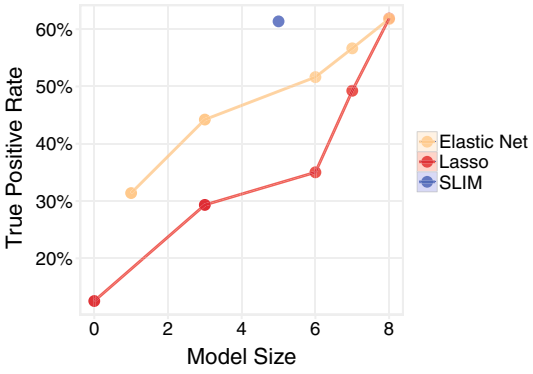



**Fig. 4** 10-CV mean test FPR for models trained with CART, C5.0, C5.0T across the full range of $W^+$. These methods cannot produce a model that satisfies the max FPR $\leq 20\,\%$ constraint

### *6.3.2 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 linear models produced by Lasso and Elastic Net—a result that we expected given that SLIM minimizes the 0–1 loss and an $\ell_0$-penalty while Lasso and Elastic Net minimize convex surrogates of these quantities. This result held true even when we relaxed various operational constraints. In Fig. 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 Fig. 6, we plot the TPR and FPR of models 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).

**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



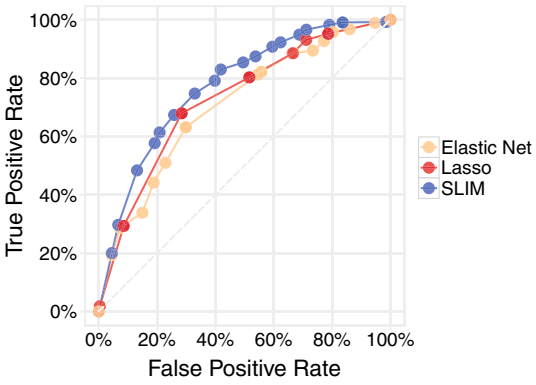



**Fig. 6** ROC curve for SLIM, Lasso and Elastic Net instances that satisfy the sign and model size constraints. For each method, we plot the instance that attains the highest 10-CV mean test TPR for 10-CV mean FPR values of 5, 10, . . . , 95 %. Note that we had to train 19 additional instances of SLIM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

### *6.3.3 On the usability and interpretability of acceptable models*

We include a head-to-head comparison between the most sensitive models that satisfied all operational constraints in Fig. 7 and show the SLIM model as a scoring system in Fig. 8.

Our collaborators commented that all three models were, in theory, usable by physicians because they were aligned with domain knowledge. Specifically, all models obeyed sign constraints and had large coefficients for well-known risk factors for sleep apnea such as *bmi*, *female*, *age*, *snoring* and/or *hypertension*. Unfortunately, the Lasso and Elastic Net models could not realistically be used 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 provide this kind of qualitative

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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. | $body\ mass\ index \geq 30$ | 2 points | + | ····· |
| 4. | $body\ mass\ index \geq 40$ | 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 %

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 ($7 \pm 2$ according to Miller 1984), and are seriously limited in estimating the association between three or more variables (Jennings et al. 1982). Accordingly, these qualities may also help users gauge the influence 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)."

## 7 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.

### 7.1 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

**Table 4** Datasets used in the numerical experiments

| 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 5** Training setup for classification methods used for the numerical experiments

| 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. + $\ell_1$ penalty | Lasso | **glmnet** (Friedman et al. 2010) | 1000 values of $\lambda$ chosen by **glmnet** |
| Log. Reg. + $\ell_2$ penalty | Ridge | **glmnet** (Friedman et al. 2010) | 1000 values of $\lambda$ chosen by **glmnet** |
| Log. Reg. + $\ell_1/\ell_2$ penalty | Elastic Net | **glmnet** (Friedman et al. 2010) | 1000 values of $\lambda$ 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 | **CPLEX 12.6.0.0** | 6 values of $C_0 \in \{0.01, 0.075, 0.05, 0.025, 0.001, 0.9/NP\}$ with $\lambda_j \in \{-10, \ldots, 10\}$; $\lambda_0 \in \{-100, \ldots, 100\}$ |

features. For the purposes of reproducibility, we include all processed datasets in Online Resource 1.

*Methods*: We summarize the training setup for each method in Table 5. We trained SLIM scoring systems using the CPLEX 12.6.0.0 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 tenfold 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 $\ell_0$-regularization path for SLIM by solving $6 \times 11$ IPs for each dataset (6 values of $C_0 \times 11$ training runs per $C_0$). We allocated at most 10 min 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 $W^- = N^+/N$.

## 7.2 Results and observations

We summarize the results of our experiments in Table 6 and Figs. 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

**Table 6** Accuracy and sparsity of all methods on all datasets

| Dataset | Details | | Metric | SLIM | Lasso | Ridge | Elastic Net | C5.0R | C5.0T | CART | SVM Lin. | SVM RBF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| adult | $N$ | 32561 | Test Error | $17.4 \pm 1.4\,\%$ | $17.3 \pm 0.9\,\%$ | $17.6 \pm 0.9\,\%$ | $17.4 \pm 0.9\,\%$ | $26.4 \pm 1.8\,\%$ | $26.3 \pm 1.4\,\%$ | $75.9 \pm 0.0\,\%$ | $16.8 \pm 0.8\,\%$ | $16.3 \pm 0.5\,\%$ |
| | $P$ | 37 | Train Error | $17.5 \pm 1.2\,\%$ | $17.2 \pm 0.1\,\%$ | $17.6 \pm 0.1\,\%$ | $17.4 \pm 0.1\,\%$ | $25.3 \pm 0.4\,\%$ | $24.9 \pm 0.4\,\%$ | $75.9 \pm 0.0\,\%$ | $16.7 \pm 0.1\,\%$ | $16.3 \pm 0.1\,\%$ |
| | $\Pr(y=+1)$ | 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 | $3.4 \pm 2.0\,\%$ | $3.4 \pm 2.2\,\%$ | $3.4 \pm 2.0\,\%$ | $3.1 \pm 2.1\,\%$ | $4.3 \pm 3.3\,\%$ | $5.3 \pm 3.4\,\%$ | $5.6 \pm 1.9\,\%$ | $3.1 \pm 2.0\,\%$ | $3.5 \pm 2.5\,\%$ |
| | $P$ | 10 | Train Error | $3.2 \pm 0.2\,\%$ | $2.9 \pm 0.3\,\%$ | $3.0 \pm 0.3\,\%$ | $2.8 \pm 0.3\,\%$ | $2.1 \pm 0.3\,\%$ | $1.6 \pm 0.4\,\%$ | $3.6 \pm 0.3\,\%$ | $2.7 \pm 0.2\,\%$ | $0.3 \pm 0.1\,\%$ |
| | $\Pr(y=+1)$ | 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 | $0.8 \pm 1.7\,\%$ | $0.0 \pm 0.0\,\%$ | $0.4 \pm 1.3\,\%$ | $0.0 \pm 0.0\,\%$ | $0.8 \pm 1.7\,\%$ | $0.8 \pm 1.7\,\%$ | $1.6 \pm 2.8\,\%$ | $0.4 \pm 1.3\,\%$ | $0.4 \pm 1.3\,\%$ |
| | $P$ | 7 | Train Error | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $0.4 \pm 0.1\,\%$ | $0.4 \pm 0.7\,\%$ | $0.4 \pm 0.2\,\%$ | $0.4 \pm 0.2\,\%$ | $1.6 \pm 0.3\,\%$ | $0.4 \pm 0.1\,\%$ | $0.4 \pm 0.1\,\%$ |
| | $\Pr(y=+1)$ | 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 | $29.2 \pm 14.0\,\%$ | $42.5 \pm 11.3\,\%$ | $36.9 \pm 15.0\,\%$ | $40.9 \pm 14.0\,\%$ | $42.7 \pm 9.4\,\%$ | $42.7 \pm 9.4\,\%$ | $43.1 \pm 8.0\,\%$ | $45.3 \pm 14.7\,\%$ | $47.5 \pm 6.2\,\%$ |
| | $P$ | 4 | Train Error | $29.7 \pm 1.5\,\%$ | $40.6 \pm 1.9\,\%$ | $41.0 \pm 9.7\,\%$ | $45.1 \pm 12.0\,\%$ | $40.4 \pm 8.5\,\%$ | $40.4 \pm 8.5\,\%$ | $34.3 \pm 2.8\,\%$ | $46.0 \pm 3.6\,\%$ | $5.4 \pm 1.5\,\%$ |
| | $\Pr(y=+1)$ | 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 |

**Table 6** continued

| Dataset | Details | | Metric | SLIM | Lasso | Ridge | Elastic Net | C5.0R | C5.0T | CART | SVM Lin. | SVM RBF |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| mammo | $N$ | 961 | Test Error | $19.5 \pm 3.0\,\%$ | $19.0 \pm 3.1\,\%$ | $19.2 \pm 3.0\,\%$ | $19.0 \pm 3.1\,\%$ | $20.5 \pm 3.3\,\%$ | $20.3 \pm 3.5\,\%$ | $20.7 \pm 3.9\,\%$ | $20.3 \pm 3.0\,\%$ | $19.1 \pm 3.1\,\%$ |
| | $P$ | 15 | Train Error | $18.3 \pm 0.3\,\%$ | $19.3 \pm 0.3\,\%$ | $19.2 \pm 0.4\,\%$ | $19.2 \pm 0.3\,\%$ | $19.8 \pm 0.3\,\%$ | $19.9 \pm 0.3\,\%$ | $20.0 \pm 0.6\,\%$ | $20.3 \pm 0.4\,\%$ | $18.2 \pm 0.4\,\%$ |
| | $\Pr(y=+1)$ | 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 | $16.5 \pm 7.8\,\%$ | $15.2 \pm 6.3\,\%$ | $14.9 \pm 5.9\,\%$ | $14.5 \pm 5.9\,\%$ | $21.2 \pm 7.5\,\%$ | $23.2 \pm 6.8\,\%$ | $19.8 \pm 6.5\,\%$ | $15.5 \pm 6.5\,\%$ | $15.2 \pm 6.0\,\%$ |
| | $P$ | 33 | Train Error | $14.9 \pm 1.1\,\%$ | $14.0 \pm 1.0\,\%$ | $13.1 \pm 0.8\,\%$ | $13.2 \pm 0.6\,\%$ | $10.0 \pm 1.8\,\%$ | $8.5 \pm 2.0\,\%$ | $14.3 \pm 0.9\,\%$ | $13.6 \pm 0.5\,\%$ | $10.4 \pm 0.8\,\%$ |
| | $\Pr(y=+1)$ | 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 | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $1.7 \pm 0.3\,\%$ | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $1.2 \pm 0.6\,\%$ | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ |
| | $P$ | 114 | Train Error | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $1.7 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ | $1.1 \pm 0.3\,\%$ | $0.0 \pm 0.0\,\%$ | $0.0 \pm 0.0\,\%$ |
| | $\Pr(y=+1)$ | 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 | $6.3 \pm 1.2\,\%$ | $10.0 \pm 1.7\,\%$ | $26.3 \pm 1.7\,\%$ | $10.0 \pm 1.7\,\%$ | $6.6 \pm 1.3\,\%$ | $7.3 \pm 1.0\,\%$ | $11.1 \pm 1.4\,\%$ | $7.8 \pm 1.5\,\%$ | $13.7 \pm 1.4\,\%$ |
| | $P$ | 58 | Train Error | $5.7 \pm 0.3\,\%$ | $9.5 \pm 0.3\,\%$ | $26.1 \pm 0.2\,\%$ | $9.6 \pm 0.2\,\%$ | $4.2 \pm 0.3\,\%$ | $3.9 \pm 0.3\,\%$ | $9.8 \pm 0.3\,\%$ | $8.1 \pm 0.8\,\%$ | $1.3 \pm 0.1\,\%$ |
| | $\Pr(y=+1)$ | 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 |

Here *Test Error* refers to the 10-CV mean test error $\pm$ the 10-CV standard deviation in test error; *Train Error* refers to the 10-CV mean training error $\pm$ 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`

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.

We show the accuracy and sparsity of all methods on all dataset in Figs. 13 and 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 $\ell_0$-regularization paths for SLIM and Lasso on the right side of Figs. 13 and 14 to show how the test error varies at different levels of sparsity for sparse linear models.

#### *7.2.1 On 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 Figs. 13 and 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 the optimization problems we solved 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 relative MIP-GAP of 0.0 %) for all scoring systems we trained for `mammo`, `mushroom`, `bankruptcy`, `breastcancer`.

#### *7.2.2 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.).

#### *7.2.3 On interpretability*

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.

**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 $0.0 \pm 0.0$ %

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 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 $0.0 \pm 0.0$ %

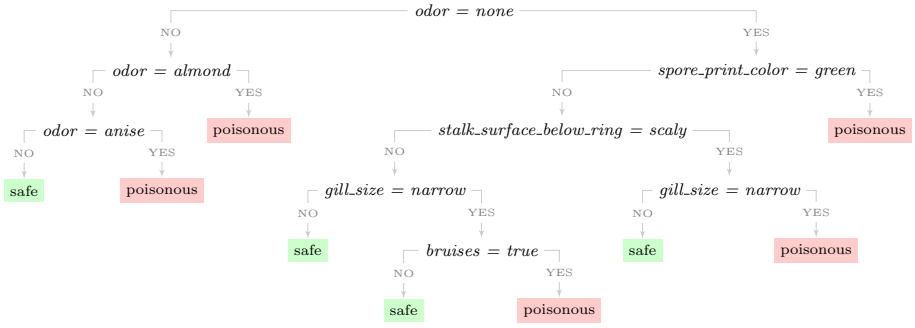



**Fig. 11** C5.0 decision tree for `mushroom`. This model has a 10-CV mean test error of $0.0 \pm 0.0$ %

| Rule | | | Confidence | Support | Lift |
|---|---|---|---|---|---|
| $odor = none \ \wedge \ gill\_size \neq narrow \ \wedge \ spore\_print\_color \neq green$ | $\Longrightarrow$ | safe | 1.000 | 3216 | 1.9 |
| $bruises = false \ \wedge \ odor = none \ \wedge \ stalk\_surface\_below\_ring \neq scaly$ | $\Longrightarrow$ | safe | 0.999 | 1440 | 1.9 |
| $odor = almond$ | $\Longrightarrow$ | safe | 0.998 | 400 | 1.9 |
| $odor = anise$ | $\Longrightarrow$ | safe | 0.998 | 400 | 1.9 |
| $odor \neq almond \ \wedge \ odor \neq anise \ \wedge \ odor \neq none$ | $\Longrightarrow$ | poisonous | 1.000 | 3796 | 2.1 |
| $spore\_print\_color = green$ | $\Longrightarrow$ | poisonous | 0.986 | 72 | 2.9 |
| $gill\_size = narrow \ \wedge \ stalk\_surface\_below\_ring = scaly$ | $\Longrightarrow$ | poisonous | 0.976 | 40 | 2.0 |

**Fig. 12** C5.0 rule list for `mushroom`. This model has a 10-CV mean test error of $0.0 \pm 0.0$ %

In Figs. 9, 10, 11 and 12, we show the sparsest models that achieve perfect predictive accuracy on the `mushroom` dataset. We omit models from some methods because they do not attain perfect accuracy (CART) or use far more features (Ridge, SVM Lin, SVM RBF) (Figs. 13, 14).

In this case, the SLIM scoring system uses 7 integer coefficients. However, it can be expressed as a 5 line scoring system due to the fact that *odor=none*, *odor=almond*, and *odor=anise* are mutually exclusive variables that all use the same coefficient. The model benefits from the fact that it not only allows users make predictions by hand but uses a linear form that helps users to gauge the influence of each input variable with respect to the others. We note that only some of these qualities are found in other models. The Lasso model,

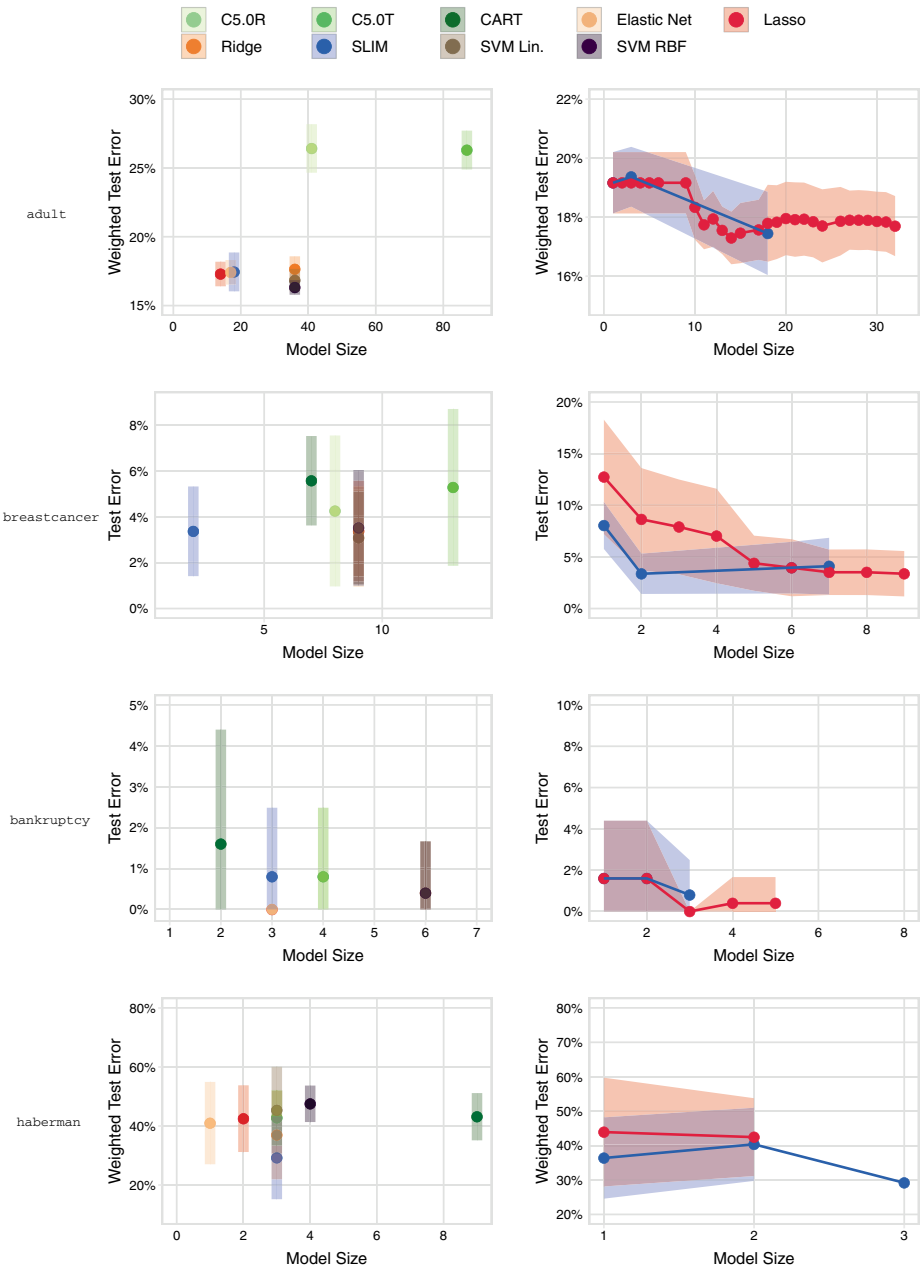



**Fig. 13** Accuracy and sparsity of all classification methods on all datasets. For each dataset, we plot the performance of models when free parameters are set to values that minimize the 10-CV mean test error (*left*), and plot the performance of SLIM and Lasso across the full $\ell_0$-regularization path (*right*)

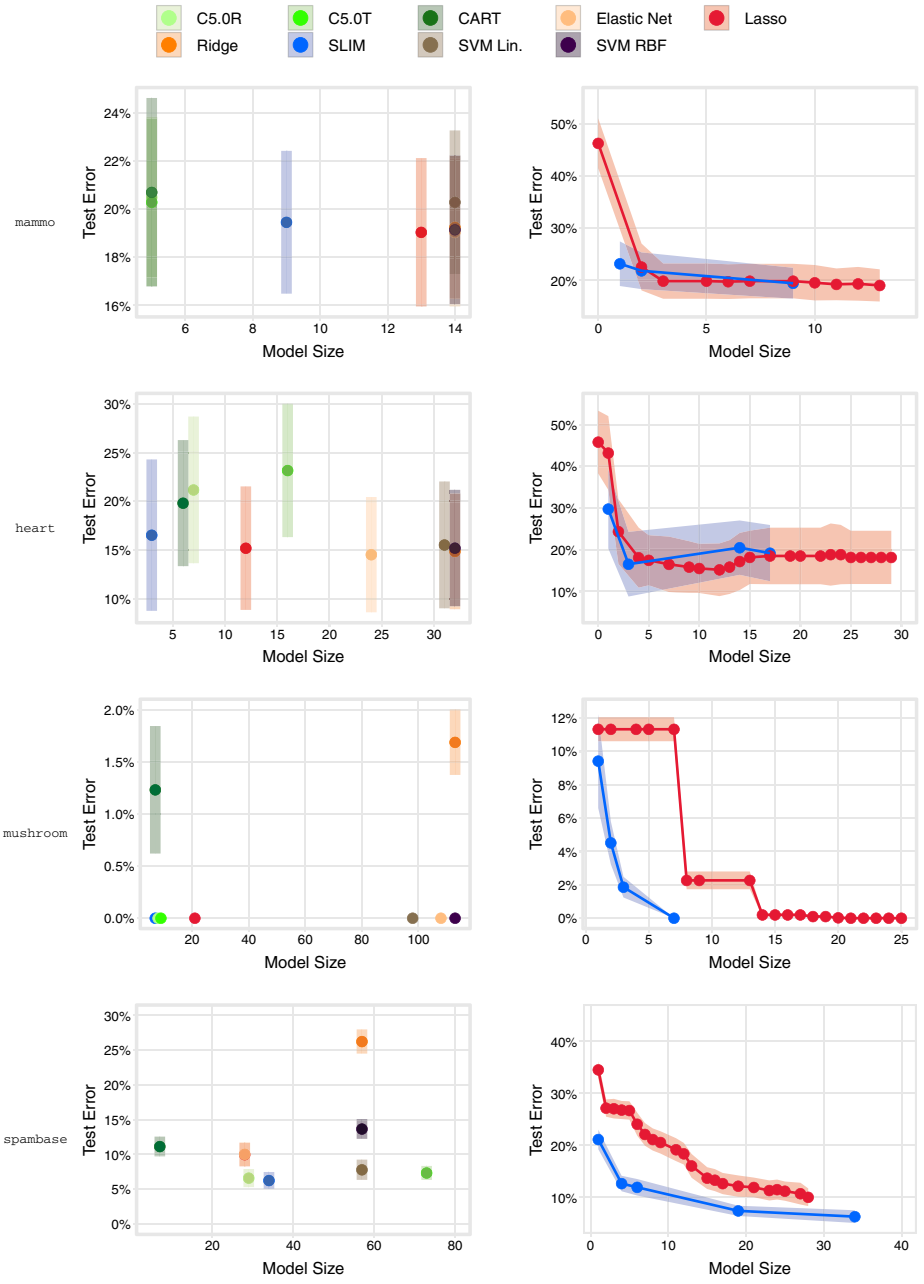



**Fig. 14** Accuracy and sparsity of all classification methods on all datasets. For each dataset, we plot the performance of models when free parameters are set to values that minimize the 10-CV mean test error (*left*) and plot the performance of SLIM and Lasso across the full $\ell_0$-regularization path (*right*)

for instance, has a linear form but uses far more features. Similarly, the C5.0 models allow users to make predictions by hand, but use a hierarchical structure that makes it difficult to assess the influence of each input variable with respect to the others (see Freitas 2014, for a discussion).

We believe that these qualities represent "baseline" interpretability benefits of SLIM scoring systems. In practice, interpretability is a subjective and multifaceted notion (i.e., it depends on who will be using the model but also depends on multiple model qualities as discussed by Kodratoff 1994; Pazzani 2000; Freitas 2014). In light of this fact, SLIM has the unique benefit in that it allows practitioners to work closely with the target audience and directly encode all interpretability requirements by means of operational constraints.

## 8 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).

### 8.1 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,

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

as well as a "personalized" interpretability penalty,

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

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 $C_0 < \cdots < C_R$. 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:

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

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

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

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

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

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

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

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

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

### 8.2 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:

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

**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 $h_{i,j} := \mathbb{1}\left[x_{i,j} \geq 3\right]$

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 $\boldsymbol{h}_{j,t} := \mathbb{1}\left[\boldsymbol{x}_j = k\right]$ 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 $\boldsymbol{h}_{j,t} := \mathbb{1}\left[\boldsymbol{x}_j \geq v_{j,t}\right]$ where $v_{j,t}$ denotes the $t$th threshold for feature $j$.

### *8.2.1 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., Fig. 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 2013) 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:

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

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 $\ell_0$-regularization using an $\ell_1$-penalty since $\left\|\lambda_{j,t}\right\|_0 = \left\|\lambda_{j,t}\right\|_1$ 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.

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

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

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

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

Here, the loss constraints and Big-M parameters in (13a) are identical to those from the SLIM IP formulation in Sect. 2. Constraints (13b) define the penalty variables $\Phi_{j,t}$ as the value of the $\ell_0$-penalty.

### 8.2.2 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), 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:

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

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 $\ell_1$-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 $\epsilon$ 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:

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

$$\text{s.t.} \quad M_i\psi_i \geq \gamma - \sum_{j=0}^{P}\sum_{t=1}^{T_j} y_i\lambda_{j,t}h_{i,j,t} \qquad i = 1,...,N \quad \textit{0-1 loss} \tag{14a}$$

$$\Phi_j = C_f\nu_j + C_t\tau_j + \epsilon\sum_{t=1}^{T_j}\beta_{j,t} \qquad j = 1,...,P \quad \textit{int. penalty} \tag{14b}$$

$$T_j\nu_j = \sum_{t=1}^{T_j}\alpha_{j,t} \qquad j = 1,...,P \quad \textit{feature use} \tag{14c}$$

$$\tau_j = \sum_{t=1}^{T_j}\alpha_{j,t} - 1 \qquad j = 1,...,P \quad \textit{threshold/feature} \tag{14d}$$

$$\tau_j \leq R_{max} + 1 \qquad j = 1,...,P \quad \textit{max thresholds}$$

$$-\Lambda_j\alpha_{j,t} \leq \lambda_{j,t} \leq \Lambda_j\alpha_{j,t} \qquad j = 1,...,P \quad t = 1,...,T_j \quad \ell_0 \textit{ norm}$$

$$-\beta_{j,t} \leq \lambda_{j,t} \leq \beta_{j,t} \qquad j = 1,...,P \quad t = 1,...,T_j \quad \ell_1 \textit{ norm} \tag{14e}$$

$$-\Lambda_j(1-\delta_j) \leq \lambda_{j,t} \leq \Lambda_j\delta_j \qquad j = 1,...,P \quad t = 1,...,T_j \quad \textit{agree in sign} \tag{14f}$$

$$\lambda_{j,t} \in \mathcal{L}_j \qquad j = 0,...,P \quad t = 1,...,T_j \quad \textit{coefficient values}$$

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

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

$$\alpha_j \in \{0,1\} \qquad j = 1,...,P \quad \ell_0 \textit{ variables}$$

$$\beta_j \in \mathbb{R}_+ \qquad j = 1,...,P \quad \ell_1 \textit{ variables}$$

$$\nu_j \in \{0,1\} \qquad j = 1,...,P \quad \textit{feature use indicators}$$

$$\tau_j \in \mathbb{Z}_+ \qquad j = 1,...,P \quad \textit{threshold/feature variables}$$

$$\delta_j \in \{0,1\} \qquad j = 1,...,P \quad \textit{sign indicators} \tag{14g}$$

Here, the loss constraints and Big-M parameters in (14a) are identical to those from the SLIM IP formulation in Sect. 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: $\nu_j$, which indicate that we use at least one threshold rule from feature $j$; $\tau_j$, which count the number of additional binary rules derived from feature $j$; and $\beta_{j,t} := |\lambda_{j,t}|$. The values of $\nu_j$ and $\tau_j$ are set using the indicator variables $\alpha_{j,t} := \mathbb{1}\left[\lambda_{j,t} \neq 0\right]$ in constraints (14c) and (14d). Constraints (14e) limit the number of binary rules from feature $j$ to $\mathbb{R}_{max}$. Constraints (14g) ensure that the coefficients of binary rules derived from feature $j$ agree in sign; these constraints are encoded using the variables $\delta_j := \mathbb{1}\left[\lambda_{j,t} \geq 0\right]$.

## 9 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

accommodate 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 $\ell_1$-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 Sect. 5. We gratefully acknowledge support from Siemens and Wistron.

## 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 $\Lambda$, we then define $\boldsymbol{\lambda}/\Lambda$ element-wise so that $\lambda_j/\Lambda$ 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 $\Lambda$ 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 $i$:

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

$$< \frac{\sqrt{P} X_{\max}}{2\left(\frac{X_{\max}\sqrt{P}}{2\min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2}}\right)} \tag{17}$$

$$= \min_i \frac{|\boldsymbol{\rho}^T \boldsymbol{x}_i|}{\|\boldsymbol{\rho}\|_2}. \tag{18}$$

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

Next, we show that our choice of $\Lambda$ and $\boldsymbol{\lambda}$ ensures that $\boldsymbol{\rho}/\|\boldsymbol{\rho}\|_2$ 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:

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

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

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

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

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

Thus, we have shown 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 $\operatorname{sign}(f(\boldsymbol{x}_i)) = \operatorname{sign}\left(\tilde{f}(\boldsymbol{x}_i)\right)$ for all $i \in \mathcal{S}$ and $f \in \tilde{\mathcal{F}}^{\varepsilon}$. Thus, we can see that for all classifiers $f \in \tilde{\mathcal{F}}^{\varepsilon}$,

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

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

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

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, $\psi$, that satisfies conditions I–IV and choose $C_\psi > 2\varepsilon$.

Our proof uses the following result: if $\|\boldsymbol{\lambda}_{01}^* - \boldsymbol{\lambda}_\psi^*\| > C_\lambda$ 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 $\|\boldsymbol{\lambda}_{01}^* - \boldsymbol{\lambda}_\psi^*\| > C_\lambda$ implies $Z_\psi\left(\boldsymbol{\lambda}_{01}^*\right) - Z_\psi\left(\boldsymbol{\lambda}_\psi^*\right) > C_\psi$. Thus,

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

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:

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

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

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

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

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

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

## References


Antman, E. M., Cohen, M., Bernink, P. J. L. M., McCabe, C. H., Horacek, T., Papuchis, G., et al. (2000). The TIMI risk score for unstable angina/non-ST elevation MI. *The Journal of the American Medical Association*, *284*(7), 835–842.

Asparoukhov, O. K., & Stam, A. (1997). Mathematical programming formulations for two-group classification with binary variables. *Annals of Operations Research*, *74*, 89–112.

Bache, K., & Lichman, M. (2013). *UCI machine learning repository*

Bajgier, S. M., & Hill, A. V. (1982). An experimental comparison of statistical and linear programming approaches to the discriminant problem. *Decision Sciences*, *13*(4), 604–618.

Bien, J., Taylor, J., Tibshirani, R., et al. (2013). A lasso for hierarchical interactions. *The Annals of Statistics*, *41*(3), 1111–1141.

Bone, R. C., Balk, R. A., Cerra, F. B., Dellinger, R. P., Fein, A. M., Knaus, W. A., et al. (1992). 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.

Bousquet, O., Boucheron, S., & Lugosi, G. (2004). Introduction to statistical learning theory. In *Advanced lectures on machine learning*. Springer, pp. 169–207

Bradley, P. S., Fayyad, U. M., & Mangasarian, O. L. (1999). Mathematical programming for data mining: Formulations and challenges. *INFORMS Journal on Computing*, *11*(3), 217–238.

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

Brooks, J. P., & Lee, E. K. (2010). Analysis of the consistency of a mixed integer programming-based multi-category constrained discriminant model. *Annals of Operations Research*, *174*(1), 147–168.

Carrizosa, E., Martín-Barragán, B., & Morales, D. R. (2010). Binarized support vector machines. *INFORMS Journal on Computing*, *22*(1), 154–167.

Carrizosa, E., Nogales-Gómez, A., & Morales, D. R. (2013). *Strongly agree or strongly disagree? Rating features in support vector machines*. Technical report, Saïd Business School, University of Oxford, UK

Chevaleyre, Y., Koriche, F. , & Zucker, J.-D. (2013). Rounding methods for discrete linear classification. In *Proceedings of the 30th international conference on machine learning (ICML-13)* , pp. 651–659.

Cranor, L. F., & LaMacchia, B. A. (1998). Spam!. *Communications of the ACM*, *41*(8), 74–83.

Detrano, R., Janosi, A., Steinbrunn, W., Pfisterer, M., Schmid, J.-J., Sandhu, S., et al. (1989). International application of a new probability algorithm for the diagnosis of coronary artery disease. *The American journal of cardiology*, *64*(5), 304–310.

Dupačová, J., Consigli, G., & Wallace, S. W. (2000). Scenarios for multistage stochastic programs. *Annals of operations research*, *100*(1–4), 25–53.

Dupačová, J., Gröwe-Kuska, N., & Römisch, W. (2003). Scenario reduction in stochastic programming. *Mathematical programming*, *95*(3), 493–511.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, *32*(2), 407–499.

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

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

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, *33*(1), 1–22.

Gage, B. F., Waterman, A. D., Shannon, W., Boechler, M., Rich, M. W., & Radford, M. J. (2001). Validation of clinical classification schemes for predicting stroke. *The Journal of the American Medical Association*, *285*(22), 2864–2870.

Glen, J. J. (1999). Integer programming methods for normalisation and variable selection in mathematical programming discriminant analysis models. *Journal of the Operational Research Society*, *50*, 1043–1053.

Goh, S. T., & Rudin, C. (2014). Box drawings for learning with imbalanced data. In *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, pp. 333–342.

Goldberg, N., & Eckstein, J. (2010). Boosting classifiers with tightened l0-relaxation penalties. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 383–390.

Goldberg, N., & Eckstein, J. (2012). Sparse weighted voting classifier selection and its linear programming relaxations. *Information Processing Letters*, *112*, 481–486.

Guan, W., Gray, A., & Leyffer, S. (2009). Mixed-integer support vector machine. In *NIPS workshop on optimization for machine learning*.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *The Journal of Machine Learning Research*, *3*, 1157–1182.

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

Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., & Tibshirani, R. (2009). *The elements of statistical learning* (Vol. 2). New York: Springer.

Jenatton, R., Audibert, J.-Y., & Bach, F. (2011). Structured variable selection with sparsity-inducing norms. *The Journal of Machine Learning Research*, *12*, 2777–2824.

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

Joachimsthaler, E. A., & Stam, A. (1990). Mathematical programming approaches for the classification problem in two-group discriminant analysis. *Multivariate Behavioral Research*, *25*(4), 427–454.

Kapur, V. K. (2010). Obstructive sleep apnea: Diagnosis, epidemiology, and economics. *Respiratory Care*, *55*(9), 1155–1167.

Kim, M.-J., & Han, I. (2003). The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms. *Expert Systems with Applications*, *25*(4), 637–646.

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

Knaus, W. A., Draper, E. A., Wagner, D. P., & Zimmerman, J. E. (1985). APACHE II: a severity of disease classification system. *Critical Care Medicine*, *13*(10), 818–829.

Knaus, W. A., Wagner, D. P., Draper, E. A., Zimmerman, J. E., Bergner, M., Bastos, P. G., et al. (1991). The APACHE III prognostic system. Risk prediction of hospital mortality for critically ill hospitalized adults. *Chest Journal*, *100*(6), 1619–1636.

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

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

Kuhn, M., Weston, S., & Coulter, N. (2012). *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, J.-R., Loirat, P., Alperovitch, A., Glaser, P., Granthil, C., Mathieu, D., et al. (1984). A simplified acute physiology score for icu patients. *Critical Care Medicine*, *12*(11), 975–977.

Le Gall, J.-R., Lemeshow, S., & Saulnier, F. (1993). 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.

Lee, E. K., & Wu, T.-L. (2009). Classification and disease prediction via mathematical programming. In *Handbook of optimization in medicine*. Springer, pp. 1–50.

Li, L., & Lin, H.-T. (2007). Optimizing 0/1 loss for perceptrons by random coordinate descent. In *International joint conference on neural networks, 2007. IJCNN 2007*. IEEE, pp. 749–754.

Light, R. W., Macgregor, M. I., Luchsinger, P. C., & Ball, W. C. (1972). Pleural effusions: The diagnostic separation of transudates and exudates. *Annals of Internal Medicine*, *77*(4), 507–513.

Liittschwager, J. M., & Wang, C. (1978). Integer programming solution of a classification problem. *Management Science*, *24*, 1515–1525.

Lin, D., Pitler, E., Foster, D. P., & Ungar, L. H. (2008). In defense of l0. In *Workshop on feature selection (ICML 2008)*.

Liu, H., Hussain, F., Tan, C. L., & Dash, M. (2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, *6*, 393–423.

Liu, H., & Zhang, J. (2009). Estimation consistency of the group lasso and its applications. In *Proceedings of the twelfth international conference on artificial intelligence and statistics*.

Mangasarian, O. L. (1994). Misclassification minimization. *Journal of Global Optimization*, *5*(4), 309–323.

Mangasarian, O. L., Street, W. N., & Wolberg, W. H. (1995). Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, *43*(4), 570–577.

Mao, K. Z. (2004). Orthogonal forward selection and backward elimination algorithms for feature subset selection. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, *34*(1), 629–634.

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

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2012). *e1071: Misc functions of the department of statistics (e1071), TU Wien*, 2012. R package version 1.6-1.

Miller, A. J. (1984). Selection of subsets of regression variables. *Journal of the Royal Statistical Society Series A (General)*, *47*, 389–425.

Moreno, R. P., Metnitz, P. G. H., Almeida, E., Jordan, B., Bauer, P., Campos, R. A., et al. (2005). 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.

Nguyen, H. T., & Franke, K. (2012). A general lp-norm support vector machine via mixed 0-1 programming. In *Machine learning and data mining in pattern recognition*. Springer, pp. 40–49.

Nguyen, T., & Sanner, S. (2013). Algorithms for direct 0–1 loss optimization in binary classification. In *Proceedings of the 30th international conference on machine learning (ICML-13)*, pp. 1085–1093.

Pazzani, M. J. (2000). Knowledge discovery from data? *IEEE Intelligent Systems and Their Applications*, *15*(2), 10–12.
R Core Team. (2014). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. URL http://www.R-project.org/.
Ranson, J. H., Rifkind, K. M., Roses, D. F., Fink, S. D., Eng, K., Spencer, F. C., et al. (1974). Prognostic signs and the role of operative management in acute pancreatitis. *Surgery, Gynecology & Obstetrics*, *139*(1), 69.
Rubin, P. A. (1990). Heuristic solution procedures for a mixed-integer programming discriminant model. *Managerial and Decision Economics*, *11*, 255–266.
Rubin, P. A. (1997). Solving mixed integer classification problems by decomposition. *Annals of Operations Research*, *74*, 51–64.
Rubin, P. A. (2009). Mixed integer classification problems. In *Encyclopedia of optimization*. Springer, pp. 2210–2214.
Schlimmer, J. C. (1987). Concept acquisition through representational adjustment.
Souillard-Mandar, W., Davis, R., Rudin, C., Au, R., Libon, D. J., Swenson, R., et al. (2015) Learning Classification Models of Cognitive Conditions from Subtle Behaviors in the Digital Clock Drawing Test. *Machine Learning*. Accepted
Therneau, T., Atkinson, B., & Ripley, B. (2012). *rpart: Recursive Partitioning*, 2012. URL http://CRAN.R-project.org/package=rpart. R package version 4.1-0.
Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B (Methodological)*, *58*, 267–288.
Towell, G. G., & Shavlik, J. W. (1993). Extracting refined rules from knowledge-based neural networks. *Machine Learning*, *13*, 71–101.
Ustun, B., Westover, M. B., Rudin, C., & Bianchi, M. T. (2015). Clinical Prediction Models for Sleep Apnea: The Importance of Medical History over Symptoms. *Journal of clinical sleep medicine: JCSM: official publication of the American Academy of Sleep Medicine*.
Van Belle, V., Neven, P., Harvey, V., Van Huffel, S., Suykens, J. A. K., & Boyd, S. (2013). Risk group detection and survival function estimation for interval coded survival methods. *Neurocomputing*, *112*, 200–210.
Vapnik, V. (1998). *Statistical Learning Theory*. New York: Wiley.
Wells, P. S., Anderson, D. R., Bormanis, J., Guy, F., Mitchell, M., Gray, L., et al. (1997). Value of assessment of pretest probability of deep-vein thrombosis in clinical management. *Lancet*, *350*(9094), 1795–1798.
Wells, P. S., Anderson, D. R., Rodger, M., Ginsberg, J. S., Kearon, C., Gent, M., et al. (2000). Derivation of a simple clinical model to categorize patients probability of pulmonary embolism-increasing the models utility with the SimpliRED D-dimer. *Thrombosis and Haemostasis*, *83*(3), 416–420.
Wolsey, L. A. (1998). *Integer programming* (Vol. 42). New York: Wiley.
Yanev, N., & Balev, S. (1999). A combinatorial approach to the classification problem. *European Journal of Operational Research*, *115*(2), 339–350.
Zhao, P., & Bin, Y. (2007). On model selection consistency of lasso. *Journal of Machine Learning Research*, *7*(2), 25–41.
Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320.
Zeng, J., Ustun, B., & Rudin, C. (2015). Interpretable Classification Models for Recidivism Prediction. arXiv preprint arXiv:1503.07810.