FasterRisk: Fast and Accurate Interpretable Risk Scores

Jiachang Liu[1]* **Chudi Zhong**[1]* **Boxuan Li**[1] **Margo Seltzer**[2] **Cynthia Rudin**[1]
[1] Duke Univeristy [2] University of British Columbia
{jiachang.liu, chudi.zhong, boxuan.li}@duke.edu
mseltzer@cs.ubc.ca, cynthia@cs.duke.edu

Abstract

Over the last century, *risk scores* have been the most popular form of predictive model used in healthcare and criminal justice. Risk scores are sparse linear models with integer coefficients; often these models can be memorized or placed on an index card. Typically, risk scores have been created either without data or by rounding logistic regression coefficients, but these methods do not reliably produce high-quality risk scores. Recent work used mathematical programming, which is computationally slow. We introduce an approach for efficiently producing a collection of high-quality risk scores learned from data. Specifically, our approach produces a pool of almost-optimal sparse continuous solutions, each with a different support set, using a beam-search algorithm. Each of these continuous solutions is transformed into a separate risk score through a "star ray" search, where a range of multipliers are considered before rounding the coefficients sequentially to maintain low logistic loss. Our algorithm returns all of these high-quality risk scores for the user to consider. This method completes within minutes and can be valuable in a broad variety of applications.

1 Introduction

Risk scores are sparse linear models with integer coefficients that predict risks. They are arguably the most popular form of predictive model for high stakes decisions through the last century and are the standard form of model used in criminal justice [4, 22] and medicine [19, 27, 34, 31, 41]. Their history dates back to at least the criminal justice work of Burgess [8], where, based on their criminal history and demographics, individuals were assigned integer point scores between 0 and 21 that determined the probability of their "making good or of failing upon parole." Other famous risk scores are arguably the most widely-used predictive models in healthcare. These include the APGAR score [3], developed in 1952 and given to newborns, and the CHADS$_2$ score [18], which estimates stroke risk for atrial fibrillation patients. Figures 1 and 2 show example risk scores, which es-

1.	Oval Shape	-2 points		. . .
2.	Irregular Shape	4 points	+	. . .
3.	Circumscribed Margin	-5 points	+	. . .
4.	Spiculated Margin	2 points	+	. . .
5.	Age $\geq$ 60	3 points	+	
		SCORE	=	

SCORE	-7	-5	-4	-3	-2	-1
RISK	6.0%	10.6%	13.8%	17.9%	22.8%	28.6%

SCORE	0	1	2	3	4	$\geq$ 5
RISK	35.2%	42.4%	50.0%	57.6%	64.8%	71.4%

Figure 1: Risk score on the mammo dataset [15], whose population is biopsy patients. It predicts the risk of malignancy of a breast lesion. Risk score is from FasterRisk on a fold of a 5-CV split. The AUCs on the training and test sets are 0.854 and 0.853, respectively.

*These authors contributed equally.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

timate risk of a breast lesion being malignant.

1.	Irregular Shape	4 points		. . .
2.	Circumscribed Margin	-5 points	$+$	. . .
3.	SpiculatedMargin	2 points	$+$	. . .
4.	Age ≥ 45	1 point	$+$	. . .
5.	Age ≥ 60	3 points	$+$	
		SCORE	$=$	

SCORE	-5	-4	-3	-2	-1	0
RISK	7.3%	9.7%	12.9%	16.9%	21.9%	27.8%

SCORE	1	2	3	4	5	6
RISK	34.6%	42.1%	50.0%	57.9%	65.4%	72.2%

Figure 2: A second risk score on the mammo dataset on the same fold as in Figure 1. The AUCs on the training and test sets are 0.855 and 0.859, respectively. FasterRisk can produce a diverse pool of high-quality models. Users can choose a model that best fits with their domain knowledge.

Risk scores have the benefit of being easily memorized; usually their names reveal the full model – for instance, the factors in CHADS$_2$ are past **C**hronic heart failure, **H**ypertension, **A**ge$\geq$75 years, **D**iabetes, and past **S**troke (where past stroke receives **2** points and the others each receive 1 point). For risk scores, counterfactuals are often trivial to compute, even without a calculator. Also, checking that the data and calculations are correct is easier with risk scores than with other approaches. In short, risk scores have been created by humans for a century to support a huge spectrum of applications [2, 23, 30, 43, 44, 47], because humans find them easy to understand.

Traditionally, risk scores have been created in two main ways: (1) without data, with expert knowledge only (and validated only afterwards on data), and (2) using a semi-manual process involving manual feature selection and rounding of logistic regression coefficients. That is, these approaches rely heavily on domain expertise and rely little on data. Unfortunately, the alternative of building a model *directly* from data leads to computationally hard problems: optimizing risk scores over a global objective on data is NP-hard, because in order to produce integer-valued scores, the feasible region must be the integer lattice. There have been only a few approaches to design risk scores automatically [5, 6, 9, 10, 16, 32, 33, 38, 39, 40], but each of these has a flaw that limits its use in practice: the optimization-based approaches use mathematical programming solvers (which require a license) that are slow and scale poorly, and the other methods are randomized greedy algorithms, producing fast but much lower-quality solutions. We need an approach that exhibits the best of both worlds: speed fast enough to operate in a few minutes on a laptop and optimization/search capability as powerful as that of the mathematical programming tools. Our method, FasterRisk, lies at this intersection. It is fast enough to enable interactive model design and can rapidly produce a large pool of models from which users can choose rather than producing only a single model.

One may wonder why simple rounding of ℓ_1-regularized logistic regression coefficients does not yield sufficiently good risk scores. Past works [37, 39] explain this as follows: the sheer amount of ℓ_1 regularization needed to get a very sparse solution leads to large biases and worse loss values, and rounding goes against the performance gradient. For example, consider the following coefficients from ℓ_1 regularization: [1.45, .87, .83, .47, .23, .15, ...]. This model is worse than its unregularized counterpart due to the bias induced by the large ℓ_1 term. Its rounded solution is [1,1,1,0,0,0,..], which leads to even worse loss. Instead, one could multiply all the coefficients by a constant and then round, but which constant is best? There are an infinite number of choices. Even if some value of the multiplier leads to minimal loss due to rounding, the bias from the ℓ_1 term still limits the quality of the solution.

The algorithm presented here does not have these disadvantages. The steps are: (1) Fast subset search with ℓ_0 optimization (avoiding the bias from ℓ_1). This requires the solution of an NP-hard problem, but our fast subset selection algorithm is able to solve this quickly. We proceed from this accurate sparse continuous solution, preserving both sparseness and accuracy in the next steps. (2) Find a pool of diverse continuous sparse solutions that are almost as good as the solution found in (1) but with different support sets. (3) A "star ray" search, where we search for feasible integer-valued solutions along multipliers of each item in the pool from (2). By using multipliers, the search space resembles the rays of a star, because it extends each coefficient in the pool outward from the origin to search for solutions. To find integer solutions, we perform a local search (a form of sequential rounding). This method yields high performance solutions: we provide a theoretical upper bound on the loss difference between the continuous sparse solution and the rounded integer sparse solution.

Through extensive experiments, we show that our proposed method is computationally fast and produces high-quality integer solutions. This work thus provides valuable and novel tools to create risk scores for professionals in many different fields, such as healthcare, finance, and criminal justice.

Contributions: Our contributions include the three-step framework for producing risk scores, a beam-search-based algorithm for logistic regression with bounded coefficients (for Step 1), the search algorithm to find pools of diverse high-quality continuous solutions (for Step 2), the star ray search technique using multipliers (Step 3), and a theorem guaranteeing the quality of the star ray search.

2 Related Work

Optimization-based approaches: Risk scores, which model $P(y = 1|\boldsymbol{x})$, are different from threshold classifiers, which predict either $y = 1$ or $y = -1$ given $\boldsymbol{x}$. Most work in the area of optimization of integer-valued sparse linear models focuses on classifiers, not risk scores [5, 6, 9, 32, 33, 37, 40, 46]. This difference is important, because a classifier generally cannot be calibrated well for use in risk scoring: only its single decision point is optimized. Despite this, several works use the hinge loss to calibrate predictions [6, 9, 32]. All of these optimization-based algorithms use mathematical programming solvers (i.e., integer programming solvers), which tend to be slow and cannot be used on larger problems. However, they can handle both feature selection and integer constraints.

To directly optimize risk scores, typically the logistic loss is used. The RiskSLIM algorithm [39] optimizes the logistic loss regularized with ℓ_0 regularization, subject to integer constraints on the coefficients. RiskSLIM uses callbacks to a MIP solver, alternating between solving linear programs and using branch-and-cut to divide and reduce the search space. The branch-and-cut procedure needs to keep track of unsolved nodes, whose number increases exponentially with the size of the feature space. Thus, RiskSLIM's major challenge is scalability.

Local search-based approaches: As discussed earlier, a natural way to produce a scoring system or risk score is by selecting features manually and rounding logistic regression coefficients or hinge-loss solutions to integers [10, 11, 39]. While rounding is fast, rounding errors can cause the solution quality to be much worse than that of the optimization-based approaches. Several works have proposed improvements over traditional rounding. In Randomized Rounding [10], each coefficient is rounded up or down randomly, based on its continuous coefficient value. However, randomized rounding does not seem to perform well in practice. Chevaleyre [10] also proposed Greedy Rounding, where coefficients are rounded sequentially. While this technique aimed to provide theoretical guarantees for the hinge loss, we identified a serious flaw in the argument, rendering the bounds incorrect (see Appendix B). The RiskSLIM paper [39] proposed SequentialRounding, which, at each iteration, chooses a coefficient to round up or down, making the best choice according to the regularized logistic loss. This gives better solutions than other types of rounding, because the coefficients are considered together through their performance on the loss function, not independently.

A drawback of SequentialRounding is that it considers rounding up or down only to the nearest integer from the continuous solution. By considering *multipliers*, we consider a much larger space of possible solutions. The idea of multipliers (i.e., "scale and round") is used for medical scoring systems [11], though, as far as we know, it has been used only with traditional rounding rather than SequentialRounding, which could easily lead to poor performance, and we have seen no previous work that studies how to perform scale-and-round in a systematic, computationally efficient way. While the general idea of scale-and-round seems simple, it is not: there are an infinite number of possible multipliers, and, for each one, a number of possible nearby integer coefficient vectors that is the size of a hypercube, expanding exponentially in the search space.

Sampling Methods: The Bayesian method of Ertekin et al. [16] samples scoring systems, favoring those that are simpler and more accurate, according to a prior. "Pooling" [39] creates multiple models through sampling along the regularization path of ElasticNet. As discussed, when regularization is tuned high enough to induce sparse solutions, it results in substantial bias and low-quality solutions (see [37, 39] for numerous experiments on this point). Note that there is a literature on finding diverse solutions to mixed-integer optimization problems [e.g., 1], but it focuses only on linear objective functions.

Algorithm 1 FasterRisk($\mathcal{D}$,k,C,B,ϵ,T,N_m) $\rightarrow \{(\boldsymbol{w}^{+t}, w_0^{+t}, m_t)\}_t$

Input: dataset $\mathcal{D}$ (consisting of feature matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ and labels $\boldsymbol{y} \in \mathbb{R}^n$), sparsity constraint k, coefficient constraint $C = 5$, beam search size $B = 10$, tolerance level $\epsilon = 0.3$, number of attempts $T = 50$, number of multipliers to try $N_m = 20$.

Output: a pool P of scoring systems $\{(\boldsymbol{w}^t, w_0^t), m^t\}$ where t is the index enumerating all found scoring systems with $\|\boldsymbol{w}^t\|_0 \leq k$ and $\|\boldsymbol{w}^t\|_\infty \leq C$ and m^t is the corresponding multiplier.

```
1: Call Algorithm 2 SparseBeamLR($\mathcal{D}, k, C, B$) to find a high-quality solution $(\boldsymbol{w}^*, w_0^*)$ to the sparse logistic regression problem with continuous coefficients satisfying a box constraint, i.e., solve Problem (3). (Algorithm SparseBeamLR will call Algorithm ExpandSuppBy1 as a subroutine, which grows the solution by beam search.)
2: Call Algorithm 5 CollectSparseDiversePool($(\boldsymbol{w}^*, w_0^*), \epsilon, T$), which solves Problem (4). Place its output $\{(\boldsymbol{w}^t, w_0^t)\}_t$ in pool $P = \{\boldsymbol{w}^*, w_0^*\}$. $P \leftarrow P \cup \{(\boldsymbol{w}^t, w_0^t)\}_t$.
3: Send each member $t$ in the pool $P$, which is $(\boldsymbol{w}^t, w_0^t)$, to Algorithm 3 StarRaySearch ($\mathcal{D}, (\boldsymbol{w}^t, w_0^t), C, N_m$) to perform a line search among possible multiplier values and obtain an integer solution $(\boldsymbol{w}^{+t}, w_0^{+t})$ with multiplier $m_t$. Algorithm 3 calls Algorithm 6 AuxiliaryLossRounding which conducts the rounding step.
4: Return the collection of risk scores $\{(\boldsymbol{w}^{+t}, w_0^{+t}, m_t)\}_t$. If desired, return only the best model according to the logistic loss.
```

3 Methodology

Define dataset $\mathcal{D} = \{1, \boldsymbol{x}_i, y_i\}_{i=1}^n$ (1 is a static feature corresponding to the intercept) and scaled dataset as $\frac{1}{m} \times \mathcal{D} = \{\frac{1}{m}, \frac{1}{m}\boldsymbol{x}_i, y_i\}_{i=1}^n$, for a real-valued m. Our goal is to produce high-quality risk scores within a few minutes on a small personal computer. We start with an optimization problem similar to RiskSLIM's [39], which minimizes the logistic loss subject to sparsity constraints and integer coefficients:

$$\min_{\boldsymbol{w},w_0} L(\boldsymbol{w}, w_0, \mathcal{D}), \quad \text{where } L(\boldsymbol{w}, w_0, \mathcal{D}) = \textstyle\sum_{i=1}^n \log(1 + \exp(-y_i(\boldsymbol{x}_i^T\boldsymbol{w} + w_0))) \tag{1}$$
$$\text{such that} \quad \|\boldsymbol{w}\|_0 \leq k \text{ and } \boldsymbol{w} \in \mathbb{Z}^p, \;\; \forall j \in [1, .., p]\; w_j \in [-5, 5], \;\; w_0 \in \mathbb{Z}.$$

In practice, the range of these box constraints $[-5, 5]$ is user-defined and can be different for each coefficient. (We use 5 for ease of exposition.) The sparsity constraint $\|\boldsymbol{w}\|_0 \leq k$ or integer constraints $\boldsymbol{w} \in \mathbb{Z}^p$ make the problem NP-hard, and this is a difficult mixed-integer nonlinear program. Transforming the original features to all possible dummy variables, which is a standard type of preprocessing [e.g., 24], changes the model into a (flexible) generalized additive model; such models can be as accurate as the best machine learning models [39, 42]. Thus, we generally process variables in $\boldsymbol{x}$ to be binary.

To make the solution space substantially larger than $[-5, -4, ..., 4, 5]^p$, we use *multipliers*. The problem becomes:

$$\min_{\boldsymbol{w},w_0,m} L\left(\boldsymbol{w}, w_0, \frac{1}{m}\mathcal{D}\right), \text{ where } L\left(\boldsymbol{w}, w_0, \frac{1}{m}\mathcal{D}\right) = \sum_{i=1}^n \log\left(1 + \exp\left(-y_i\frac{\boldsymbol{x}_i^T\boldsymbol{w} + w_0}{m}\right)\right) \tag{2}$$
$$\text{such that } \|\boldsymbol{w}\|_0 \leq k, \boldsymbol{w} \in \mathbb{Z}^p, \;\; \forall j \in [1, .., p], \;\; w_j \in [-5, 5], \;\; w_0 \in \mathbb{Z}, \;\; m > 0.$$

Note that the use of multipliers does not weaken the interpretability of the risk score: the user still sees integer risk scores composed of values $w_j \in \{-5, -4, .., 4, 5\}$, $w_0 \in \mathbb{Z}$. Only the risk conversion table is calculated differently, as $P(Y = 1|\boldsymbol{x}) = 1/(1 + e^{-f(\boldsymbol{x})})$ where $f(\boldsymbol{x}) = \frac{1}{m}(\boldsymbol{w}^T\boldsymbol{x} + w_0)$.

Our method proceeds in three steps, as outlined in Algorithm 1. In the first step, it approximately solves the following **sparse logistic regression** problem with a box constraint (but not integer constraints), detailed in Section 3.1 and Algorithm 2.

$$(\boldsymbol{w}^*, w_0^*) \in \operatorname*{argmin}_{\boldsymbol{w},w_0} L(\boldsymbol{w}, w_0, \mathcal{D}), \;\; \|\boldsymbol{w}\|_0 \leq k, \boldsymbol{w} \in \mathbb{R}^p, \forall j \in [1, ..., p], \;\; \boldsymbol{w}_j \in [-5, 5], w_0 \in \mathbb{R}. \tag{3}$$

The algorithm gives an accurate and sparse real-valued solution $(\boldsymbol{w}^*, w_0^*)$.

The second step produces **many near-optimal sparse logistic regression solutions**, again without integer constraints, detailed in Section 3.2 and Algorithm 5. Algorithm 5 uses $(\boldsymbol{w}^*, w_0^*)$ from the first step to find a set $\{(\boldsymbol{w}^t, w_0^t)\}_t$ such that for all t and a given threshold $\epsilon_{\boldsymbol{w}}$:

$$\begin{aligned} &(\boldsymbol{w}^t, w_0^t) \text{ obeys } L(\boldsymbol{w}^t, w_0^t, \mathcal{D}) \leq L(\boldsymbol{w}^*, w_0^*, \mathcal{D}) \times (1 + \epsilon_{\boldsymbol{w}^*}) \\ &\|\boldsymbol{w}^t\|_0 \leq k, \ \boldsymbol{w}^t \in \mathbb{R}^p, \ \ \forall j \in [1, ..., p], \ w_j^t \in [-5, 5], w_0^t \in \mathbb{R}. \end{aligned} \tag{4}$$

After these steps, we have a pool of almost-optimal sparse logistic regression models. In the third step, for each coefficient vector in the pool, we **compute a risk score**. It is a feasible integer solution $(\boldsymbol{w}^{+t}, w_0^{+t})$ to the following, which includes a positive multiplier $m^t > 0$:

$$\begin{gathered} L\left(\boldsymbol{w}^{+t}, w_0^{+t}, \frac{1}{m^t}\mathcal{D}\right) \leq L(\boldsymbol{w}^t, w_0^t, \mathcal{D}) + \epsilon_t, \\ \boldsymbol{w}^{+t} \in \mathbb{Z}^p, \ \ \forall j \in [1, ..., p], w_j^{+t} \in [-5, 5], w_0^{+t} \in \mathbb{Z}, \end{gathered} \tag{5}$$

where we derive a tight theoretical upper bound on ϵ_t. A detailed solution to (5) is shown in Algorithm 6 in Appendix A. We solve the optimization problem for a large range of multipliers in Algorithm 3 for each coefficient vector in the pool, choosing the best multiplier for each coefficient vector. This third step yields a large collection of risk scores, all of which are approximately as accurate as the best sparse logistic regression model that can be obtained. All steps in this process are fast and scalable.

```
Algorithm 2 SparseBeamLR($\mathcal{D}$,$k$,$C$,$B$) $\rightarrow$ $(\boldsymbol{w}, w_0)$
Input: dataset $\mathcal{D}$, sparsity constraint $k$, coefficient constraint $C$, and beam search size $B$.
Output: a sparse continuous coefficient vector $(\boldsymbol{w}, w_0)$ with $\|\boldsymbol{w}\|_0 \leq k$, $\|\boldsymbol{w}\|_\infty \leq C$.
Define $N_+$ and $N_-$ as numbers of positive and negative labels, respectively.
$w_0 \leftarrow \log(-N_+/N_-), \boldsymbol{w} \leftarrow \mathbf{0}$        ▷Initialize the intercept and coefficients.
$\mathcal{F} \leftarrow \emptyset$        ▷Initialize the collection of found supports as an empty set
$(\mathcal{W}, \mathcal{F}) \leftarrow$ ExpandSuppBy1$(\mathcal{D}, (\boldsymbol{w}, w_0), \mathcal{F}, B)$.        ▷Returns $\leq B$ models of support 1
for $t = 2, ..., k$ do        ▷Beam search to expand the support
    $\mathcal{W}_{\text{tmp}} \leftarrow \emptyset$
    for $(\boldsymbol{w}', w_0') \in \mathcal{W}$ do        ▷Each of these has support $t - 1$
        $(\mathcal{W}', \mathcal{F}) \leftarrow$ ExpandSuppBy1$(\mathcal{D}, (\boldsymbol{w}', w_0'), \mathcal{F}, B)$.        ▷Returns $\leq B$ models with supp. $t$.
        $\mathcal{W}_{\text{tmp}} \leftarrow \mathcal{W}_{\text{tmp}} \cup \mathcal{W}'$
    end for
    Reset $\mathcal{W}$ to be the $B$ solutions in $\mathcal{W}_{\text{tmp}}$ with the smallest logistic loss values.
end for
Pick $(\boldsymbol{w}, w_0)$ from $\mathcal{W}$ with the smallest logistic loss.
Return $(\boldsymbol{w}, w_0)$.
```

3.1 High-quality Sparse Continuous Solution

There are many different approaches for sparse logistic regression, including ℓ_1 regularization [35], ElasticNet [48], ℓ_0 regularization [13, 24], and orthogonal matching pursuit (OMP) [14, 25], but none of these approaches seem to be able to handle both the box constraints and the sparsity constraint in Problem 3, so we developed a new approach. This approach, in Algorithm 2, SparseBeamLR, uses beam search for best subset selection: each iteration contains several coordinate descent steps to determine whether a new variable should be added to the support, and it clips coefficients to the box $[-5, 5]$ as it proceeds. Hence the algorithm is able to determine, before committing to the new variable, whether it is likely to decrease the loss while obeying the box constraints. This beam search algorithm for solving (3) implicitly uses the assumption that one of the best models of size k implicitly contains variables of one of the best models of size $k - 1$. This type of assumption has been studied in the sparse learning literature [14] (Theorem 5). However, we are not aware of any other work that applies box constraints or beam search for sparse logistic regression. In Appendix E, we show that our method produces better solutions than the OMP method presented in [14].

Algorithm 2 calls the ExpandSuppBy1 Algorithm, which has two major steps. The detailed algorithm can be found in Appendix A. For the first step, given a solution $\boldsymbol{w}$, we perform optimization on each single coordinate j outside of the current support $supp(\boldsymbol{w})$:

$$d_j^* \in \operatorname*{argmin}_{d \in [-5,5]} L(\boldsymbol{w} + d\boldsymbol{e}_j, w_0, \mathcal{D}) \text{ for } \forall j \text{ where } w_j = 0. \tag{6}$$

Vector $\boldsymbol{e}_j$ is 1 for the jth coordinate and 0 otherwise. We find d_j^* for each j through an iterative thresholding operation, which is done on all coordinates in parallel, iterating several (~ 10) times:

$$\text{for iteration } i\text{: } d_j \leftarrow \text{Threshold}(j, d_j, \boldsymbol{w}, w_0, \mathcal{D}) := \min(\max(c_{d_j}, -5), 5), \tag{7}$$

where $c_{d_j} = d_j - \frac{1}{l_j}\nabla_j L(\boldsymbol{w} + d_j \boldsymbol{e}_j, w_0, \mathcal{D})$, and l_j is a Lipschitz constant on coordinate j [24]. Importantly, we can perform Equation 7 on all j where $w_j = 0$ in parallel using matrix form.

For the second step, after the parallel single coordinate optimization is done, we pick the top B indices (j's) with the smallest logistic losses $L(\boldsymbol{w} + d_j^* \boldsymbol{e}_j)$ and fine tune on the new support:

$$\boldsymbol{w}_{\text{new}}^j, {w_0}_{\text{new}}^j \in \underset{\boldsymbol{a}\in[-5,5]^p, b}{\text{argmin}} \; L(\boldsymbol{a}, b, \mathcal{D}) \text{ with } supp(\boldsymbol{a}) = supp(\boldsymbol{w}) \cup \{j\}. \tag{8}$$

This can be done again using a variant of Equation 7 iteratively on all the coordinates in the new support. We get B pairs of $(\boldsymbol{w}_{\text{new}}^j, {w_0}_{\text{new}}^j)$ through this ExpandSuppBy1 procedure, and the collection of these pairs form the set $\mathcal{W}'$ in Line 8 of Algorithm 2.

At the end, Algorithm 2 (SparseBeamLR) returns the best model with the smallest logistic loss found by the beam search procedure. This model satisfies both the sparsity and box constraints.

3.2 Collect Sparse Diverse Pool (Rashomon Set)

We now collect the sparse diverse pool. In Section 3.1, our goal was to find a sparse model $(\boldsymbol{w}^*, w_0^*)$ with the smallest logistic loss. For high dimensional features or in the presence of highly correlated features, there could exist many sparse models with almost equally good performance [7]. This set of models is also known as the Rashomon set. Let us find those and turn them into risk scores. We first predefine a tolerance gap level ϵ (hyperparameter, usually set to 0.3). Then, we delete a feature with index j_- in the support $\text{supp}(\boldsymbol{w}^*)$ and add a new feature with index j_+. We select each new index to be j_+ whose logistic loss is within the tolerance gap:

$$\text{Find all } j_+ \text{ s.t. } \min_{a\in[-5,5]} L(\boldsymbol{w}^* - w_{j-}^* \boldsymbol{e}_{j-} + a\boldsymbol{e}_{j+}, w_0, \mathcal{D}) \leq L(\boldsymbol{w}^*, w_0^*, \mathcal{D})(1+\epsilon). \tag{9}$$

We fine-tune the coefficients on each of the new supports and then save the new solution in our pool. Details can be found in Algorithm 5. Swapping one feature at a time is computationally efficient, and our experiments show it produces sufficiently diverse pools over many datasets. We call this method the CollectSparseDiversePool Algorithm.

3.3 "Star Ray" Search for Integer Solutions

The last challenge is how to get an integer solution from a continuous solution. To achieve this, we use a "star ray" search that searches along each "ray" of the star, extending each continuous solution outward from the origin using many values of a multiplier, as shown in Algorithm 3. The star ray search provides much more flexibility in finding a good integer solution than simple rounding. The largest multiplier $m_{\max}$ is set to $5/\max_j(|w_j^*|)$ which will take one of the coefficients to the boundary of the box constraint at 5. We set the smallest multiplier to be 1.0 and pick N_m (usually 20) equally spaced points from $[m_{\min}, m_{\max}]$. If $m_{\max} = 1$, we set $m_{\min} = 0.5$ to allow shrinkage of the coefficients. We scale the coefficients and datasets with each multiplier and round the coefficients to integers using the sequential rounding technique in Algorithm 6. For each continuous solution (each "ray" of the "star"), we report the integer solution and multiplier with the smallest logistic loss. This process yields our collection of risk scores. Note here that a standard line search along the multiplier does not work, because the rounding error is highly non-convex.

We briefly discuss how the sequential rounding technique works. Details of this method can be found in Appendix A. We initialize $\boldsymbol{w}^+ = \boldsymbol{w}$. Then we round the fractional part of $\boldsymbol{w}^+$ one coordinate at a time. At each step, some of the w_j^+'s are integer-valued (so $w_j^+ - w_j$ is nonzero) and we pick the coordinate and rounding operation (either floor or ceil) based on which can minimize the following objective function, where we will round to an integer at coordinate r^*:

$$r^*, v^* \in \underset{r,v}{\text{argmin}} \sum_{i=1}^{n} l_i^2 \left(x_{ir}(v - w_r) + \sum_{j\neq r} x_{ij}(w_j^+ - w_j) \right)^2, \tag{10}$$

$$\text{subject to } r \in \{j \mid w_j^+ \notin \mathbb{Z}\} \text{ and } v \in \{\lfloor w_r^+ \rfloor, \lceil w_r^+ \rceil\},$$

Algorithm 3 StarRaySearch$(\mathcal{D}, (\boldsymbol{w}, w_0), C, N_m) \rightarrow (\boldsymbol{w}^+, w_0^+), m$

```
**Input:** dataset $\mathcal{D}$, a sparse continuous solution $(\boldsymbol{w}, w_0)$, coefficient constraint $C$, and number of multipliers to try $N_m$.
**Output:** a sparse integer solution $(\boldsymbol{w}^+, w_0^+)$ with $\|\boldsymbol{w}^+\|_\infty \leq C$ and multiplier $m$.
Define $m_{\max} \leftarrow C/\max|\boldsymbol{w}|$ as discussed in Section 3.3. If $m_{\max} = 1$, set $m_{\min} \leftarrow 0.5$; if $m_{\max} > 1$, set $m_{\min} \leftarrow 1$.
Pick $N_m$ equally spaced multiplier values $m_l \in [m_{\min}, m_{\max}]$ for $l \in [1, ..., N_m]$ and call this set $\mathcal{M} = \{m_l\}_l$.
Use each multiplier to scale the good continuous solution $(\boldsymbol{w}, w_0)$, to obtain $(m_l\boldsymbol{w},\ m_l w_0)$, which is a good continuous solution to the rescaled dataset $\frac{1}{m_l}\mathcal{D}$.
Send each rescaled solution $(m_l\boldsymbol{w},\ m_l w_0)$ and its rescaled dataset $\frac{1}{m_l}\mathcal{D}$ to Algorithm 6 AuxiliaryLossRounding$(\frac{1}{m_l}\mathcal{D}, m_l\boldsymbol{w}, m_l w_0)$ for rounding. It returns $(\boldsymbol{w}^{+l}, w_0^{+l}, m_l)$, where $(\boldsymbol{w}^{+l}, w_0^{+l})$ is close to $(m_l\boldsymbol{w},\ m_l w_0)$, and where $(\boldsymbol{w}^{+l}, w_0^{+l})$ on $\frac{1}{m_l}\mathcal{D}$ has a small logistic loss.
Evaluate the logistic loss to pick the best multiplier $l^* \in \text{argmin}_l\ L(\boldsymbol{w}^{+l}, w_0^{+l}, \frac{1}{m^l}\mathcal{D})$
Return $(\boldsymbol{w}^{+l^*}, w_0^{+l^*})$ and $m_{l^*}$.
```

where l_i is the Lipschitz constant restricted to the rounding interval and can be computed as $l_i = 1/(1 + \exp(y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i))$ with $\gamma_{ij} = \lfloor w_j \rfloor$ if $y_i x_{ij} > 0$ and $\gamma_{ij} = \lceil w_j \rceil$ otherwise. (The Lipschitz constant here is much smaller than the one in Section 3.1 due to the interval restriction.) After we select r^* and find value v^*, we update $\boldsymbol{w}^+$ by setting $w_{r^*}^+ = v^*$. We repeat this process until $\boldsymbol{w}^+$ is on the integer lattice: $\boldsymbol{w}^+ \in \mathbb{Z}^p$. The objective function in Equation 10 can be understood as an auxiliary upper bound of the logistic loss. Our algorithm provides an upper bound on the difference between the logistic losses of the continuous solution and the final rounded solution before we start the rounding algorithm (Theorem 3.1 below). Additionally, during the sequential rounding procedure, we do not need to perform expensive operations such as logarithms or exponentials as required by the logistic loss function; the bound and auxiliary function require only sums of squares, not logarithms or exponentials. Its derivation and proof are in Appendix C.

Theorem 3.1. *Let $\boldsymbol{w}$ be the real-valued coefficients for the logistic regression model with objective function $L(\boldsymbol{w}) = \sum_{i=1}^n \log(1 + \exp(-y_i \boldsymbol{x}_i^T \boldsymbol{w}))$ (the intercept is incorporated). Let $\boldsymbol{w}^+$ be the integer-valued coefficients returned by the AuxiliaryLossRounding method. Furthermore, let $u_j = w_j - \lfloor w_j \rfloor$. Let $l_i = 1/(1 + \exp(y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i))$ with $\gamma_{ij} = \lfloor w_j \rfloor$ if $y_i x_{ij} > 0$ and $\gamma_{ij} = \lceil w_j \rceil$ otherwise. Then, we have an upper bound on the difference between the loss $L(\boldsymbol{w})$ and the loss $L(\boldsymbol{w}^+)$:*

$$L(\boldsymbol{w}^+) - L(\boldsymbol{w}) \leq \sqrt{n \sum_{i=1}^n \sum_{j=1}^p (l_i x_{ij})^2 u_j (1 - u_j)}. \tag{11}$$

Note. *Our method has a higher prediction capacity than RiskSLIM: its search space is much larger.* Compared to RiskSLIM, our use of the multiplier permits a number of solutions that grows exponentially in k as we increase the multiplier. To see this, consider that for each support of k features, since logistic loss is convex, it contains a hypersphere in coefficient space. The volume of that hypersphere is (as usual) $V = \frac{\pi^{k/2}}{\Gamma(\frac{k}{2}+1)} r^k$ where r is the radius of the hypersphere. If we increase the multiplier to 2, the grid becomes finer by a factor of 2, which is equivalent to increasing the radius by a factor of 2. Thus, the volume increases by a factor of 2^k. In general, for maximum multiplier m, the search space is increased by a factor of m^k over RiskSLIM.

4 Experiments

We experimentally focus on two questions: (1) How good is FasterRisk's solution quality compared to baselines? (§4.1) (2) How fast is FasterRisk compared with the state-of-the-art? (§4.2) In the appendix, we address three more questions: (3) How much do the sparse beam search, diverse pools, and multipliers contribute to our solution quality? (E.4) (4) How well-calibrated are the models produced by FasterRisk? (E.9) (5) How sensitive is FasterRisk to each of the hyperparameters in the algorithm? (E.10)

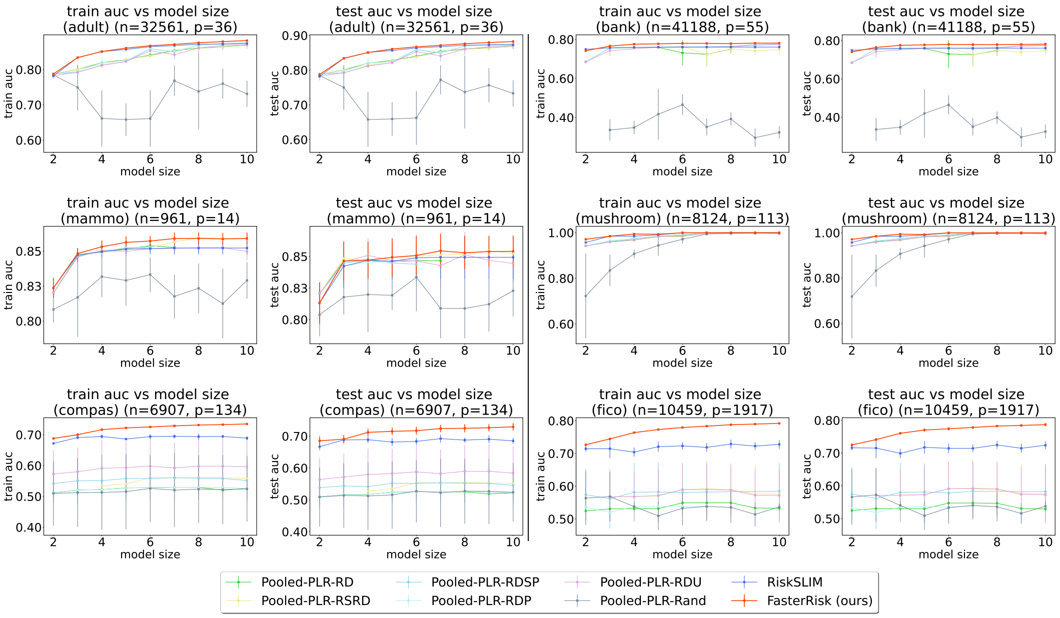

Figure 3: Performance comparison. FasterRisk outperforms all baselines due to its larger hypothesis space. On the datasets with highly-correlated variables such as COMPAS and FICO (both in the bottom row), FasterRisk outperforms other methods by a large margin.

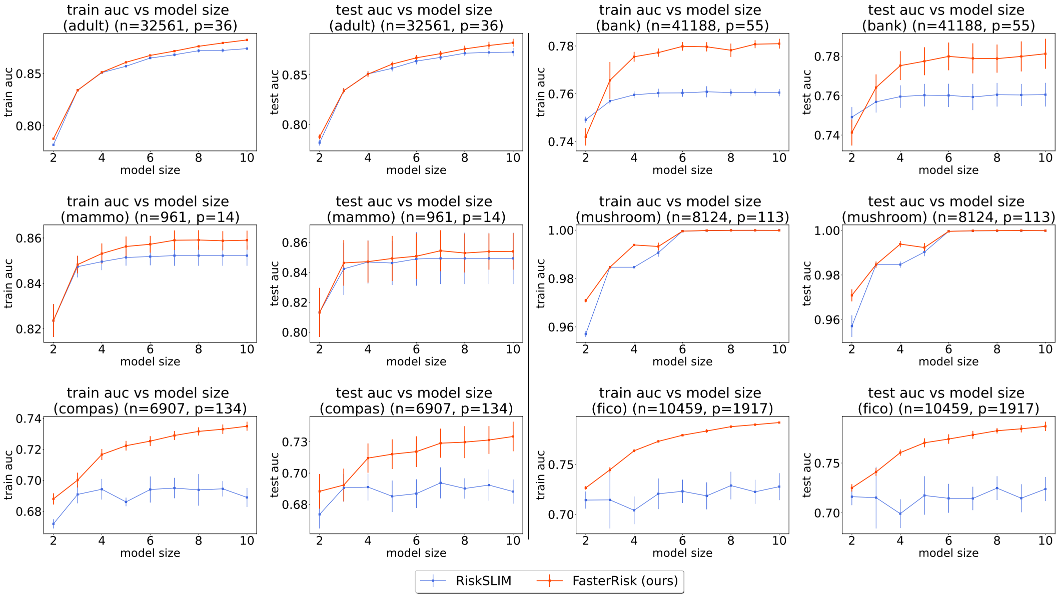

Figure 4: Performance comparison between FasterRisk and RiskSLIM.

We compare with RiskSLIM (the current state-of-the-art), as well as algorithms Pooled-PLR-RD, Pooled-PLR-RSRD, Pooled-PRL-RDSP, Pooled-PLR-Rand and Pooled-PRL-RDP. These algorithms were all previously shown to be inferior to RiskSLIM [39]. These methods first find a pool of sparse continuous solutions using different regularizations of ElasticNet (hence the name “Pooled Penalized Logistic Regression” – Pooled-PLR) and then round the coefficients with different techniques. Details are in Appendix D.3. The best solution is chosen from this pool of integer solutions that obeys the sparsity and box constraints and has the smallest logistic loss. We also compare with the baseline AutoScore [44]. However, on some datasets, the results produced by AutoScore are so poor that they distort the AUC scale, so we show those results only in Appendix E.11. As there is no publicly available code for any of [10, 16, 32, 33], they do not appear in the experiments. For each dataset, we perform 5-fold cross validation and report training and test AUC. Appendix D presents details of the datasets, experimental setup, evaluation metrics, loss values, and computing platform/environment. More experimental results appear in Appendix E.

4.1 Solution Quality

We first evaluate FasterRisk's solution quality. Figure 3 shows the training and test AUC on six datasets (results for training loss appear in Appendix E). **FasterRisk (the red line) outperforms all baselines, consistently obtaining the highest AUC scores on both the training and test sets.** Notably, our method obtains better results than RiskSLIM, which uses a mathematical solver and is the current state-of-the-art method for scoring systems. This superior performance is due to the use of multipliers, which increases the complexity of the hypothesis space. Figure 4 provides a more detailed comparison between FasterRisk and RiskSLIM. One may wonder whether running RiskSLIM longer would make this MIP-based method comparable to our FasterRisk, since the current running time limit for RiskSLIM is only 15 minutes. We extended RiskSLIM's running time limit up to 1 hour and show the comparison in Appendix E.8; FasterRisk still outperforms RiskSLIM by a large margin.

FasterRisk performs significantly better than the other baselines for two reasons. First, the continuous sparse solutions produced by ElasticNet are low quality for very sparse models. Second, it is difficult to obtain an exact model size by controlling ℓ_1 regularization. For example, Pooled-PLR-RD and Pooled-PLR-RDSP do not have results for model size 10 on the mammo datasets, because no such model size exists in the pooled solutions after rounding.

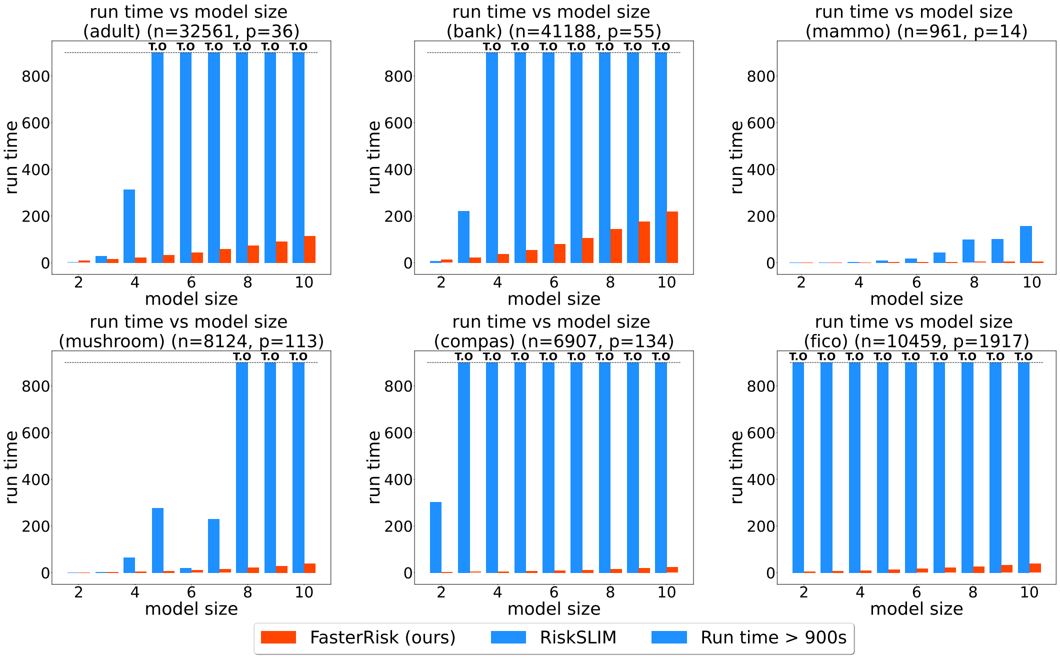

Figure 5: Runtime Comparison. Runtime (in seconds) versus model size for our method FasterRisk (in red) and the RiskSLIM (in blue). The shaded blue bars indicate cases that timed out (T.O.) at 900 seconds.

4.2 Runtime Comparison

The major drawback of RiskSLIM is its limited scalability. Runtime is important to allow interactive model development and to handle larger datasets. Figure 5 shows that **FasterRisk (red bars) is significantly faster than RiskSLIM (blue bars) in general**. We ran these experiments with a 900 second (15 minute) timeout. RiskSLIM finishes running on the small dataset mammo, but it times out on the larger datasets, timing out on models larger than 4 features for adult, larger than 3 features for bank, larger than 7 features for mushroom, larger than 2 features for COMPAS, and larger than 1 feature for FICO. RiskSLIM times out early on COMPAS and FICO datasets, suggesting that the MIP-based method struggles with high-dimensional and highly-correlated features. Thus, we see that FasterRisk tends to be both faster and more accurate than RiskSLIM.

4.3 Example Scoring Systems

The main benefit of risk scores is their interpretability. We place a few example risk scores in Table 1 to allow the reader to judge for themselves. More risk scores examples can be found in Appendix F.1. Additionally, we provide a pool of solutions for the top 12 models on the bank, mammo, and Netherlands datasets in Appendix F.2. Prediction performance is generally not the only criteria users consider when deciding to deploy a model. Provided with a pool of solutions that perform equally well, a user can choose the one that best incorporates domain knowledge [45]. After the pool of models is generated, interacting with the pool is essentially computationally instantaneous. Finally, we can reduce some models to relatively prime coefficients or transform some features for better interpretability. Examples of such transformations are given in Appendix G.1.

1.	no high school diploma	-4 points		...
2.	high school diploma only	-2 points	+	...
3.	age 22 to 29	-2 points	+	...
4.	any capital gains	3 points	+	...
5.	married	4 points	+	...
		SCORE	=	

SCORE	<-4	-3	-2	-1	0
RISK	<1.3%	2.4%	4.4%	7.8%	13.6%

SCORE	1	2	3	4	7
RISK	22.5%	35.0%	50.5%	65.0%	92.2%

(a) FasterRisk models for the adult dataset, predicting salary$>$ 50K.

1.	odor=almond	-5 points		...
2.	odor=anise	-5 points	+	...
3.	odor=none	-5 points	+	...
4.	odor=foul	5 points	+	...
5.	gill size=broad	-3 points	+	...
		SCORE	=	

SCORE	-8	-5	-3	≥2
RISK	1.62%	26.4%	73.6%	>99.8%

(b) FasterRisk model for the mushroom dataset, predicting whether a mushroom is poisonous.

Table 1: Example FasterRisk models

5 Conclusion

FasterRisk produces a collection of high-quality risk scores within minutes. Its performance owes to three key ideas: a new algorithm for sparsity- and box-constrained continuous models, using a pool of diverse solutions, and the use of the star ray search, which leverages multipliers and a new sequential rounding technique. FasterRisk is suitable for high-stakes decisions, and permits domain experts a collection of interpretable models to choose from.

Code Availability

Implementations of FasterRisk discussed in this paper are available at `https://github.com/jiachangliu/FasterRisk`.

Acknowledgements

The authors acknowledge funding from the National Science Foundation under grants IIS-2147061 and IIS-2130250, National Institute on Drug Abuse under grant R01 DA054994, Department of Energy under grants DE-SC0021358 and DE-SC0023194, and National Research Traineeship Program under NSF grants DGE-2022040 and CCF-1934964. We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC). Nous remercions le Conseil de recherches en sciences naturelles et en génie du Canada (CRSNG) de son soutien.

References

[1] Izuwa Ahanor, Hugh Medal, and Andrew C. Trapp. Diversitree: Computing diverse sets of near-optimal solutions to mixed-integer optimization problems. *arXiv*, 2022.

[2] Mohamed Farouk Allam. Scoring system for the diagnosis of COVID-19. *The Open Public Health Journal*, 13(1), 2020.

[3] Virginia Apgar. A proposal for a new method of evaluation of the newborn infant. *Current Researches in Anesthesia and Analgesia*, 1953(32):260–267, 1953.

[4] James Austin, Roger Ocker, and Avi Bhati. Kentucky pretrial risk assessment instrument validation. *Bureau of Justice Statistics*, 2010.

[5] Lieven Billiet, Sabine Van Huffel, and Vanya Van Belle. Interval coded scoring extensions for larger problems. In *Proceedings of the IEEE Symposium on Computers and Communications*, pages 198–203. IEEE, 2017.

[6] Lieven Billiet, Sabine Van Huffel, and Vanya Van Belle. Interval Coded Scoring: A toolbox for interpretable scoring systems. *PeerJ Computer Science*, 4:e150, 04 2018.

[7] Leo Breiman. Statistical modeling: The two cultures. *Statistical Science*, 16(3):199–231, 2001.

[8] Ernest W Burgess. Factors determining success or failure on parole. Illinois Committee on Indeterminate-Sentence Law and Parole Springfield, IL, 1928.

[9] E. Carrizosa, A. Nogales-Gómez, and D. Romero Morales. Strongly agree or strongly disagree?: Rating features in support vector machines. Technical report, Saïd Business School, University of Oxford, UK, 2013.

[10] Yann Chevaleyre, Frédéerick Koriche, and Jean-Daniel Zucker. Rounding methods for discrete linear classification. In *International Conference on Machine Learning*, pages 651–659. PMLR, 2013.

[11] TJ Cole. Scaling and rounding regression coefficients to integers. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 42:261–268, 1993.

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

[13] Antoine Dedieu, Hussein Hazimeh, and Rahul Mazumder. Learning sparse classifiers: Continuous and mixed integer optimization perspectives. *Journal of Machine Learning Research*, 22: 135–1, 2021.

[14] Ethan R Elenberg, Rajiv Khanna, Alexandros G Dimakis, and Sahand Negahban. Restricted strong convexity implies weak submodularity. *The Annals of Statistics*, 46(6B):3539–3568, 2018.

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

[16] Sëyda Ertekin and Cynthia Rudin. A bayesian approach to learning scoring systems. *Big Data*, 3(4):267–276, Dec 2015.

[17] FICO, Google, Imperial College London, MIT, University of Oxford, UC Irvine, and UC Berkeley. Explainable Machine Learning Challenge. `https://community.fico.com/s/explainable-machine-learning-challenge`, 2018.

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

[19] Ronald C Kessler, Lenard Adler, Minnie Ames, Olga Demler, Steve Faraone, EVA Hiripi, Mary J Howes, Robert Jin, Kristina Secnik, Thomas Spencer, and et al. The world health organization adult ADHD self-report scale (ASRS): a short screening scale for use in the general population. *Psychological Medicine*, 35(02):245–256, 2005.

[20] Ron Kohavi. Scaling up the accuracy of naive-bayes classifiers: A decision-tree hybrid. In *Proceedings Knowledge Discovery and Data Mining (KDD)*, volume 96, pages 202–207, August 1996.

[21] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How we analyzed the COMPAS recidivism algorithm. ProPublica, May 23, `https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm`, 2016.

[22] Edward Latessa, Paula Smith, Richard Lemke, Matthew Makarios, and Christopher Lowenkamp. Creation and validation of the Ohio risk assessment system: Final report. `https://www.ocjs.ohio.gov/ORAS_FinalReport.pdf`, 2009.

[23] Ji Yeon Lee, Byung-Ho Nam, Mhinjine Kim, Jongmin Hwang, Jin Young Kim, Miri Hyun, Hyun Ah Kim, and Chi-Heum Cho. A risk scoring system to predict progression to severe pneumonia in patients with COVID-19. *Scientific Reports*, 12(1):1–8, 2022.

[24] Jiachang Liu, Chudi Zhong, Margo Seltzer, and Cynthia Rudin. Fast sparse classification for generalized linear and additive models. In *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2022.

[25] Aurelie Lozano, Grzegorz Swirszcz, and Naoki Abe. Group orthogonal matching pursuit for logistic regression. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 452–460, 2011.

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

[27] Rui P Moreno, Philipp GH Metnitz, Eduardo Almeida, Barbara Jordan, Peter Bauer, Ricardo Abizanda Campos, Gaetano Iapichino, David Edbrooke, Maurizia Capuzzo, and Jean-Roger Le Gall. SAPS3 – from evaluation of the patient to evaluation of the intensive care unit. part 2: Development of a prognostic model for hospital mortality at ICU admission. *Intensive Care Medicine*, 31(10):1345–1355, 2005.

[28] Sérgio Moro, Paulo Cortez, and Paulo Rita. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62:22–31, 2014.

[29] Jeffrey Curtis Schlimmer. *Concept acquisition through representational adjustment*. PhD thesis, University of California, Irvine, 1987.

[30] Yufeng Shang, Tao Liu, Yongchang Wei, Jingfeng Li, Liang Shao, Minghui Liu, Yongxi Zhang, Zhigang Zhao, Haibo Xu, Zhiyong Peng, et al. Scoring systems for predicting mortality for severe patients with COVID-19. *EClinicalMedicine*, 24:100426, 2020.

[31] A. Jacob. Six, Barbra E. Backus, and Johannes C. Kelder. Chest pain in the emergency room: value of the heart score. *Netherlands Heart Journal*, 16(6):191–196, 2008.

[32] Nataliya Sokolovska, Yann Chevaleyre, Karine Clément, and Jean-Daniel Zucker. The fused lasso penalty for learning interpretable medical scoring systems. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4504–4511. IEEE, 2017.

[33] Nataliya Sokolovska, Yann Chevaleyre, and Jean-Daniel Zucker. A provable algorithm for learning interpretable scoring systems. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, pages 566–574. PMLR, 2018.

[34] Martin Than, Dylan Flaws, Sharon Sanders, Jenny Doust, Paul Glasziou, Jeffery Kline, Sally Aldous, Richard Troughton, Christopher Reid, and William A Parsonage. Development and validation of the emergency department assessment of chest pain score and 2h accelerated diagnostic protocol. *Emergency Medicine Australasia*, 26(1):34–44, 2014.

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

[36] Nikolaj Tollenaar and P.G.M. van der Heijden. Which method predicts recidivism best?: a comparison of statistical, machine learning and data mining predictive models. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 176(2):565–584, 2013.

[37] Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, pages 1–43, 2015. ISSN 0885-6125. doi: 10.1007/s10994-015-5528-6. URL `http://dx.doi.org/10.1007/s10994-015-5528-6`.

[38] Berk Ustun and Cynthia Rudin. Optimized risk scores. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1125–1134, 2017.

[39] Berk Ustun and Cynthia Rudin. Learning optimized risk scores. *Journal of Maching Learning Research*, 20:150–1, 2019.

[40] Berk Ustun, Stefano Traca, and Cynthia Rudin. Supersparse linear integer models for predictive scoring systems. In *Workshops at the Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[41] Berk Ustun, Lenard A Adler, Cynthia Rudin, Stephen V Faraone, Thomas J Spencer, Patricia Berglund, Michael J Gruber, and Ronald C Kessler. The world health organization adult attention-deficit / hyperactivity disorder self-report screening scale for DSM-5. *JAMA Psychiatry*, 74(5):520–526, 2017.

[42] Caroline Wang, Bin Han, Bhrij Patel, and Cynthia Rudin. In Pursuit of Interpretable, Fair and Accurate Machine Learning for Criminal Recidivism Prediction. *Journal of Quantitative Criminology*, pages 1–63, 2022. ISSN 0748-4518. doi: 10.1007/s10940-022-09545-w.

[43] Piotr Wasilewski, Bartosz Mruk, Samuel Mazur, Gabriela Półtorak-Szymczak, Katarzyna Sklinda, and Jerzy Walecki. COVID-19 severity scoring systems in radiological imaging–a

review. *Polish Journal of Radiology*, 85(1):361–368, 2020.

[44] Feng Xie, Bibhas Chakraborty, Marcus Eng Hock Ong, Benjamin Alan Goldstein, Nan Liu, et al. Autoscore: A machine learning–based automatic clinical score generator and its application to mortality prediction using electronic health records. *JMIR Medical Informatics*, 8(10):e21798, 2020.

[45] Rui Xin, Chudi Zhong, Zhi Chen, Takuya Takagi, Margo Seltzer, and Cynthia Rudin. Exploring the whole rashomon set of sparse decision trees. In *Proceedings of Neural Information Processing Systems*, 2022.

[46] Jiaming Zeng, Berk Ustun, and Cynthia Rudin. Interpretable classification models for recidivism prediction. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 180(3): 689–722, 2017.

[47] Chi Zhang, Ling Qin, Kang Li, Qi Wang, Yan Zhao, Bin Xu, Lianchun Liang, Yanchao Dai, Yingmei Feng, Jianping Sun, et al. A novel scoring system for prediction of disease severity in COVID-19. *Frontiers in Cellular and Infection Microbiology*, 10:318, 2020.

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

Appendix to FasterRisk: Fast and Accurate Interpretable Risk Scores

Table of Contents

A Additional Algorithms

A.1 Expand Support by One More Feature

Algorithm 4 ExpandSuppBy1

Input: Dataset $\mathcal{D}$, coefficient constraint C, and beam search size B, current coefficient vector $(\boldsymbol{w}, w_0)$, and a set of found supports $\mathcal{F}$.

Output: a collection of solutions $\mathcal{W} = \{(\boldsymbol{u}^t, u_0^t)\}$ with $\|\boldsymbol{u}^t\|_0 = \|\boldsymbol{w}\|_0 + 1, \|\boldsymbol{u}^t\|_\infty \leq C$ for $\forall t$. All of these solutions include the support of $(\boldsymbol{w}, w_0)$ plus one more feature. None of the solutions have the same support as any element of $\mathcal{F}$, meaning we do not discover the same support set multiple times. We also output the updated $\mathcal{F}$.

```
Let $\mathcal{S}^c \leftarrow \{j \mid w_j = 0\}$    ▷Non-support of the given solution
$\boldsymbol{w}' \leftarrow \boldsymbol{0}$
for $p = 1, ..., 10$ do    ▷10 steps of parallel coordinate descent with projection
    $w'_j \leftarrow w'_j - \nabla_j L(\boldsymbol{w} + w'_j \boldsymbol{e}_j, w_0)/l_j$ for $\forall j \in \mathcal{S}^c$    ▷$l_j$ is the smallest Lipschitz constant on coordinate $j$ with $L(\boldsymbol{w} + w'_j \boldsymbol{e}_j + d\boldsymbol{e}_j) - L(\boldsymbol{w} + w'_j \boldsymbol{e}_j) \leq l_j d$ for any $d \in \mathbb{R}$.
    $w'_j \leftarrow \text{Clip}(w'_j, -C, C)$ for $\forall j \in \mathcal{S}^c$    ▷Clip$(x, a, b) = \max(a, \min(x, b))$
end for
Pick the $B$ coords ($j$'s) in $\mathcal{S}^c$ with smallest logistic loss $L(\mathcal{D}, \boldsymbol{w} + \boldsymbol{e}_j w'_j, w_0)$, call this set $\mathcal{J}'$.
    ▷We will use these supports, which include the support of $\boldsymbol{w}$ plus one more.
$\mathcal{W} \leftarrow \emptyset$
for $j \in \mathcal{J}'$ do    ▷Optimize on the top $B$ coordinates
    If supp$(\boldsymbol{w} + \boldsymbol{e}_j w'_j) \in \mathcal{F}$, continue.    ▷We've already seen this support, so skip.
    $\mathcal{F} \leftarrow \mathcal{F} \cup \{supp(\boldsymbol{w} + \boldsymbol{e}_j w'_j)\}$.    ▷Add new support to $\mathcal{F}$.
    $(\boldsymbol{w}'', w''_0) \in \text{argmin}_{\boldsymbol{u}, u_0} L(\mathcal{D}, \boldsymbol{u}, u_0)$ with supp$(\boldsymbol{u})$ = supp$(\boldsymbol{w} + \boldsymbol{e}_j w'_j)$ and $\|\boldsymbol{u}\|_\infty \leq C$.
        ▷Fine tune on the newly expanded support using 100×|support| coordinate descent steps and clip operation, or until convergence; use $(\boldsymbol{w} + \boldsymbol{e}_j w'_j, w_0)$ as a warm start for computational efficiency
    $\mathcal{W} \leftarrow \mathcal{W} \cup \{(\boldsymbol{w}'', w''_0)\}$
end for
Return $\mathcal{W}$ and $\mathcal{F}$.
```

A.2 Collect Sparse Diverse Pool

Algorithm 5 CollectSparseDiversePool

Input: Dataset $\mathcal{D}$, a coefficient vector $(\boldsymbol{w}, w_0)$, an optimality gap tolerance ϵ, and the number of attempts T.

Output: a set $\mathcal{S}$ containing good sparse continuous solutions.

```
$\mathcal{S} \leftarrow \{(\boldsymbol{w}, w_0)\}$    ▷Initialize the sparse level set
$L^* \leftarrow L(\mathcal{D}, \boldsymbol{w}, w_0)$    ▷Get the current loss
$\mathcal{J} \leftarrow \{j \mid w_j \neq 0\}$    ▷Get the current support
for $j_- \in \mathcal{J}$ do    ▷Remove a feature in the support
    Pick the $T$ coords ($j_+$'s) in $[1, ..., p] \setminus \mathcal{J}$ with the biggest magnitudes of partial derivative $\nabla_{j+} L(\mathcal{D}, \boldsymbol{w} - \boldsymbol{e}_{j\_} w_j, w_0)$, call this set $\mathcal{J}_+$.
    for $j_+ \in \mathcal{J}_+$ do    ▷Put a new feature into the support
        $(\boldsymbol{w}'', w''_0) \in \text{argmin}_{\boldsymbol{w}', w'_0} L(\mathcal{D}, \boldsymbol{w}', w'_0)$ where $w'_j = 0$ if $j \in [1, ..., p] \setminus \mathcal{J} \cup \{j_-\}$    ▷Fit on the new support. Problem is convex. We use coordinate descent for this.
        $L_{\text{swap}} \leftarrow L(\mathcal{D}, \boldsymbol{w}'', w''_0)$    ▷Loss of newly formed and optimized coefficient vector
        if $L_{\text{swap}} \leq (1 + \epsilon)L^*$ then    ▷If its loss is good enough, include it in $\mathcal{S}$
            $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\boldsymbol{w}'', w''_0)\}$    ▷Expand the sparse level set if loss is within the gap
        end if
    end for
end for
return $\mathcal{S}$
```

A.3 Round Continuous Coefficients to Integers

Algorithm 6 AuxiliaryLossRounding

```
Input: Dataset $\mathcal{D} = (\boldsymbol{x}_i, y_i)_{i=1}^n$, a sparse continuous solution $(\boldsymbol{w}, w_0)$, where $\boldsymbol{w} \in \mathbb{R}^p, w_0 \in \mathbb{R}$.
Output: an integer-valued solution $(\boldsymbol{w}^+, w_0^+)$, where $\boldsymbol{w}^+ \in \mathbb{Z}^p, w_0^+ \in \mathbb{Z}$.
$\boldsymbol{w}^c \leftarrow [w_0, \boldsymbol{w}]$, and $\boldsymbol{x}_i \leftarrow [1, \boldsymbol{x}_i]$ for $\forall i$.    ▷Concatenate to incorporate the intercept
$\boldsymbol{w}^+ \leftarrow \boldsymbol{w}^c$
$\mathcal{J} \leftarrow \{j : \lceil w_j^+ \rceil \neq \lfloor w_j^+ \rfloor\}$    ▷Feature indices with fractional coefficients
$\mathbf{\Gamma} \leftarrow [\lfloor \boldsymbol{w}^+ \rfloor; \lfloor \boldsymbol{w}^+ \rfloor; ...; \lfloor \boldsymbol{w}^+ \rfloor]^T$    ▷n rows of $\lfloor \boldsymbol{w}^+ \rfloor$
Define a new matrix $\boldsymbol{Z}$ with entries $Z_{ij} = y_i x_{ij}$
$\mathbf{\Gamma} \leftarrow \mathbf{\Gamma} + \mathbf{1}_{\boldsymbol{Z} \leq 0}$.    ▷See Theorem3.1 and Second Inequality (Lipschitz continuity). This line
    performs the calculation: $\gamma_{ij} = \lfloor w_j \rfloor$ if $y_i x_{ij} > 0$ and $\gamma_{ij} = \lceil w_j \rceil$ otherwise.
for $i$ = 1 to $n$ do
    $l_i \leftarrow 1/(1 + \exp(y_i \sum_{j=1}^p x_{ij} \Gamma_{ij}))$    ▷Chosen so we can calculate local Lipschitz constant
end for
while $\mathcal{J} \neq \emptyset$ do    ▷We iteratively round more coeffs in $\boldsymbol{w}^+$ until fractional coeffs are gone.
    for $j \in \mathcal{J}$ do    ▷Try rounding both up and down for each j
        $\boldsymbol{w}^{+j,up} \leftarrow (w_1^+, ..., \lceil w_j^+ \rceil, ...w_{p+1}^+)^T$,    $\boldsymbol{w}^{+j,down} \leftarrow (w_1^+, ..., \lfloor w_j^+ \rfloor, ...w_{p+1}^+)^T$
        $U^{j,up} \leftarrow \sum_{i=1}^n (l_i \boldsymbol{x}_i^T (\boldsymbol{w}^{+j,up} - \boldsymbol{w}^c))^2$,    $U^{j,down} \leftarrow \sum_{i=1}^n (l_i \boldsymbol{x}_i^T (\boldsymbol{w}^{+j,down} - \boldsymbol{w}^c))^2$
    end for
        ▷Now find the best j and whether to round up or down.
    $U^{up} \leftarrow \min_{j \in \mathcal{J}} U^{j,up}$,    $U^{down} \leftarrow \min_{j \in \mathcal{J}} U^{j,down}$
    if $U^{up} \leq U^{down}$ then
        $j' \leftarrow \text{argmin}_{j \in \mathcal{J}} U^{j,up}$, $\mathcal{J} \leftarrow \mathcal{J} \setminus \{j'\}$
        $w_{j'}^+ \leftarrow \lceil w_{j'}^+ \rceil$    ▷Round up
    else
        $j' \leftarrow \text{argmin}_{j \in \mathcal{J}} U^{j,down}$, $\mathcal{J} \leftarrow \mathcal{J} \setminus \{j'\}$
        $w_{j'}^+ \leftarrow \lfloor w_{j'}^+ \rfloor$    ▷Round down
    end if
end while
$w_0^+ \leftarrow w^+[1], \boldsymbol{w}^+ \leftarrow \boldsymbol{w}^+[2 : end]$    ▷Separate the intercept and the coefficients
Return $(\boldsymbol{w}^+, w_0^+)$
```

B Comments on Proof of Chevaleyre *et al.*

Chevaleyre *et al.* [10] proposed Greedy Rounding, where coefficients are rounded sequentially. While this technique provides theoretical guarantees for greedy rounding for the hinge loss, we identified a serious flaw in their argument, rendering the bounds incorrect. We elaborate on this matter in this appendix.

The flaw is in the proof of Lemma 7. The proof essentially shows that for each sample i, there is at least one a (from the set $\{0, 1\}$) such that the inequality holds. However, the same a that works for sample $i = 3$ is not guaranteed to work for sample $i = 5$ for the inequality. It is not clear whether there exists one a that make all inequalities (for all samples i in $[1, ..., m]$) hold at the time.

To paraphrase, for each sample i, the proof shows that we can pick a set of a (either $\{0\}$, $\{1\}$, or $\{0, 1\}$) so that the inequality holds individually. However, we can not rule out the case that intersection of these individual sets is empty.

Without this extra argument, there is a gap between the statement of Lemma 7 and the proof of Lemma 7. Then, the bound for the greedy algorithm in Theorem 8 will not hold in the paper.

C Theoretical Upper Bound for the Rounding Method, Algorithm 6

The following theorem (as also shown in the main paper) states that we can provide an upper bound on the difference of the total loss between the integer solution $\boldsymbol{w}^+$ given by Algorithm 6 and the real-valued solution $\boldsymbol{w}$.

Theorem 3.1 (Loss incurred from rounding) Let $\boldsymbol{w}$ be the real-valued coefficients for the logistic regression model with objective function $L(\boldsymbol{w}) = \sum_{i=1}^n \log(1+\exp(-y_i \boldsymbol{x}_i^T \boldsymbol{w}))$. Let $\boldsymbol{w}^+$ be the integer-valued coefficients returned by the Auxiliaryloss Rounding method, Algorithm 6. Furthermore, let $u_j = w_j - \lfloor w_j \rfloor$. Let $l_i = 1/(1+\exp(y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i))$ with $\gamma_{ij} = \lfloor w_j \rfloor$ if $y_i x_{ij} > 0$ and $\gamma_{ij} = \lceil w_j \rceil$ otherwise. Then, we have an upper bound on the difference between the loss $L(\boldsymbol{w})$ and the loss $L(\boldsymbol{w}^+)$:

$$L(\boldsymbol{w}^+) - L(\boldsymbol{w}) \leq \sqrt{n \sum_{i=1}^n \sum_{j=1}^p l_i^2 x_{ij}^2 u_j (1-u_j)}. \tag{12}$$

To prove Theorem 3.1, we need to use the following Lemma C.1, which states that during each successive step of rounding a real-valued coefficient to the integer value, the deviation can be characterized and bounded by the data features and the real-valued coefficient.

Lemma C.1. *Suppose we have rounded the first $k-1$ real-valued coefficients to integers. Then for the k-th real-valued coefficient, if we set $w_k^+ = \text{argmin}_{v \in \{\lfloor w_k \rfloor, \lceil w_k \rceil\}} \sum_{i=1}^n l_i^2 (\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) + x_{ik}(v - w_k))^2$, then we have*

$$\sum_{i=1}^n l_i^2 \left(\sum_{j=1}^k x_{ij}(w_j^+ - w_j) \right)^2 \leq \sum_{i=1}^n l_i^2 \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) \right)^2 + \sum_{i=1}^n l_i^2 x_{ik}^2 (1-u_k) u_k \tag{13}$$

where $u_k = w_k - \lfloor w_k \rfloor$.

Proof. Let z_k be a binomial random variable so that $z_k = 1$ with probability u_j and $z_k = 0$ with probability $1 - u_k$. For notational convenience, let us define the function $f(v) := \sum_{i=1}^n l_i^2 \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) + x_{ik}(v - w_k) \right)^2$. Then $f(\lfloor w_k \rfloor + z_k)$ is a random variable, and the input to function $f(\cdot)$, which is $\lfloor w_k \rfloor + z_k$, takes on values either $\lfloor w_k \rfloor$ or $\lceil w_k \rceil$.

The expectation of this random variable is

$$\begin{aligned}
&\mathbb{E}_{z_k}\left[f(\lfloor w_k \rfloor + z_k)\right] \\
=&\mathbb{E}_{z_k}\left[\sum_{i=1}^n l_i^2 \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) + x_{ik}(\lfloor w_k \rfloor + z_k - w_k) \right)^2 \right] \\
=&\sum_{i=1}^n l_i^2 \, \mathbb{E}_{z_k}\left[\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) + x_{ik}(\lfloor w_k \rfloor + z_k - w_k) \right)^2 \right] && \textit{\# move } \mathbb{E}(\cdot) \textit{ inside the } \textstyle\sum(\cdot) \\
=&\sum_{i=1}^n l_i^2 \, \mathbb{E}_{z_k}\left[\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) + x_{ik}(z_k - u_k) \right)^2 \right] && \textit{\# substitute with } u_k = w_k - \lfloor w_k \rfloor \\
=&\sum_{i=1}^n l_i^2 \left[\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) \right)^2 + 2x_{ik} \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j) \right) \mathbb{E}_{z_k}\left[z_k - u_k\right] \right. \\
&\qquad \left. + x_{ik}^2 \mathbb{E}_{z_k}\left[(z_k - u_k)^2\right] \right]. && \textit{\# expand the square term}
\end{aligned}$$

Notice that because $\mathbb{P}(z_k = 1) = u_k$, $\mathbb{P}(z_k = 0) = 1 - u_k$, we have

$$\mathbb{E}_{z_k}\left[z_k - u_k\right] = (1 - u_k)u_k + (0 - u_k)(1 - u_k) = 0$$

and

$$\mathbb{E}_{z_k}\left[(z_k - u_k)^2\right] = (1 - u_k)^2 u_k + (0 - u_k)^2(1 - u_k) = u_k(1 - u_k). \qquad \textit{\# similar as above}$$

Therefore, we have

$$\begin{aligned}
&\mathbb{E}_{z_k}\left[f(\lfloor w_k \rfloor + z_k)\right] \\
=&\sum_{i=1}^{n} l_i^2 \left[\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)^2\right. \\
&\qquad \left. +2x_{ik}\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)\mathbb{E}_{z_k}\left[z_k - u_k\right] + x_{ik}^2 \mathbb{E}_{z_k}\left[(z_k - u_k)^2\right]\right] \\
=&\sum_{i=1}^{n} l_i^2 \left[\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)^2 + x_{ik}^2 u_k(1 - u_k)\right] \qquad \textit{\# plug in the two expectations above} \\
=&\sum_{i=1}^{n} l_i^2 \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)^2 + \sum_{i=1}^{n} l_i^2\ x_{ik}^2 u_k(1 - u_k). \qquad \textit{\# split into two summation terms}
\end{aligned}$$

Since the expectation of $f(\lfloor w_k \rfloor + z_k)$ is equal to $\sum_{i=1}^{n} l_i^2 \quad \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)^2 + \sum_{i=1}^{n} l_i^2\ x_{ik}^2 u_k(1 - u_k)$, there exists a $z_k' \in \{0, 1\}$ such that

$$f(\lfloor w_k \rfloor + z_k') \leq \sum_{i=1}^{n} l_i^2 \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)^2 + \sum_{i=1}^{n} l_i^2\ x_{ik}^2 u_k(1 - u_k). \tag{14}$$

Note that $\lfloor w_k \rfloor + z_k'$ is the minimizer of $f(\cdot)$ because the other input value $\lfloor w_k \rfloor + 1 - z_k'$ will take the value $f(\lfloor w_k \rfloor + 1 - z_k')$, which is greater than or equal to the expectation $\mathbb{E}_{z_k}[f(\lfloor w_k \rfloor + z_k)]$.

If we round w_k to an integer by setting $w_k^+ = \lfloor w_k \rfloor + z_k'$, then $w_k^+ = \operatorname{argmin}_{v \in \{\lfloor w_k \rfloor, \lceil w_k \rceil\}} f(v)$. We now have:

$$\begin{aligned}
\sum_{i=1}^{n} l_i^2 \left(\sum_{j=1}^{k} x_{ij}(w_j^+ - w_j)\right)^2 &= \min_{v \in \{\lfloor w_k \rfloor, \lceil w_k \rceil\}} f(v) && \textit{\# definition of } w_k^+ \text{ and } f(\cdot) \\
&= \min_{c \in \{0,1\}} f(\lfloor w_k \rfloor + c) && \textit{\# substitute } v = \lfloor w_k \rfloor + c \\
&= f(\lfloor w_k \rfloor + z_k') && \textit{\# } \lfloor w_k \rfloor + z_k' \textit{ is the minimizer of } f(\cdot) \\
&\leq \sum_{i=1}^{n} l_i^2 \left(\sum_{j=1}^{k-1} x_{ij}(w_j^+ - w_j)\right)^2 + \sum_{i=1}^{n} l_i^2 x_{ik}^2 (1 - u_k) u_k, \\
& && \textit{\# Inequality 14}
\end{aligned}$$

thus completing our proof.

□

Now we can use Lemma C.1 to prove Theorem 3.1.

Proof of Theorem 3.1. For simplicity, let us first consider the case where we round coefficients sequentially from w_1^+ to w_p^+. We claim that if at each step r, we round $w_r^+ =$

$\operatorname{argmin}_{v\in\{\lfloor w_r\rfloor,\lceil w_r\rceil\}}\sum_{i=1}^n l_i^2\left(\sum_{j=1}^{l-1}x_{ij}(w_j^+-w_j)+x_{ir}(v-w_r)\right)^2$, then for $\forall k\in[1,...,p]$

$$\sum_{i=1}^n l_i^2\left(\sum_{j=1}^k x_{ij}(w_j^+-w_j)\right)^2\le\sum_{i=1}^n\sum_{j=1}^k l_i^2x_{ij}^2u_j(1-u_j). \tag{15}$$

We prove this by the principle of induction. Suppose for step $k-1$, we have

$$\sum_{i=1}^n l_i^2\left(\sum_{j=1}^{k-1} x_{ij}(w_j^+-w_j)\right)^2\le\sum_{i=1}^n\sum_{j=1}^{k-1} l_i^2x_{ij}^2u_j(1-u_j).$$

Then, according to Lemma C.1 and the previous line, we have

$$\begin{aligned}\sum_{i=1}^n l_i^2\left(\sum_{j=1}^k x_{ij}(w_j^+-w_j)\right)^2 &\le\sum_{i=1}^n\sum_{j=1}^{k-1} l_i^2x_{ij}^2u_j(1-u_j)+\sum_{i=1}^n l_i^2x_{ik}^2u_k(1-u_k) && \textit{\# Lemma C.1}\\ &=\sum_{i=1}^n\sum_{j=1}^{k} l_i^2x_{ij}^2u_j(1-u_j). && \textit{\# use a single sum } \textstyle\sum_{i=1}^n(\cdot)\end{aligned}$$

For the base step $k=1$, Lemma C.1 also implies that

$$\sum_{i=1}^n l_i^2(x_{i1}(w_1^+-w_1))^2\le\sum_{i=1}^n l_i^2x_{i1}^2u_1(1-u_1).$$

Thus, Inequality (15) works for all k. If we let $k=p$, we have

$$\sum_{i=1}^n l_i^2\left(\sum_{j=1}^p x_{ij}(w_j^+-w_j)\right)^2\le\sum_{i=1}^n\sum_{j=1}^p l_i^2x_{ij}^2u_j(1-u_j). \tag{16}$$

Also, notice that this inequality holds for sequential rounding of any permutation of the feature indices $[1,...,p]$, and the rounding order of the AuxiliaryLossRounding method is one specific feature order. Therefore, the Inequality (16) works for the AuxiliaryLossRounding method as well.

Lastly, we use Inequality 16 to derive an upper bound on the logistic loss of the AuxiliaryLossRounding method. Recall that our objective is:

$$L(\boldsymbol{w})=\sum_{i=1}^n\log(1+\exp(-y_i\boldsymbol{x}_i^T\boldsymbol{w})). \tag{17}$$

The loss difference between the rounded solution and the real-valued solution can be bounded as follows:

$$\begin{aligned}L(\boldsymbol{w}^+)-L(\boldsymbol{w}^*) &\le\sum_{i=1}^n\left[\log(1+\exp(-y_i\boldsymbol{x}_i^T\boldsymbol{w}^+))-\log(1+\exp(-y_i\boldsymbol{x}_i^T\boldsymbol{w}))\right]\\ &\le\sum_{i=1}^n|l_i(y_i\boldsymbol{x}_i^T\boldsymbol{w}^+-y_i\boldsymbol{x}_i^T\boldsymbol{w})| && \textit{\# Lipschitz continuity, see details below}\\ &=\sum_{i=1}^n|l_iy_i\boldsymbol{x}_i^T(\boldsymbol{w}^+-\boldsymbol{w})| && \textit{\# pull out common factor}\\ &=\sum_{i=1}^n|l_i\boldsymbol{x}_i^T(\boldsymbol{w}^+-\boldsymbol{w})| && \textit{\# since } |y_i|=1\\ &\le\sum_{i=1}^n\sqrt{l_i^2\left(\sum_{j=1}^p x_{ij}(w_j^+-w_j)\right)^2} && \textit{\# rewrite } |\cdot| \textit{ in terms of } \sqrt{\cdot}\\ &\le\sqrt{n\sum_{i=1}^n l_i^2\left(\sum_{j=1}^p x_{ij}(w_j^+-w_j)\right)^2} && \textit{\# Jensen's Inequality, see details below}\end{aligned}$$

There are two inequalities we need to elaborate in details, the second and the last inequalities (Lipschitz continuity and Jensen's Inequaltiy).

Second Inequality (Lipschitz continuity):

The second inequality holds because the logistic loss $g(a) = \log(1 + \exp(-a))$ is Lipschitz continuous. If the Lipschitz constant is l, then we have $|g(a) - g(b)| \le l\ |a - b|$. We now explain how we derive the Lipschitz constant $l_i = 1/(1 + \exp(y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i))$ with $\gamma_{ij} = \lfloor w_j \rfloor$ if $y_i x_{ij} > 0$ and $\gamma_{ij} = \lceil w_j \rceil$ as stated in Theorem 3.1.

Since the logistic loss function $g(\cdot)$ is differentiable, the smallest Lipschitz constant of the function $g(\cdot)$ is $l_{\min}(g) = \sup_{a \in \text{Domain}(g)} |g'(a)|$. To see this, by the definition of the Lipschitz constant, we have $\frac{|g(a)-g(b)|}{|a-b|} \le l$. If we take the limit $b \to a$, the inequality still holds, $\lim_{b \to a} \frac{|g(a)-g(b)|}{|a-b|} \le l$. The left hand side converges to the absolute value of the derivative of $g(\cdot)$ at a. Therefore, we have $|g'(a)| \le l$. Since this works for all a, and we want to find the smallest Lipschitz value, we have $l_{\min}(g) = \sup_{a \in \text{Domain}(g)} |g'(a)|$.

For the logistic loss $g(a) = \log(1 + e^{-a})$, the absolute value of the derivative is $|g'(a)| = \frac{1}{1+e^a}$. Thus, if a is lower-bounded so that $a \ge a_1$, the smallest Lipschitz constant of the logistic loss is $l_{\min}(g) = \frac{1}{1+e^{a_1}}$.

We can apply this fact to calculate a smaller Lipschitz constant for each sample's term. If $\gamma_{ij} := \lfloor w_j \rfloor$ if $y_i x_{ij} > 0$ and $\gamma_{ij} := \lceil w_j \rceil$ otherwise, then

$$y_i \boldsymbol{x}_i \boldsymbol{w}^+ \ge y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i, \text{ and } |g'(y_i \boldsymbol{x}_i \boldsymbol{w}^+)| \le 1/(1 + \exp(y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i)).$$

Therefore, $l_i = 1/(1 + \exp(y_i \boldsymbol{x}_i^T \boldsymbol{\gamma}_i))$ is a valid Lipschitz constant for the i-th sample.

Last Inequality (Jensen's inequality):

Jensen's Inequality states that $\mathbb{E}_z[\phi(g(z))] \ge \phi(\mathbb{E}_z[g(z)])$ for any convex function $\phi(\cdot)$. For this specific problem, let $\phi(b) = -\sqrt{b}$ and let $g(z) = l_i^2 (\sum_{j=1}^p x_{ij}(w_j^+ - w_j))^2$ for a particular i with probability $\frac{1}{n}$. Then, we have

$$\begin{aligned}
\sum_{i=1}^n \sqrt{l_i^2 \left(\sum_{j=1}^p x_{ij}(w_j^+ - w_j) \right)^2} &= n \sum_{i=1}^n \frac{1}{n} \sqrt{l_i^2 \left(\sum_{j=1}^p x_{ij}(w_j^+ - w_j) \right)^2} \\
&\qquad\qquad\qquad\qquad\qquad \textit{\# multiply and divide by } n \\
&= -n \mathbb{E}_z[\phi(g(z))] \qquad\qquad \textit{\# definition of } \phi(\cdot),\ g(\cdot), \text{ and } \mathbb{E}(\cdot) \\
&\le -n \phi(\mathbb{E}_z[g(z)]) \qquad\qquad\qquad\qquad \textit{\# Jensen's Inequality} \\
&= n \sqrt{\frac{1}{n} \sum_{i=1}^n l_i^2 \left(\sum_{j=1}^p x_{ij}(w_j^+ - w_j) \right)^2} \\
&\qquad\qquad\qquad \textit{\# write out } \phi(\cdot),\ g(\cdot), \text{ and } \mathbb{E}(\cdot) \text{ explicitly} \\
&= \sqrt{n \sum_{i=1}^n l_i^2 \left(\sum_{j=1}^p x_{ij}(w_j^+ - w_j) \right)^2}. \qquad \textit{\# move } n \textit{ inside } \sqrt{\cdot}
\end{aligned}$$

Therefore, using Inequality 16, we can now bound the loss difference between the rounded solution and the real-valued solution as stated in Theorem 3.1:

$$\begin{aligned}
L(\boldsymbol{w}^+) - L(\boldsymbol{w}) &\le \sqrt{n \sum_{i=1}^n l_i^2 \left(\sum_{j=1}^p x_{ij}(w_j^+ - w_j) \right)^2} \\
&\le \sqrt{n \sum_{i=1}^n \sum_{j=1}^p l_i^2 x_{ij}^2 u_j (1 - u_j)}.
\end{aligned}$$

□

D Experimental Setup

D.1 Dataset Information

The dataset names, data source, number of samples and features, and the classification tasks can be found in Table 2. The datasets with results shown in the main paper (adult, bank, breastcancer, mammo, mushroom, spambase) were directly downloaded from this link: `https://github.com/ustunb/risk-slim/tree/master/examples/data`. The COMPAS dataset can be downloaded from this link: `https://github.com/propublica/compas-analysis/blob/master/compas-scores-two-years.csv`. The FICO dataset can be requested and downloaded from this website: `https://community.fico.com/s/explainable-machine-learning-challenge`. The Netherlands dataset is available through Data Archiving and Networked Services `https://easy.dans.knaw.nl/ui/datasets/id/easy-dataset:78692`.

For our experiments on the COMPAS, FICO, and Netherlands datasets, we convert the continuous features into a set of highly correlated dummy variables, with all entries equal to 1 or 0. By conducting experiments on these three datasets, we can test how well FasterRisk works for highly correlated features. We use the preprocessing steps as explained in Section C2 of [24]. We list the key preprocessing steps below.

COMPAS: In addition to the label *"two_year_recid"*, we use features *"sex", "age", "juv_fel_count", "juv_misd_count", "juv_other_count", "priors_count"*, and *"c_charge_degree"*.

FICO: All continuous features are used.

Netherlands: In addition to the label *"recidivism_in_4y"*, we use features *"sex", "country of birth", "log # of previous penal cases", "11-20 previous case", and ">20 previous case", "age in years", "age at first penal case"*, and *"offence type"*.

For each continuous variable $x_{\cdot,j}$, it is converted into a set of highly correlated dummy variables $\tilde{x}_{\cdot,j,\theta} = \mathbf{1}_{[x_{\cdot,j} \leq \theta]}$, where θ are all unique values that have appeared in feature column j. For Netherlands, special preprocessing steps are performed for *"age in years"* (which is real-valued, not integer) and *"age at first penal case"*. Instead of considering all unique values in the feature column, we consider 1000 quantiles.

Dataset	Source	N	P	Classification task
adult	[20]	32561	36	Predict if a U.S. resident earns more than \$50,000
bank	[28]	41188	55	Predict if a person opens account after marketing call
breastcancer	[26]	683	9	Detect breast cancer using a biopsy
mammo	[15]	961	14	Detect breast cancer using a mammogram
mushroom	[29]	8124	113	Predict if a mushroom is poisonous
spambase	[12]	4601	57	Predict if an e-mail is spam
COMPAS	[21]	6907	134	Predict if someone will be arrested $\leq$ 2 years of release
FICO	[17]	10459	1917	Predict if someone will default on a loan
Netherlands	[36]	20000	2024	Predict if someone will have any charge within 4 years

Table 2: Dataset information. Breastcancer and spambase datasets have real-valued features. All other datasets have binary (0 or 1) features.

D.2 Computing Platform

We ran all experiments on a TensorEX TS2-673917-DPN Intel Xeon Gold 6226 Processor with 2.7Ghz (768GB RAM 48 cores). For all experiments, we used only two cores because we observed using more cores did not improve the computational speed further.

D.3 Baselines

We compare with several baselines in our experiments.

RiskSLIM The current state-of-the-art method is RiskSLIM. We installed this package from the following GitHub link: `https://github.com/ustunb/risk-slim`[2]. RiskSLIM uses the IBM CPLEX MIP solver to do the optimization. The CPLEX version we used is 12.8.

Pooled Approaches For other baselines, we first found a pool of continuous sparse solutions by the ElasticNet [48] method and then rounded the coefficients to integers with different rounding techniques. Because ElasticNet has ℓ_1 and ℓ_2 penalties, we call this method the penalized logistic regression (PLR) approach. The best integer solution was selected from this pool based on which solution produces the smallest logistic loss while obeying the sparsity constraint and box constraints. These baselines correspond to the Pooled Approaches in Section 5.1 of [39], where Figure 11 and Figure 12 clearly show that pooled approaches are much better than traditional approaches. We include Unit Weighting and Rescaled Rounding as two additional rounding methods. The details of the pooled approach and the rounding techniques can be found in Section 5.1 of [39].

The ElasticNet method tries to solve the following optimization problem:

$$\min_{\boldsymbol{w}} \frac{1}{2n} \sum_{i=1}^{n} \log(1 + \exp(-y_i \boldsymbol{x}_i^T \boldsymbol{w})) + \lambda \cdot (\alpha \|\boldsymbol{w}\|_1 + (1 - \alpha)\|\boldsymbol{w}\|_2^2) \tag{18}$$

where $\alpha \in$ [0,1] is a hyperparameter. By controlling α, we choose the best model over Ridge (α =0), Lasso (α=1), and Elastic net ($0 < \alpha < 1$). We generated 1,100 models using the glmnet package[3]. To do this, we first choose 11 values of $\alpha \in \{0, 0.1, 0.2, ..., 0.9, 1.0\}$. For each given α, the package then internally and automatically selects 100 λ's equi-spaced on the logarithmic scale between $\lambda_{\min}$ and $\lambda_{\max}$ (the smallest value for λ such that all the coefficients are zero). We call this part the *Pooled-PLR* (Pooled Penalized Logistic Regression).

To convert each continuous sparse model to an integer sparse model, we applied the following rounding methods:

- 1) Pooled-PLR-RD: For each of the 1,100 PLR models in the pool, we first truncated all the coefficients (except the intercept β_0) to be within the range [-5,5] and did simple rounding: $\beta_j = \lceil \min(\max(\beta_j, -5), 5) \rfloor$, and $\beta_0 = \lceil \beta_0 \rfloor$. The $\lceil \cdot \rfloor$ operation is defined as $\lceil a \rfloor = \lceil a \rceil$ if $|a - \lceil a \rceil| < |a - \lfloor a \rfloor|$ and $\lceil a \rfloor = \lfloor a \rfloor$ otherwise.
- 2) Pooled-PLR-RDU: For each solution, we rounded each of its coefficients to be ± 1 based on its signs: $w_j = \text{sign}(w_j) \mathbb{1}_{[w_j \neq 0]}$ and $w_0 = \lceil w_0 \rfloor$ This rounding technique is known as unit weighting or the Burgess method.
- 3) Pooled-PLR-RSRD: For each solution, we rescaled its coefficients by a factor γ so that $\gamma w_{\max} = \pm 5$ and then rounded each rescaled coefficient to the nearest integer: $w_j = \lceil \gamma w_j \rfloor, \gamma = \frac{5}{\max_j |\lambda_j|}$ and $w_0 = \lceil w_0 \rfloor$.
- 4) Pooled-PLR-Rand: For each model in the pool, for each coefficient, denote its fractional part by $u_j = w_j - \lfloor w_j \rfloor$. We rounded each coefficient up to $\lceil w_j \rceil$ with probability u_j and down to $\lceil w_j \rceil$ with probability $1 - u_j$. After all rounding was done, we selected the best model in the pool.
- 5) Pooled-PLR-RDP: For each model in the pool, we iterated through each coefficient β_j and calculated the loss for both $\lceil \beta_j \rceil$ and $\lfloor \beta_j \rfloor$ and selected the rounding that minimizes the loss. This is called Sequential Rounding in [39].
- 6) Pooled-PLR-RDSP: we first rounded through Sequential Rounding (Method 5, just above), and then we applied Discrete Coordinate Descent (DCD) [39] to iteratively improve the loss by adjusting one coefficient at a time. At each round, DCD selects the coefficient and its new value that decreases the logistic loss the most.

As mentioned earlier, after we get the 1,100 integer sparse models via each rounding technique, we selected the best model from the pool based on which solution has the smallest logistic loss.

[2] The license for this package is BSD 3-Clause license. The license can be viewed on the GitHub page.

[3] We installed the package from the following GitHub link: `https://github.com/bbalasub1/glmnet_python` The package contains GNU license, which can be viewed on the GitHub website.

D.4 Hyperparameters Specification

We used the default values in Algorithm 1 for all hyperparameters. We reiterate the hyperparameters used in the experiments below.

- beam search size: $B = 10$.
- tolerance level for sparse diverse pool: $\epsilon = 0.3$ (or 30%).
- number of attempts to try for sparse diverse pool: $T = 50$.
- number of multipliers to try: $N_m = 20$.

Performance is not particularly sensitive to these choices (see Appendix E.10). If T, N_m, B are chosen too large, the algorithm will take longer to execute.

E Additional Experimental Results

E.1 Additional Results on Solution Quality

In addition to the six datasets we show in the main paper, we provide results on the breastcancer, spambase, and Netherlands datasets (see Section D.1 for more data information). The comparison of solution quality is shown in Figure 6. We see that FasterRisk outperforms both RiskSLIM and other pooled approaches, even with high dimensional feature spaces and in the presence of highly correlated features (the Netherlands dataset).

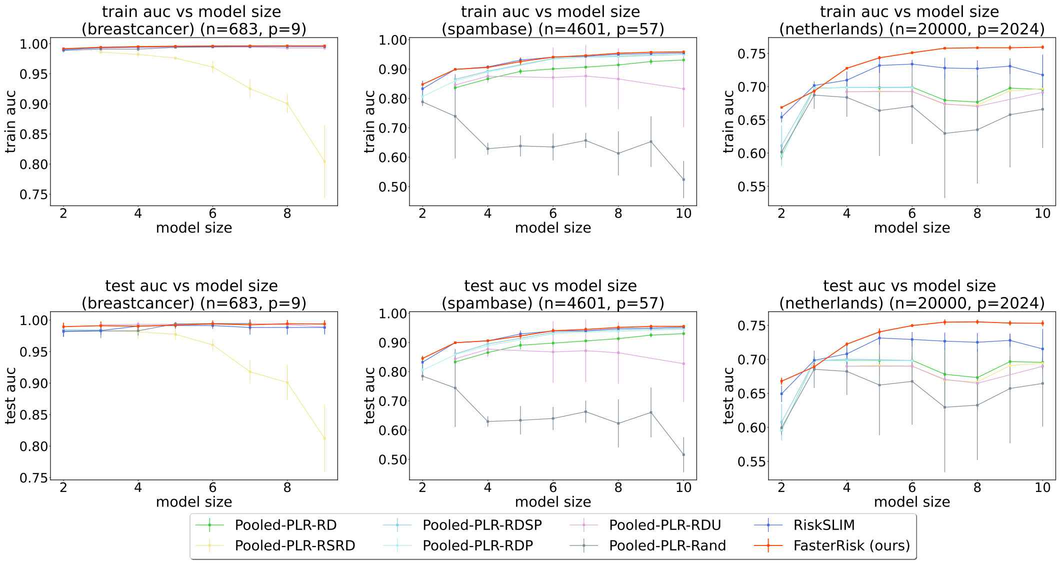

Figure 6: Performance comparison on the breastcancer, spambase, and Netherlands datasets. Top row is training AUC (higher is better) and bottom row is test AUC (higher is better).

E.2 Additional Results on Direct Comparison with RiskSLIM

As RiskSLIM provides state-of-the-art performance, we compare it to FasterRisk in isolation to highlight the differences between the two approaches/algorithms. The results are shown in Figure 7 on the breastcancer, spambase, and Netherlands datasets.

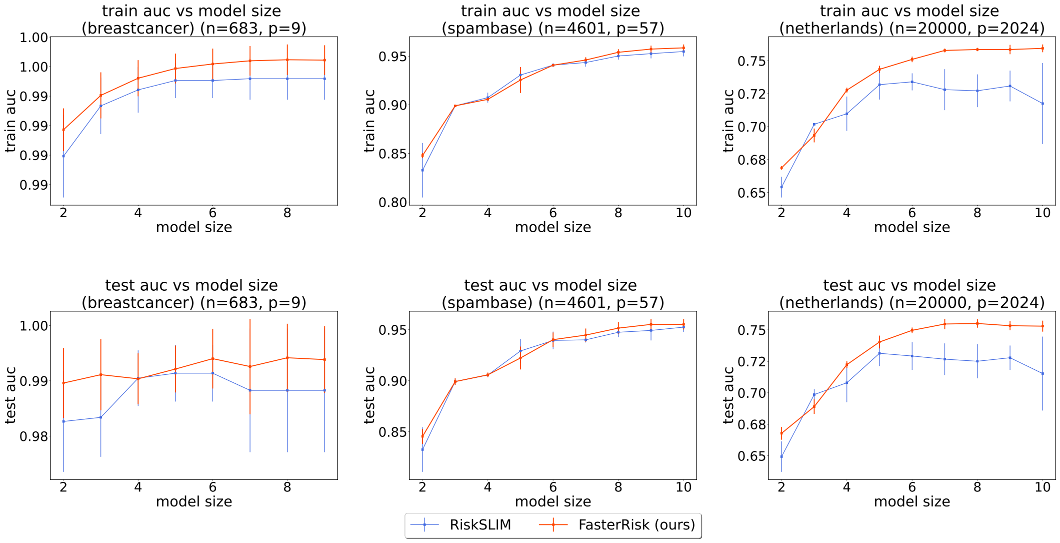

Figure 7: Detailed performance comparison between FasterRisk and RiskSLIM on breastcancer, spambase, and Netherlands. Top row is training AUC (higher is better) and bottom row is test AUC (higher is better). We can improve FasterRisk's results on the spambase dataset by increasing the beam size in the algorithm. See Figure 29 for the perturbation study on this hyperparameter.

E.3 Additional Results on Running Time

We also provide a runtime comparison between RiskSLIM and FasterRisk in Figure 8. Except for the small dataset breastcancer, RiskSLIM timed out in all other instances. In contrast, FasterRisk finishes running under 50s or 100s on all cases, showing great scalability, even in high dimensional feature space and in presence of highly correlated features (the Netherlands dataset).

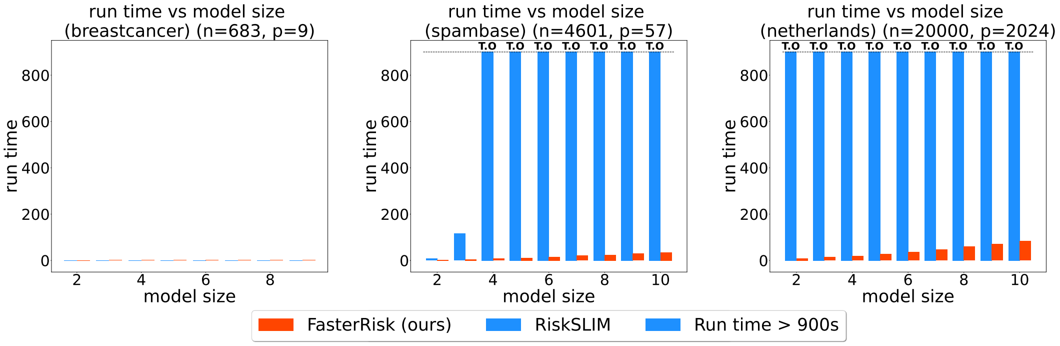

Figure 8: Runtime Comparison. Runtime (in seconds) versus model size for our method FasterRisk (in red) and the RiskSLIM (in blue). The shaded blue bars indicate cases that timed out ("T.O") at 900 seconds.

E.4 Ablation Study of the Proposed Techniques

We investigate how each component of FasterRisk, including sparse beam search, diverse pool, and multipliers, contribute to solution quality. We quantify the contribution of each part of the algorithm by means of an ablation study in which we run variations of FasterRisk, each with a single component disabled.

The results are shown in Figure 9-11. “no beam search” means that the beam size is 1, so we expand the support by picking the next feature based on which new feature can induce the smallest logistic loss via the single coordinate optimization. “no sparse diverse” means that the sparse diverse pool contains only the solution by Algorithm 2, the SparseBeamLR method. “no multiplier” means that there is no “star ray search” of the multiplier. There is no scaling of coefficients or the data, so we think of this as setting multiplier to 1.

The ablation study shows that different parts of our algorithm provide the biggest benefit to different data sets — that is, there is no single component of the algorithm that uniformly assists with performance; instead, the combination of these techniques, working in concert, is responsible. We provide the detailed analysis of the contributions for each specific dataset in the figure captions.

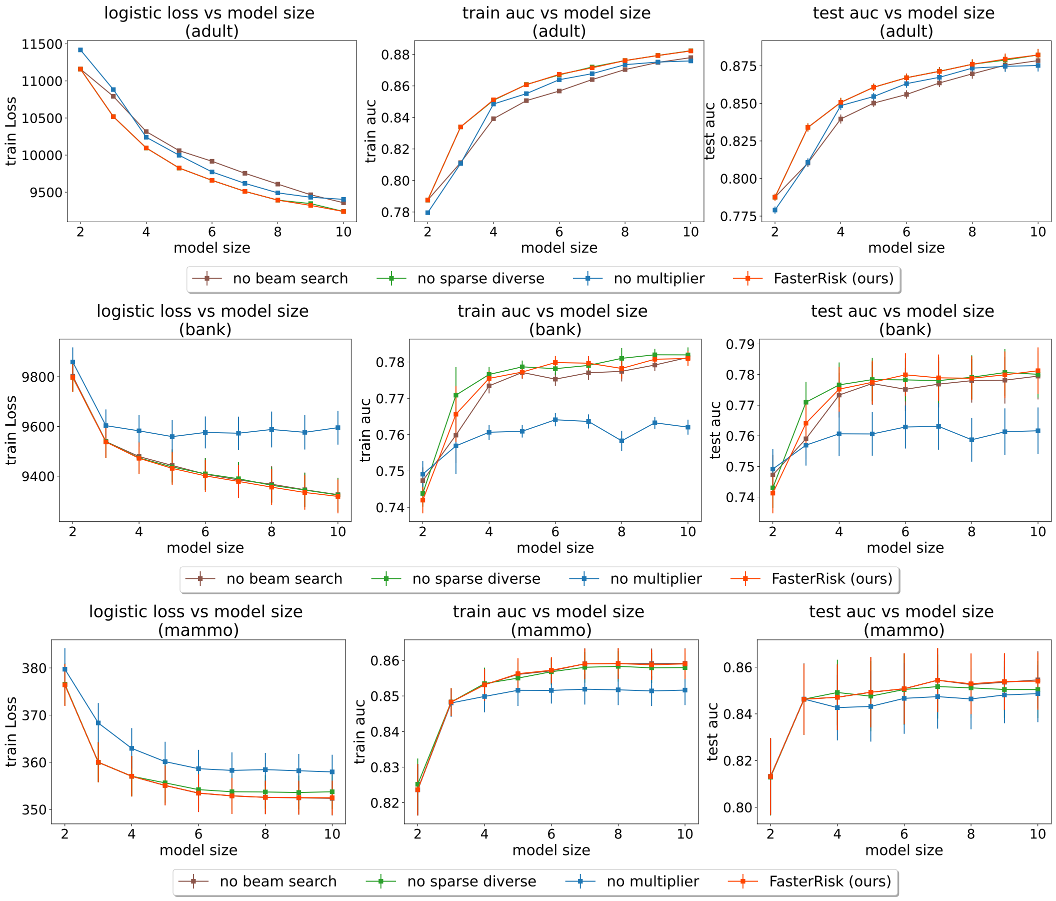

Figure 9: Ablation study on the adult, bank, and mammo datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). The “beam search” method is particularly helpful on the adult dataset. The use of “multiplier” is particularly helpful on all three datasets. The “diverse pool” technique is somewhat helpful on the mammo dataset. More significant contributions from “diverse pool” can be found in Figure 11 and Figure 10.

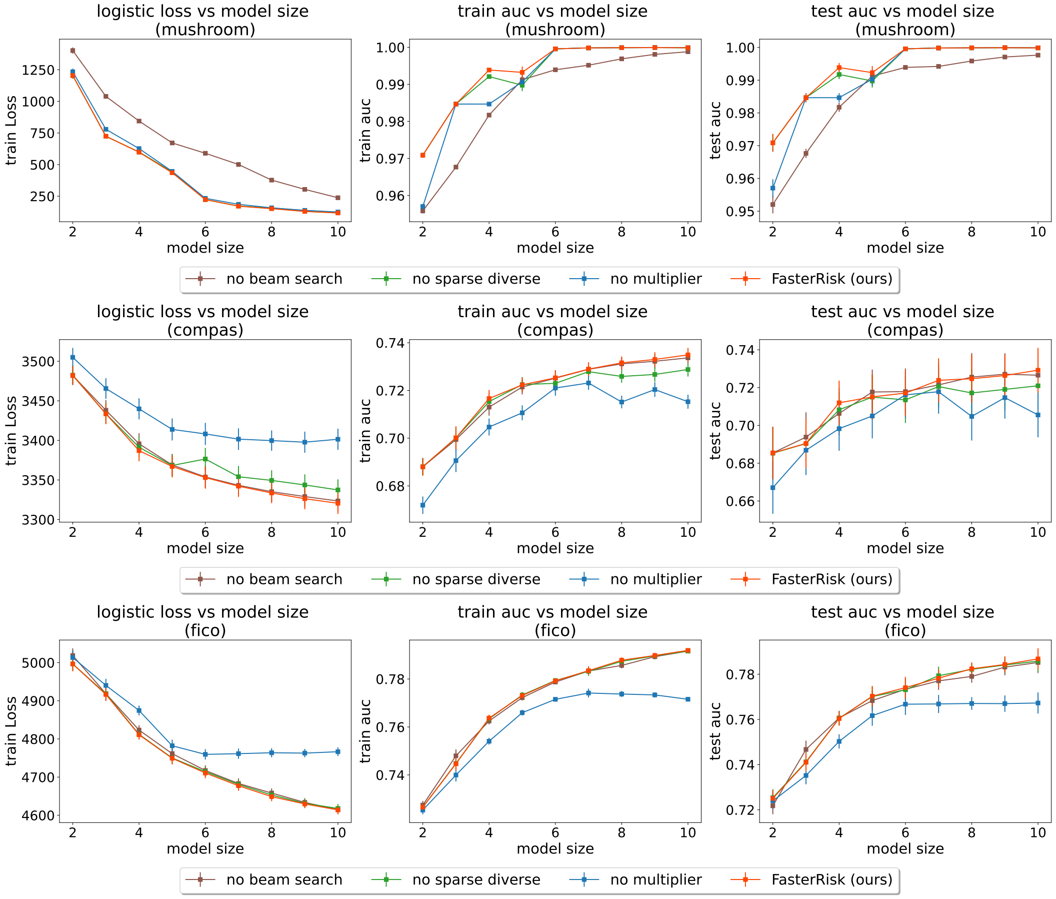

Figure 10: Ablation study on the mushroom, COMPAS, and FICO datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). The "beam search" method is particularly helpful on the mushroom dataset. The use of "multiplier" is particularly helpful on the COMPAS and FICO datasets. The "diverse pool" technique is particularly helpful on the COMPAS dataset.

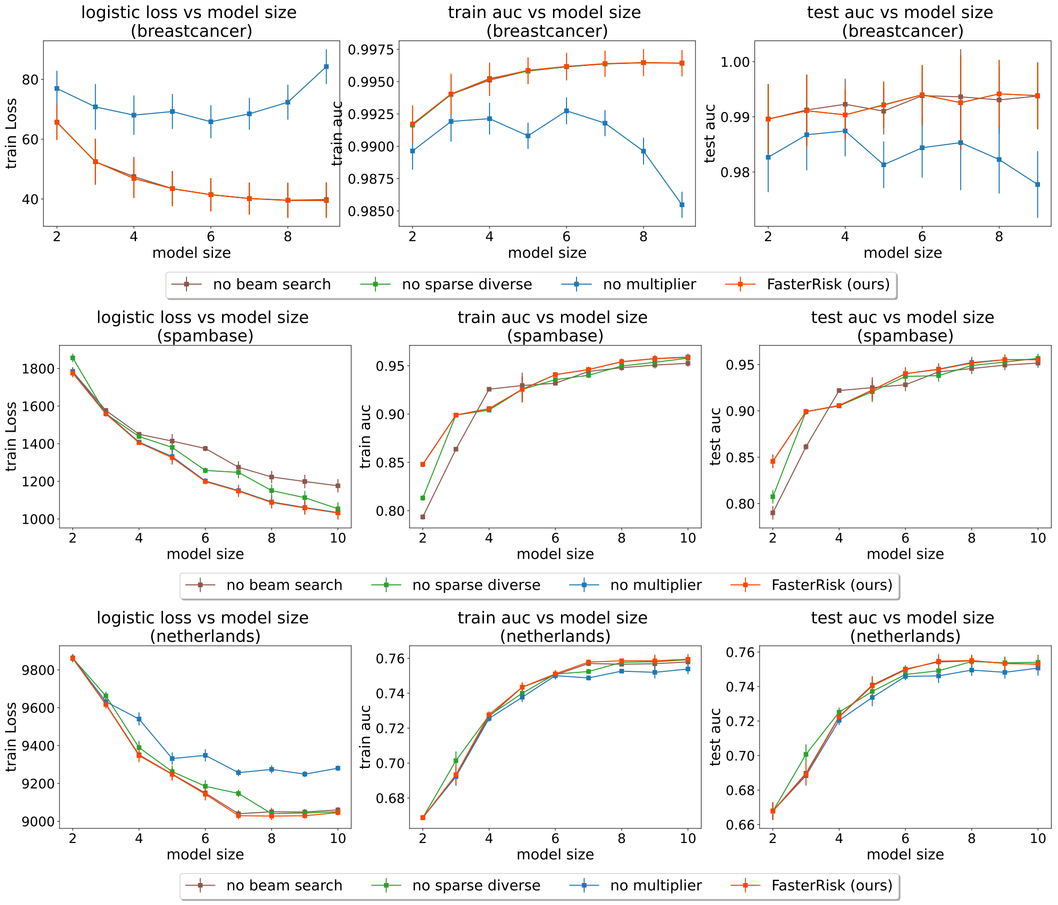

Figure 11: Ablation study on the COMPAS, FICO, and Netherlands datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). The "beam search" method is particularly helpful on the spambase dataset. The use of "multiplier" is particularly helpful on breastcancer and netherlands datasets. The "diverse pool" technique is particularly helpful on the spambase and Netherlands datasets.

E.5 Training Losses of FasterRisk vs. RiskSLIM

In the main paper, due to the page limit, we have only compared the training and test AUCs between RiskSLIM and our FasterRisk. Here, we provide the comparison of training loss (logistic loss) between these two methods. The results are shown in Figure 12. We can see that FasterRisk outperforms RiskSLIM in almost all model size instances and datasets.

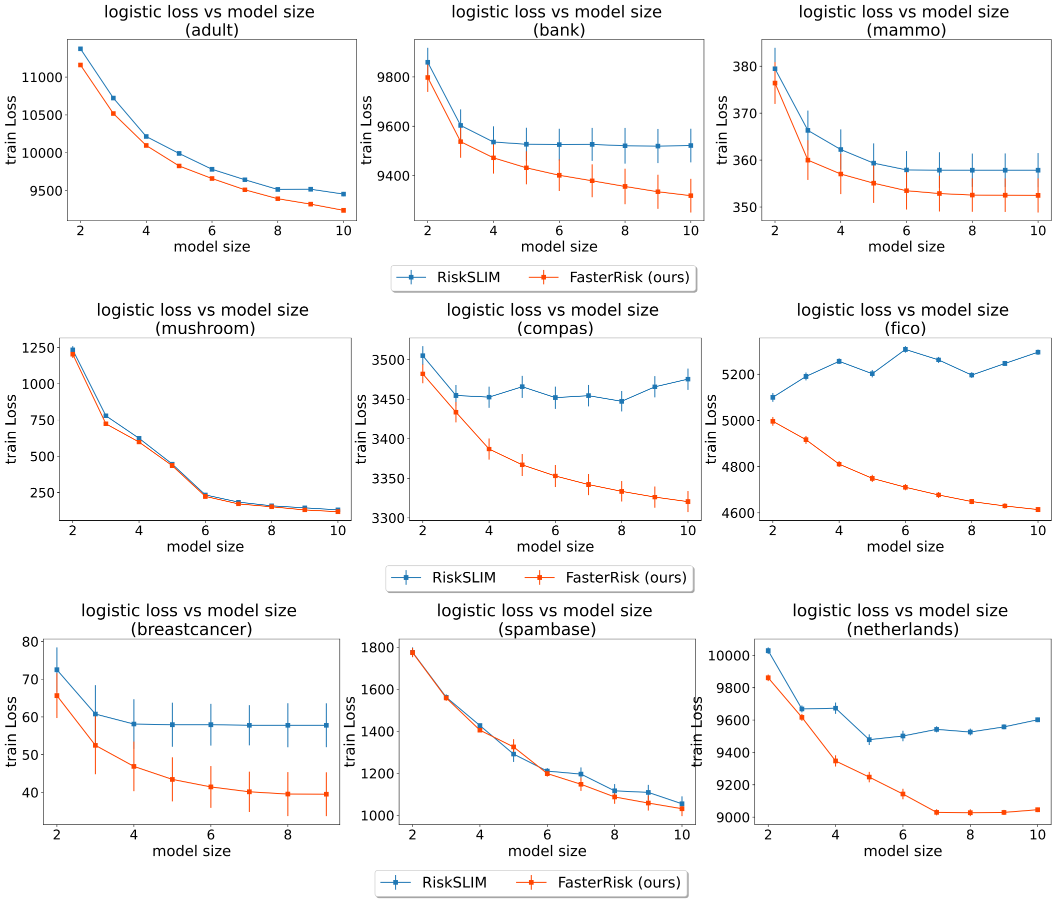

Figure 12: Training loss between RiskSLIM and our FasterRisk methods. (lower is better)

E.6 Comparison of SparseBeamLR with OMP and fastSparse

We next study how effective SparseBeamLR is in producing continuous sparse coefficients under the ℓ_0 sparsity and box constraints. We compare with two existing methods, OMP [14] and fastSparse [24]. OMP stands for Orthogonal Matching Pursuit, which expands the support by selecting the next feature with the largest magnitude of partial derivative. fastSparse tries to solve the logistic loss objective with an ℓ_0 regularization. For fastSparse, we use the default λ_0 values (coefficient for the ℓ_0 regularization) internally selected by the software. Specifically, the software first apply a large λ_0 value to produce a super-sparse solution (with support size equal to 1 or close to 1). Then, in the solution path, the λ_0 value is sequentially decreased until the produced sparse model violates the model size constraint.

The results are shown in Figure 13-15. Although OMP and fastSparse can somtimes produce high-quality solutions on some model size instances and datasets, SparseBeamLR is the only method that consistently produces high-quality sparse solutions in all cases.

OMP's solution quality is usually worse than that of SparseBeamLR, and OMP could not produce coefficients that satisfy the box constraints on the mushroom and spambase datasets.

fastSparse also cannot produce coefficients that satisfy the box constraints on the mushroom and spambase datasets. Additionally, it is hard to control the λ_0 regularization to produce the exact model size desired. In Figure 14, we do not obtain any model with model size equal to 9 or 10 in the solution path.

The limitations of OMP and fastSparse stated above are our main motivations for developing the SparseBeamLR method.

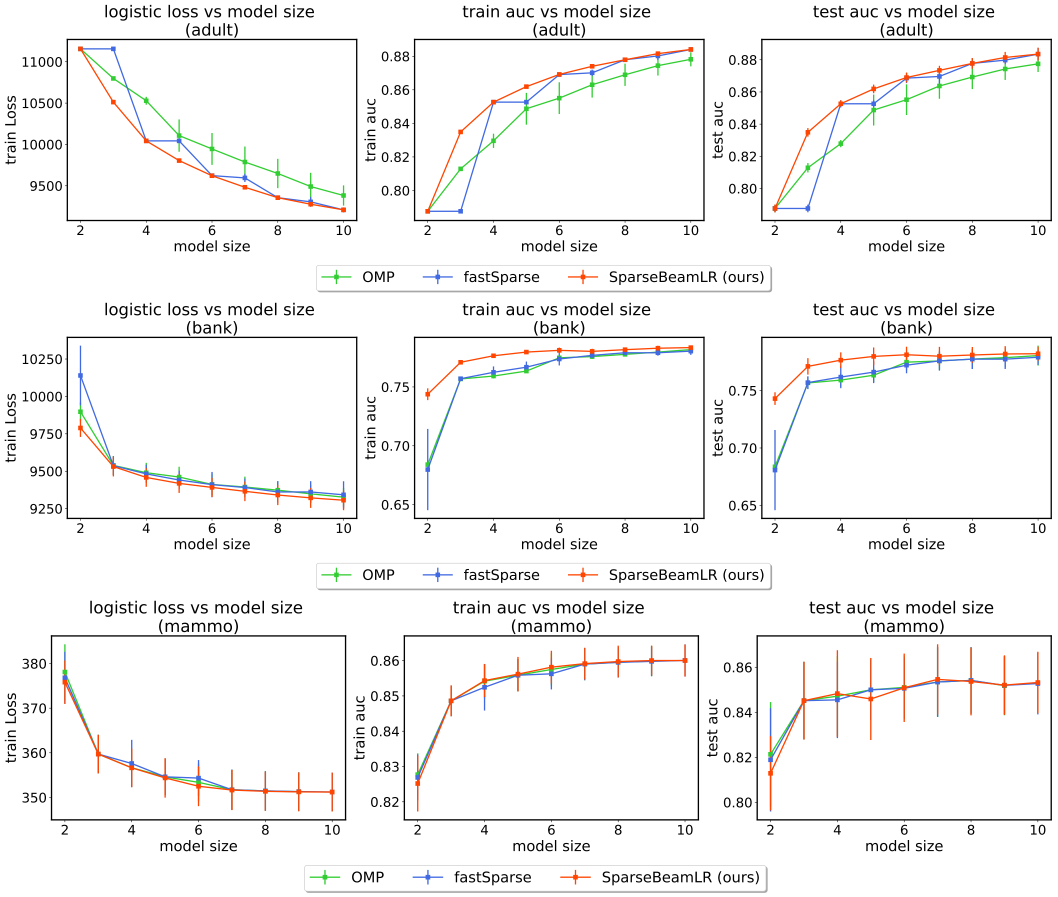

Figure 13: Sparse continuous solutions on the adult, bank, and mammo datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). SparseBeamLR consistently produces high-quality continuous sparse solutions.

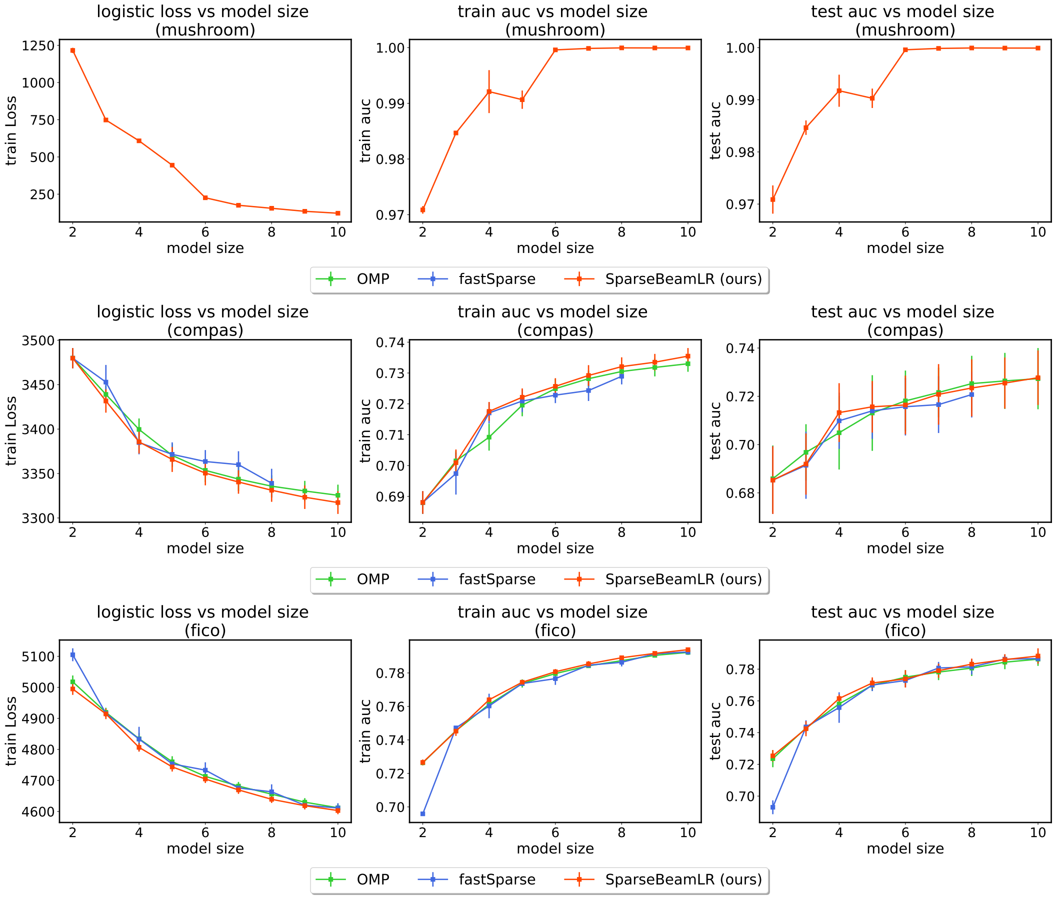

Figure 14: Sparse continuous solutions on the mushroom, COMPAS, and FICO datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). The solution coefficients by the OMP and fastSparse methods violate the box constraints on the mushroom dataset, so we omit the results in the plot. fastSparse cannot obtain solutions with model size equal to 9 or 10 on the COMPAS dataset, so we do not show those points in the plot.

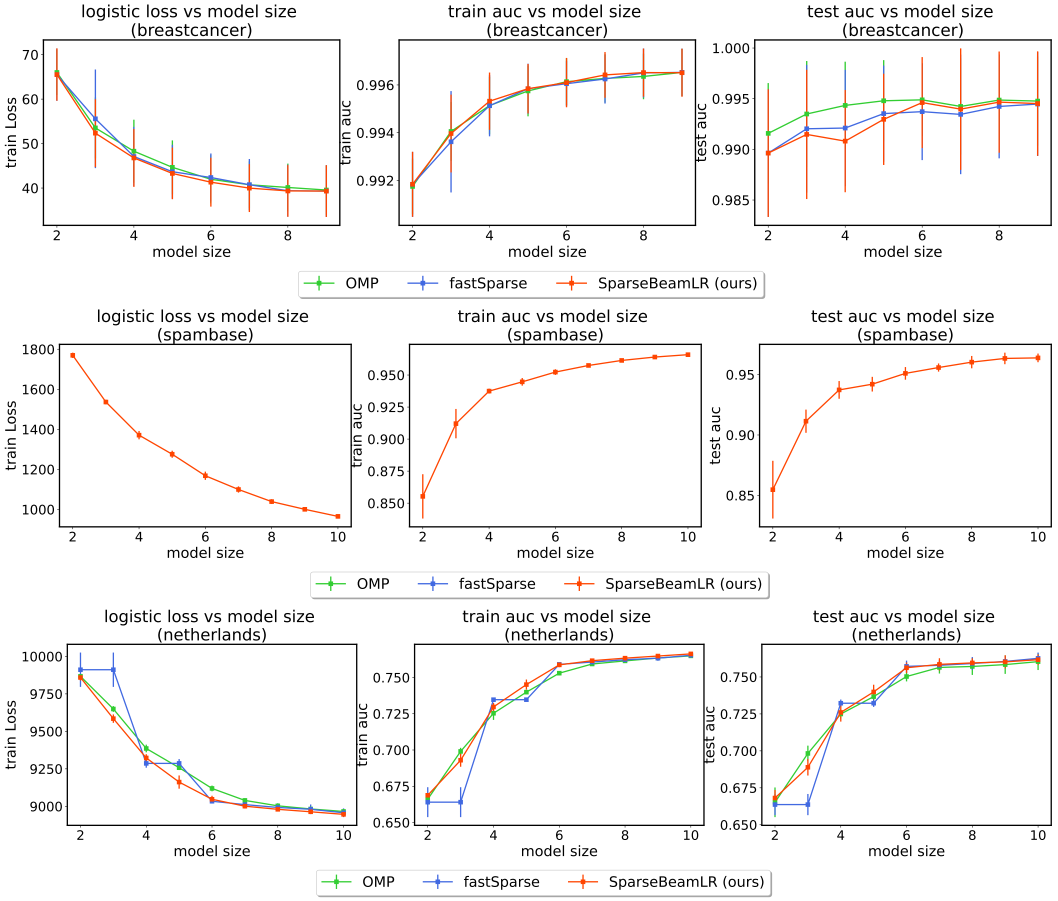

Figure 15: Sparse continuous solutions on the breastcancer, spambase, and Netherlands datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). SparseBeamLR consistently produces high-quality continuous sparse solutions. The solution coefficients of OMP and fastSparse violate the box constraints on the spambase dataset, so we omit the results in the plot.

E.7 Comparison of FasterRisk with OMP (or fastSparse) + Sequential Rounding

Having compared the continuous sparse solutions, we next compare the integer sparse solutions produced by OMP, fastSparse, and FasterRisk. After obtaining the continuous sparse solutions from OMP and fastSparse from Section E.6, we round the continuous coefficients to integers using the Sequential Rounding method as stated in Method 5 of D.3.

The results are shown in Figure 16-18. FasterRisk consistently outperforms the other two methods, due to higher quality of continuous sparse solutions and the use of multipliers.

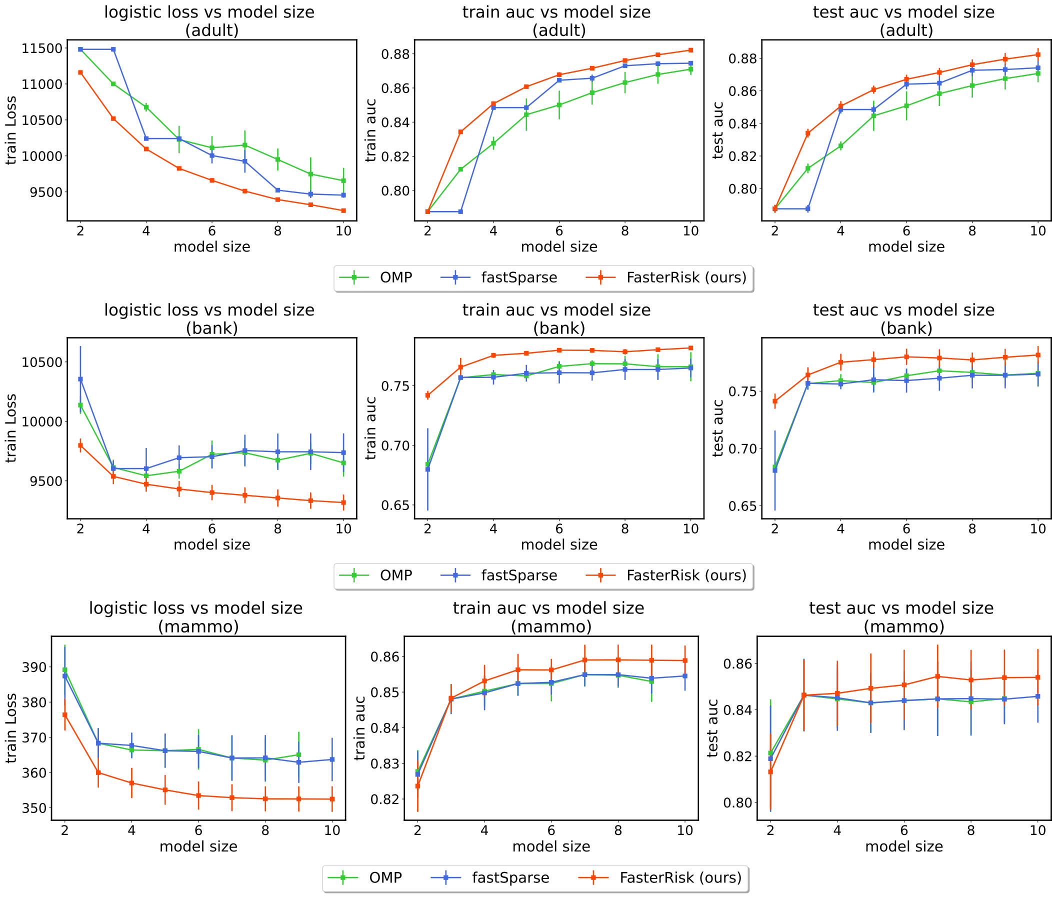

Figure 16: Sparse integer solutions on the adult, bank, and mammo datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). FasterRisk consistently outperforms the other two methods, due to higher quality of continuous sparse solutions and the use of multipliers.

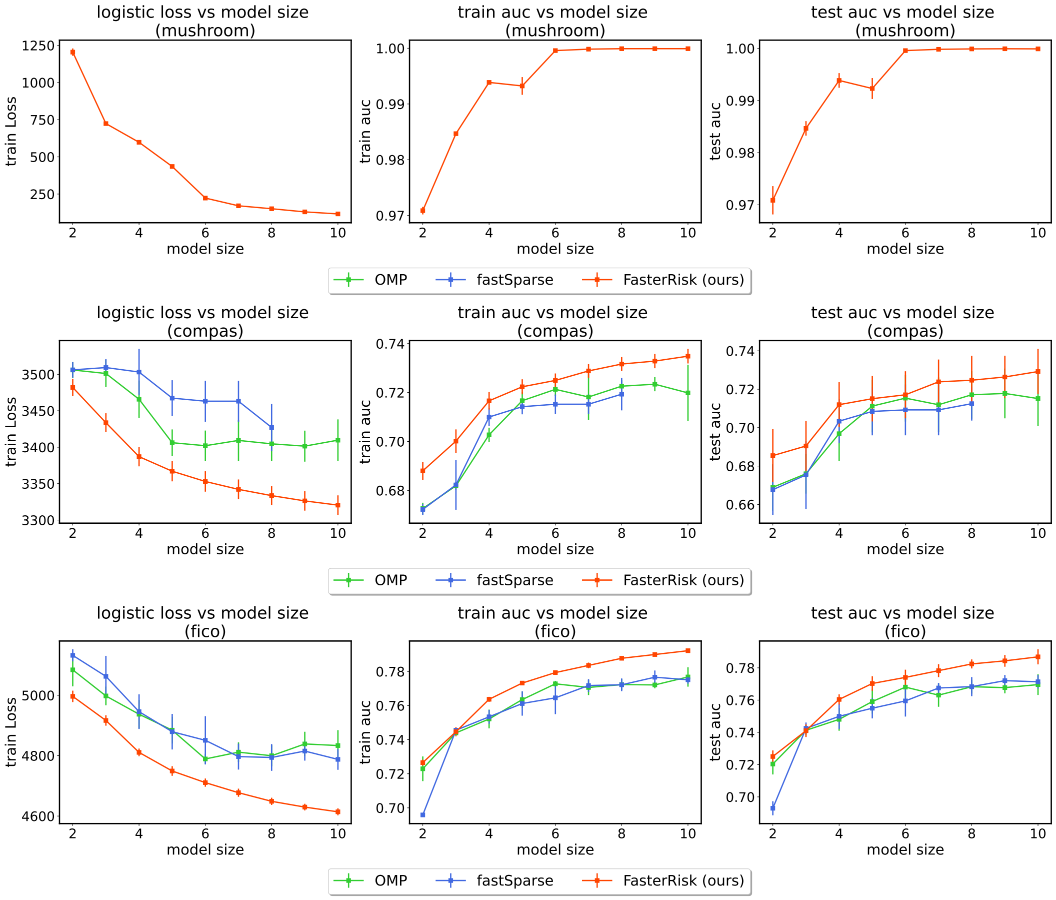

Figure 17: Sparse integer solutions on the mushroom, COMPAS, and FICO datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). The solution coefficients from the OMP and fastSparse methods violate the box constraints on the mushroom dataset, so we omit the results on the plot. fastSparse cannot obtain solutions with model size equal to 9 or 10 on the COMPAS dataset, so we do not show those points on the plot.

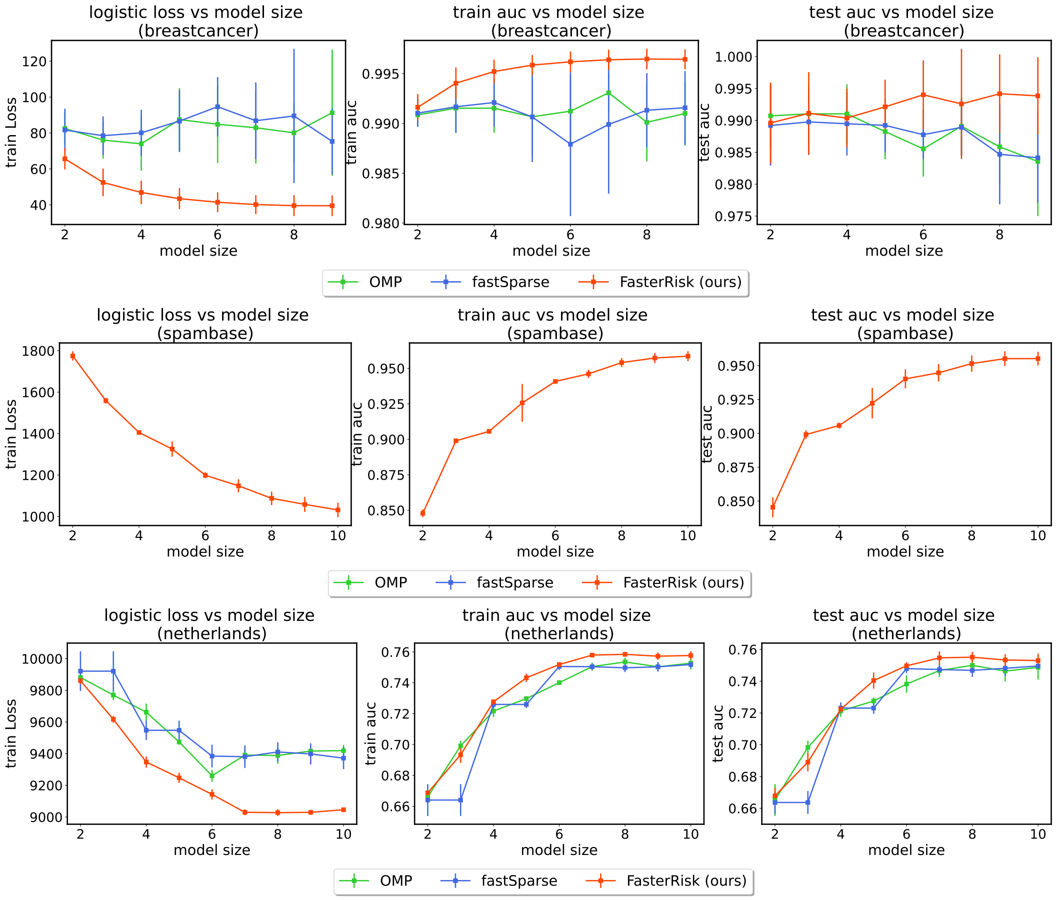

Figure 18: Sparse integer solutions on the breastcancer, spambase, and Netherlands datasets. Left column is loss (lower is better), middle column is training AUC (higher is better) and right column is test AUC (higher is better). FasterRisk consistently outperforms the other two methods, due to the higher quality of the continuous sparse solutions and the use of multipliers. The solution coefficients by the OMP and fastSparse methods violate the box constraints on the spambase dataset, so we omit the results in the plot.

E.8 Running RiskSLIM Longer

The experiments in Section 4 imposed a 900-second timeout, and RiskSLIM frequently did not complete within the 900 seconds. Here, we run RiskSLIM with longer timeouts (1 hour, and 4 days). We find that even with these long runtimes, FasterRisk still outperforms RiskSLIM in both solution quality and runtime.

Runtime is important for two reasons: (1) We may not be able to compute the answer at all using the slow method because it does not scale to reasonably-sized datasets. It could take a week or more to compute the solution for even reasonably small datasets. We will show this shortly through experiments. (2) Machine learning in the wild is never a single run of an algorithm. Often, users want to explore the data and adjust various constraints as they become more familiar with possible models. A fast speed allows users to go through this iteration process many times without lengthy interruptions between runs. This is where FasterRisk will be very useful in high stakes offline settings. FasterRisk's pool of models is generated within 5 minutes, and interacting with the pool is essentially instantaneous after it is generated.

E.8.1 Solution Quality of Running RiskSLIM for 1 hour

We ran RiskSLIM for a time limit of 1 hour on all 5 folds and all model sizes (2-10). Thus, we ran experiments for 2 days per dataset. As a reminder, our method FasterRisk runs in less than 5 minutes (on all datasets). The results of logistic loss on the training set, AUC on the training set, and AUC on the test set are in Figures 19-21. FasterRisk still outperforms RiskSLIM in almost all cases, because it uses a larger search space.

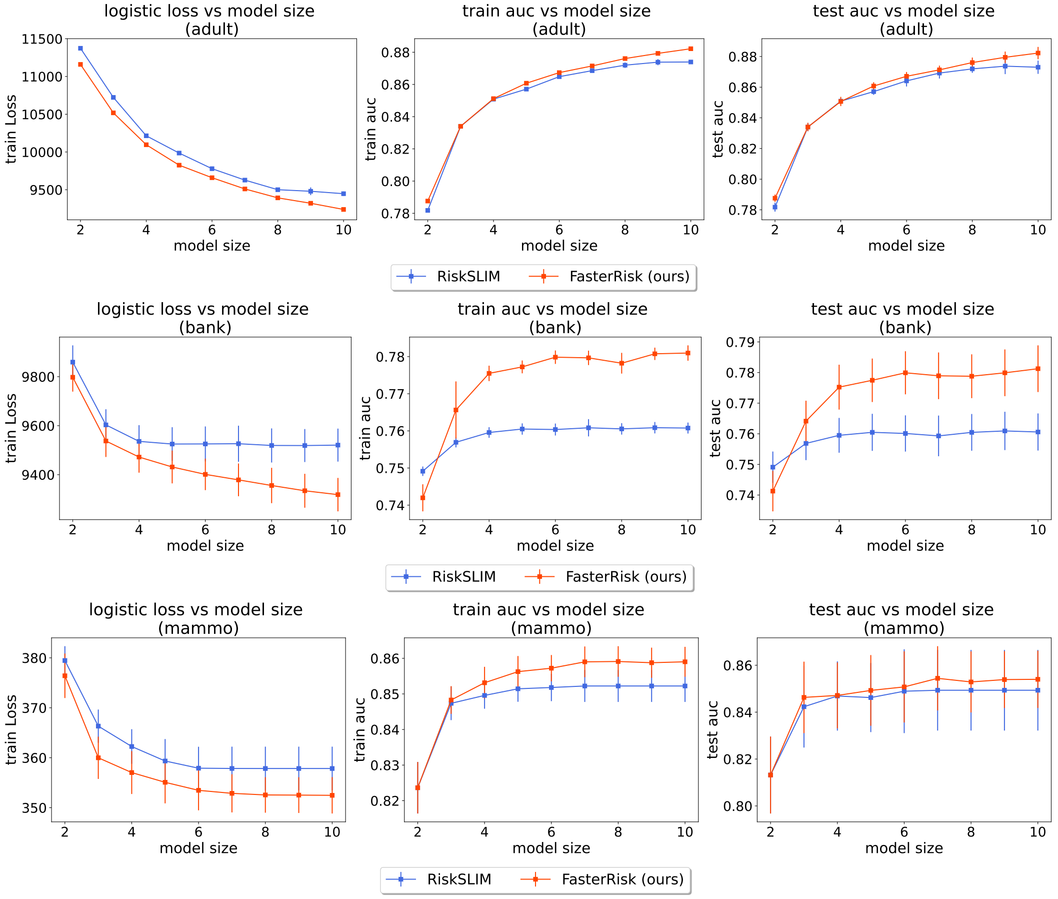

Figure 19: Comparison with the state-of-the-art baseline RiskSLIM (running for 1 hour) on the adult, bank, and mammo datasets. The left column is loss (lower is better), the middle column is training AUC (higher is better) and the right column is test AUC (higher is better).

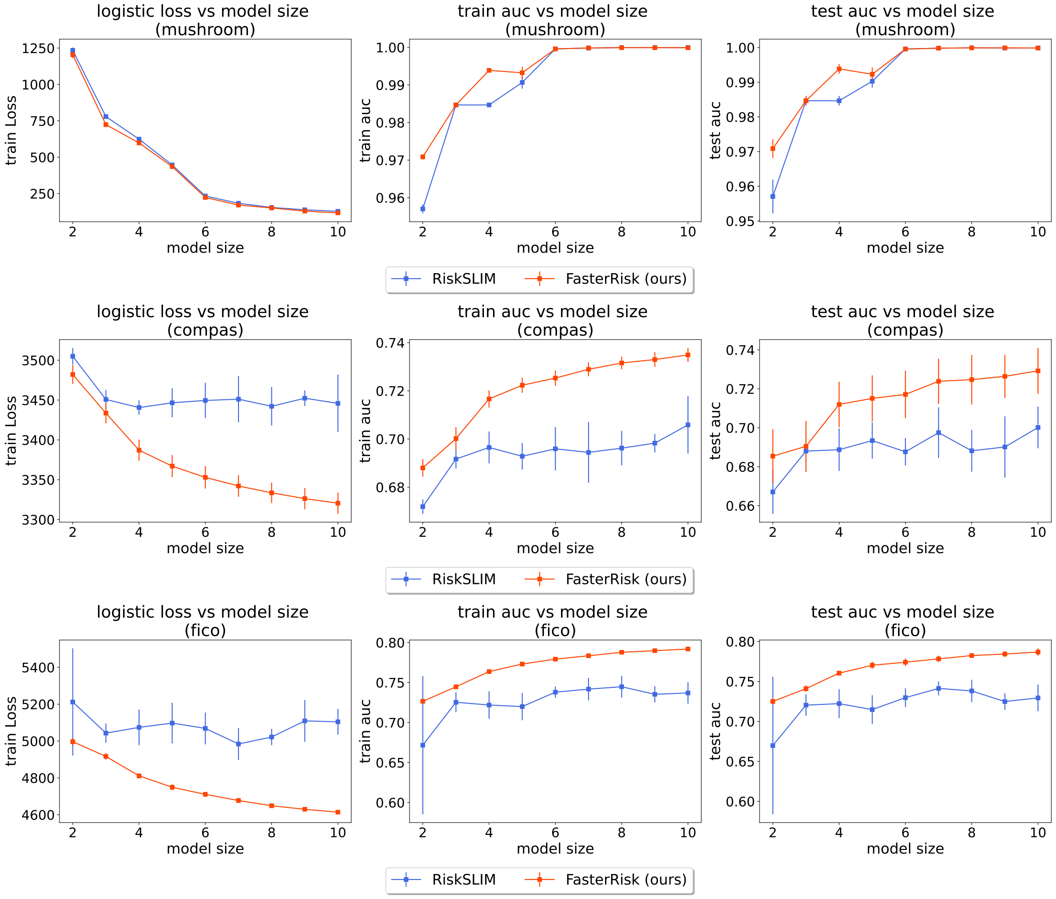

Figure 20: Comparison with the state-of-the-art baseline RiskSLIM (running for 1 hour) on the mushroom, COMPAS, and FICO datasets. The left column is loss (lower is better), the middle column is training AUC (higher is better) and the right column is test AUC (higher is better).

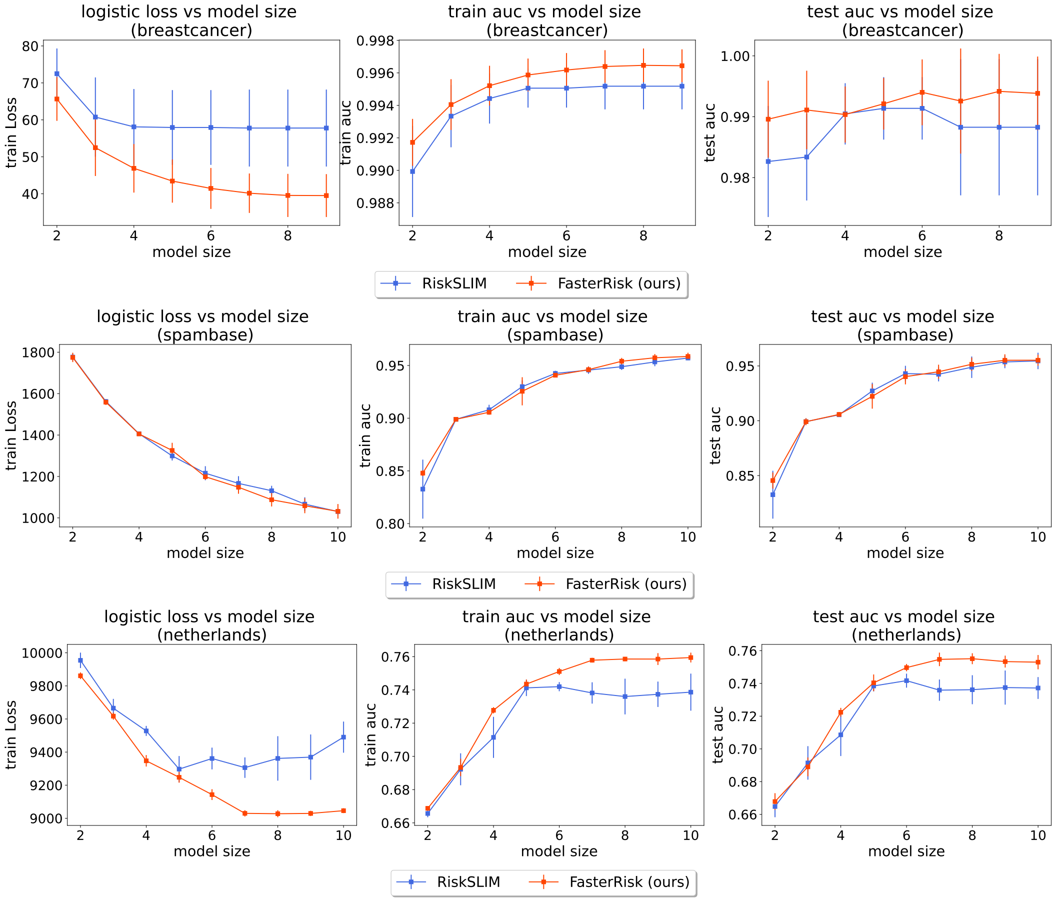

Figure 21: Comparison with the state-of-the-art baseline RiskSLIM (running for 1h) on the breastcancer, spambase, and Netherlands datasets. The left column is loss (lower is better), the middle column is training AUC (higher is better) and the right column is test AUC (higher is better).

E.8.2 Time Comparison of Running RiskSLIM for 1 hour

We plot the running time comparison between FasterRisk and RiskSLIM (with a time limit of 1 hour). The original time results with the 15-minute time limit are shown in Figure 5 and Figure 8.

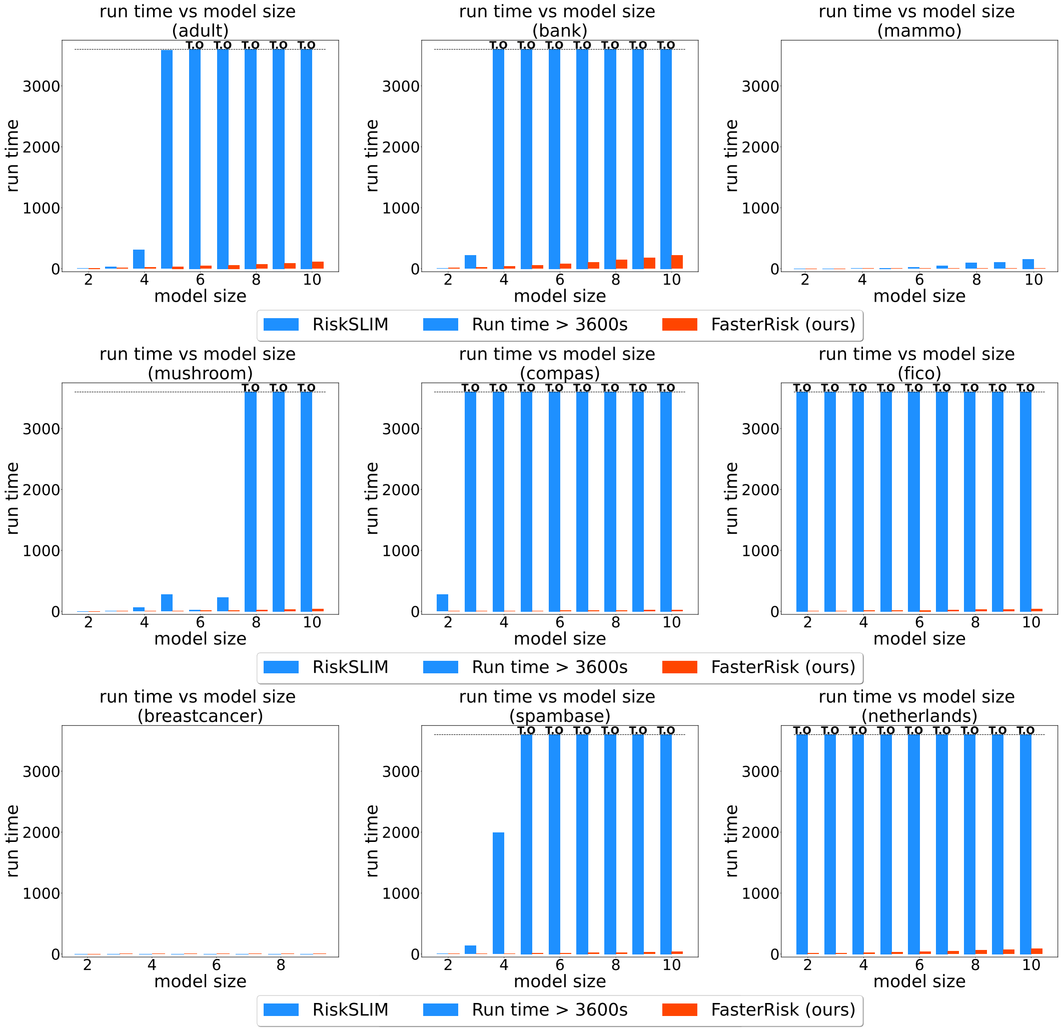

Figure 22: Runtime Comparison. Runtime (in seconds) versus model size for our method FasterRisk (in red) and the RiskSLIM (in blue). The shaded blue bars indicate cases that timed out at 1 hour. Breastcancer is a small dataset so it takes approximately 2 seconds for both algorithms. For more zoomed-in results on the breastcancer and mammo datasets, please refer to Figure 5 and Figure 8.

E.8.3 Solution Quality of Running RiskSLIM for Days

We report results of running the baseline RiskSLIM for 4 days. Due to this long running time demand on our servers, we could not run this experiment on all folds and all model sizes, so we only run on the 3rd fold of the 5-CV split. We plot the logistic loss progression over time.

The results are shown in Figure 23. We see that FasterRisk still achieves lower loss than RiskSLIM even after letting RiskSLIM run for 4 days, again because FasterRisk uses a larger model class. The only exceptions are on the Mushroom and the Spambase datasets, where the logistic losses are close to each other.

The major disadvantage of letting an algorithm run for days is that it is challenging to interact with the algorithm, because one has to wait for the results between interactions – ideally this process would be instantaneous. Furthermore, there could be memory issues for the MIP solver if we let it run for days since the branch-and-bound tree could become too large.

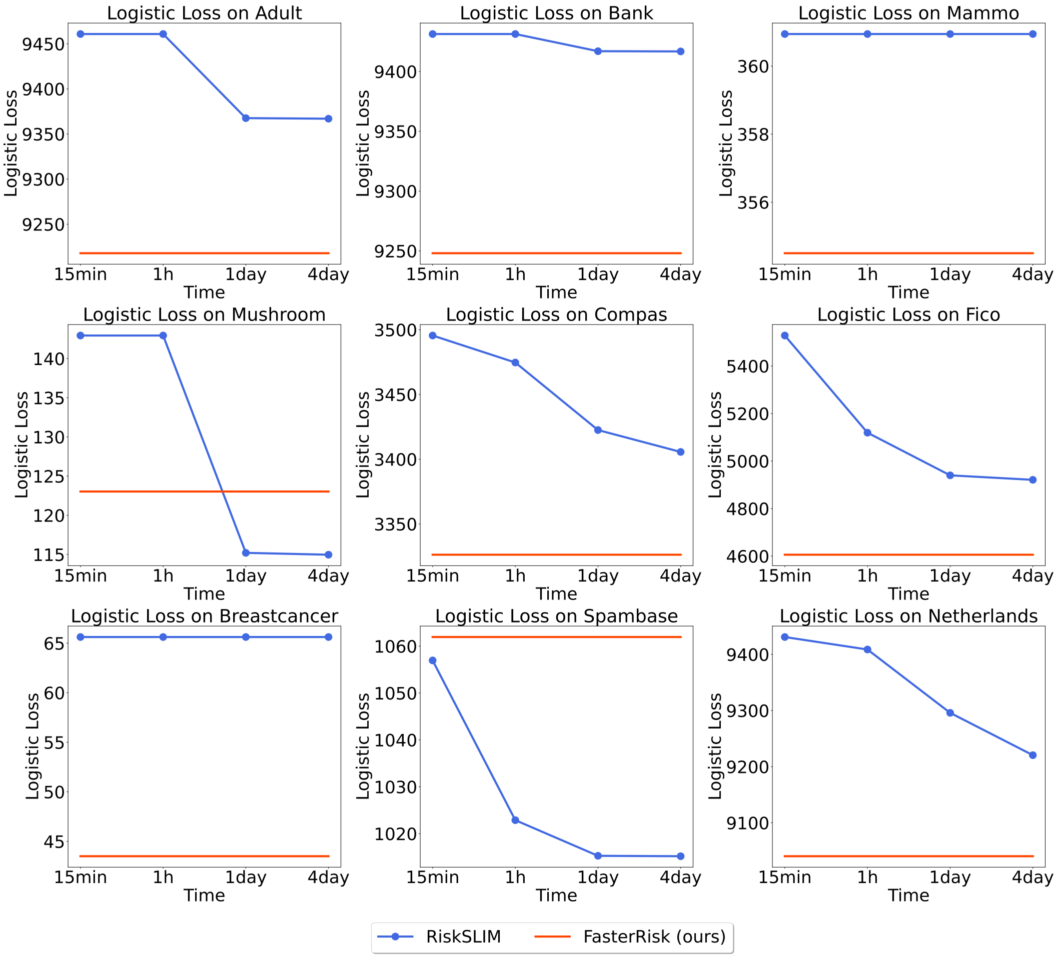

Figure 23: Curves of logistic loss vs. training time for the RiskSLIM model on the 3rd fold of the 5-CV split with model size equal to 10. All plots report logistic loss (lower is better).

E.9 Calibration Curves

The calibration curves for RiskSLIM and FasterRisk are shown in Figures 24-26 with model sizes equal to 3, 5, and 7, respectively. We use the sklearn package[4] from python to plot the figures. We use the default value for the number of bins (number of bins is 5) and the default strategy to define the widths of the bins (the strategy is "uniform").

The calibration curves on the breastcancer and mammo datasets are more spread out than those on the other datasets. This is perhaps due to the limited number of samples in these datasets (both datasets have fewer than 1000 samples in total; see Table 2), which increases the variance in the calculation of the curves.

On other datasets, both methods have good calibration curves, showing consistency between predicted score and actual risk. However, as shown in Figures 19-21, FasterRisk has higher AUC scores, which means our method has higher discrimination ability than RiskSLIM.

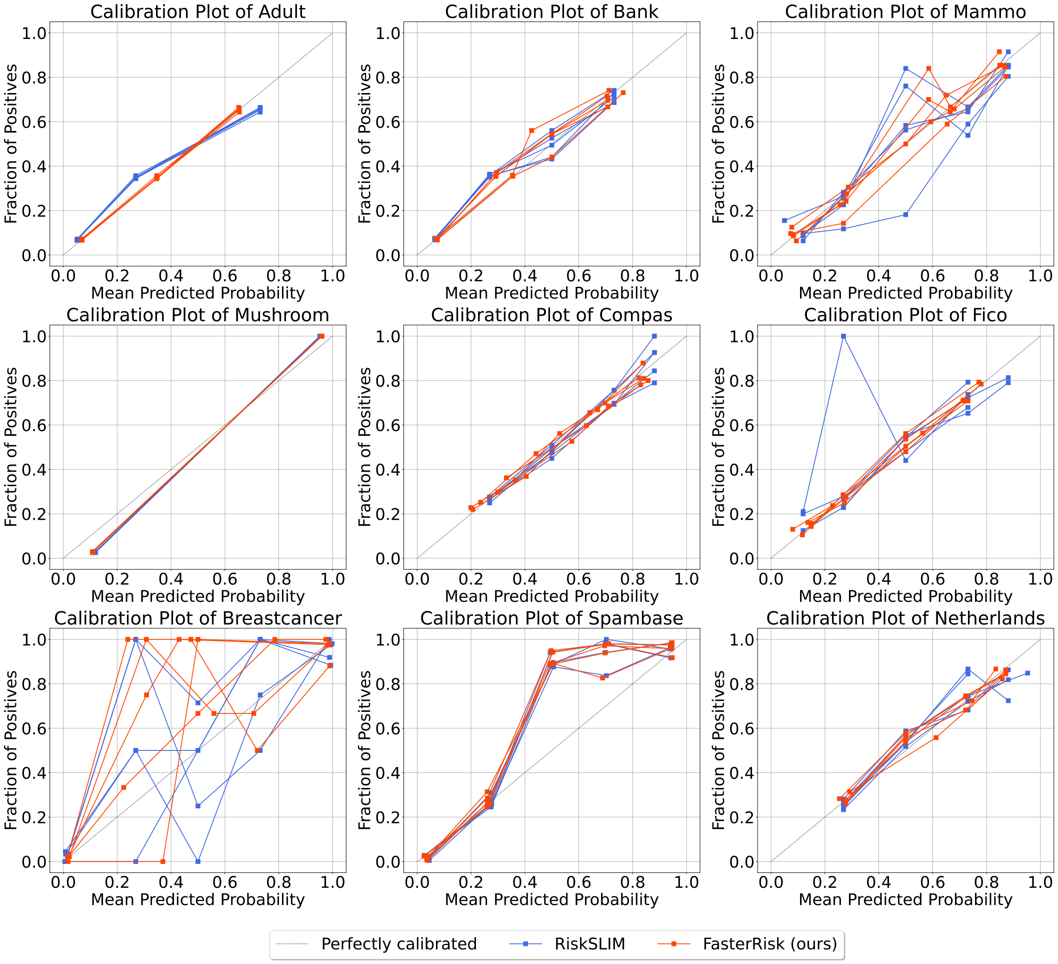

Figure 24: Calibration curves for RiskSLIM and FasterRisk with model size equal to 3. We plot results from each test fold. The FasterRisk model selected from the pool is that with the smallest logistic loss on the training set.

[4] `https://scikit-learn.org/stable/modules/generated/sklearn.calibration.calibration_curve.html`

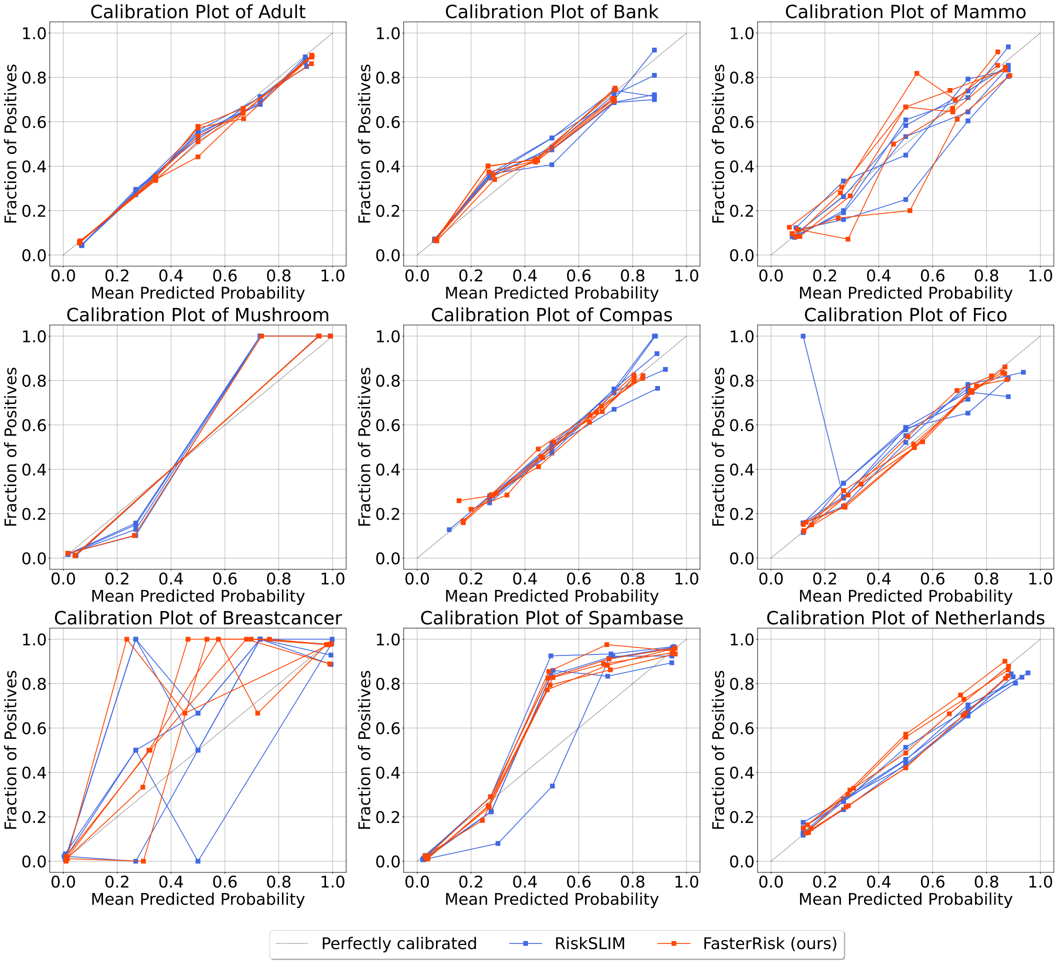

Figure 25: Calibration curves for RiskSLIM and FasterRisk with model size equal to 5. We plot results from each test fold. The FasterRisk model selected from the pool is that with the smallest logistic loss on the training set.

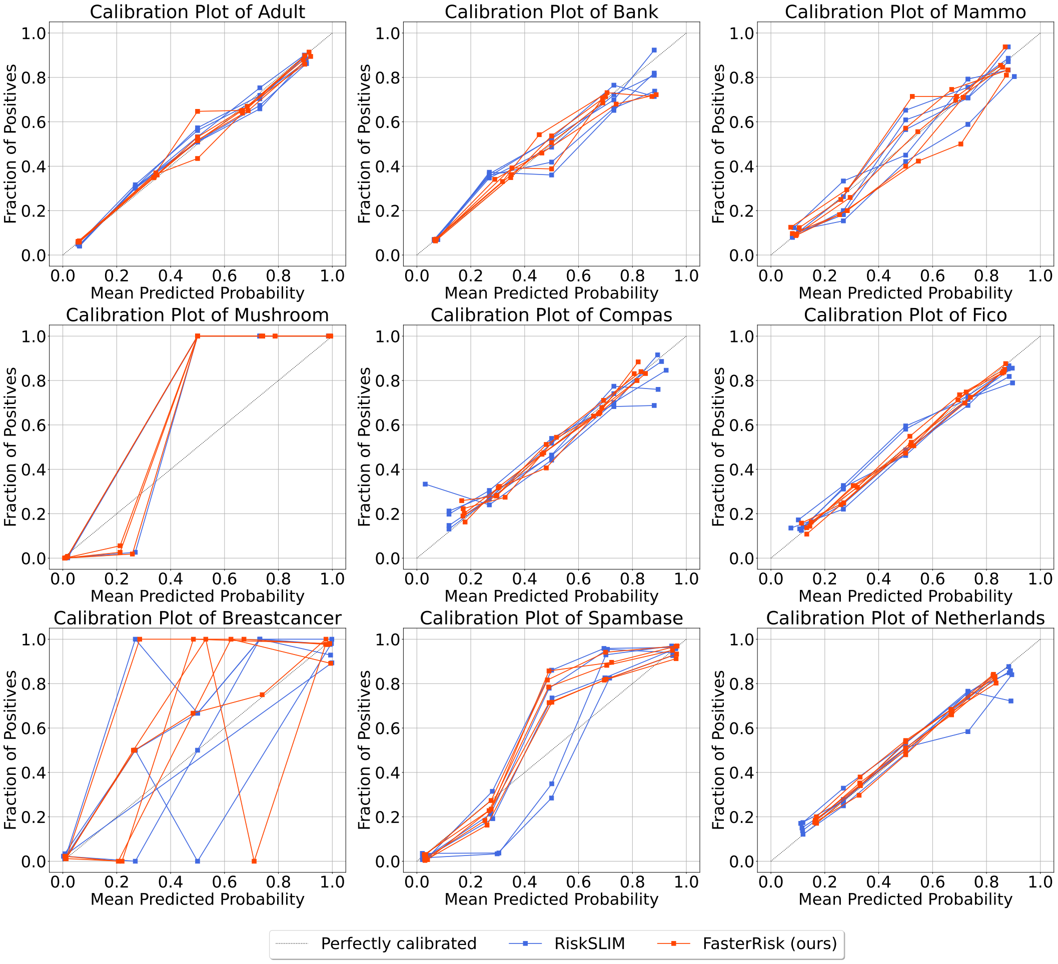

Figure 26: Calibration curves for RiskSLIM and FasterRisk with model size equal to 7. We plot results from each fold on the test set. The FasterRisk model selected from the pool is that with the smallest logistic loss on the training set.

E.10 Hyperparameter Perturbation Study

E.10.1 Perturbation Study on Beam Size B

We perform a perturbation study on the hyperparameter beam size B as mentioned in Appendix D.4. We set the beam size to 5, 10, and 15, respectively. The results are shown in Figures 27-29. The curves greatly overlap, confirming our previous claim that the performance is not particularly sensitive to the choice of B.

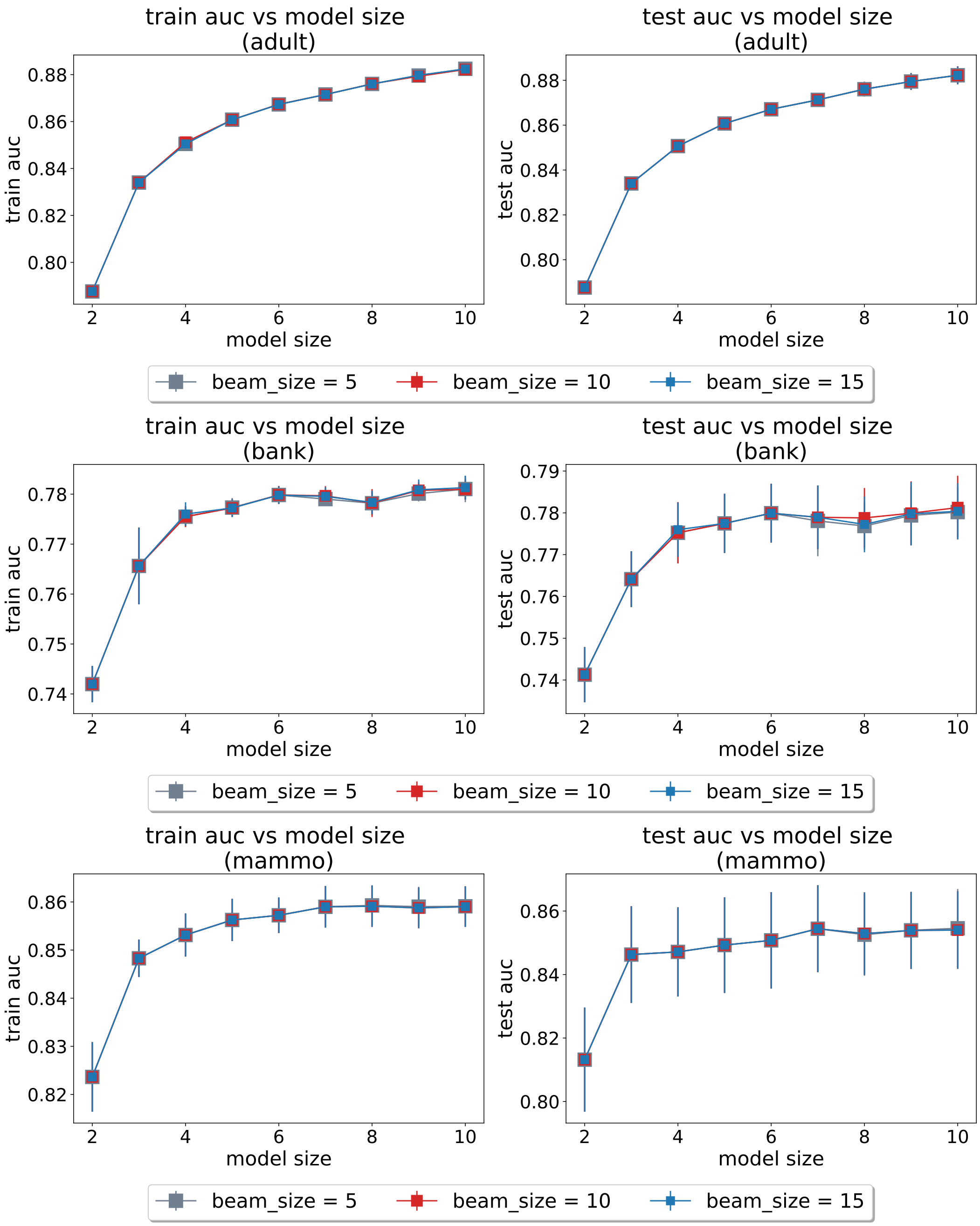

Figure 27: Perturbation study for beam size, B, on the adult, bank, and mammo datasets. The default value used in the paper is 10.

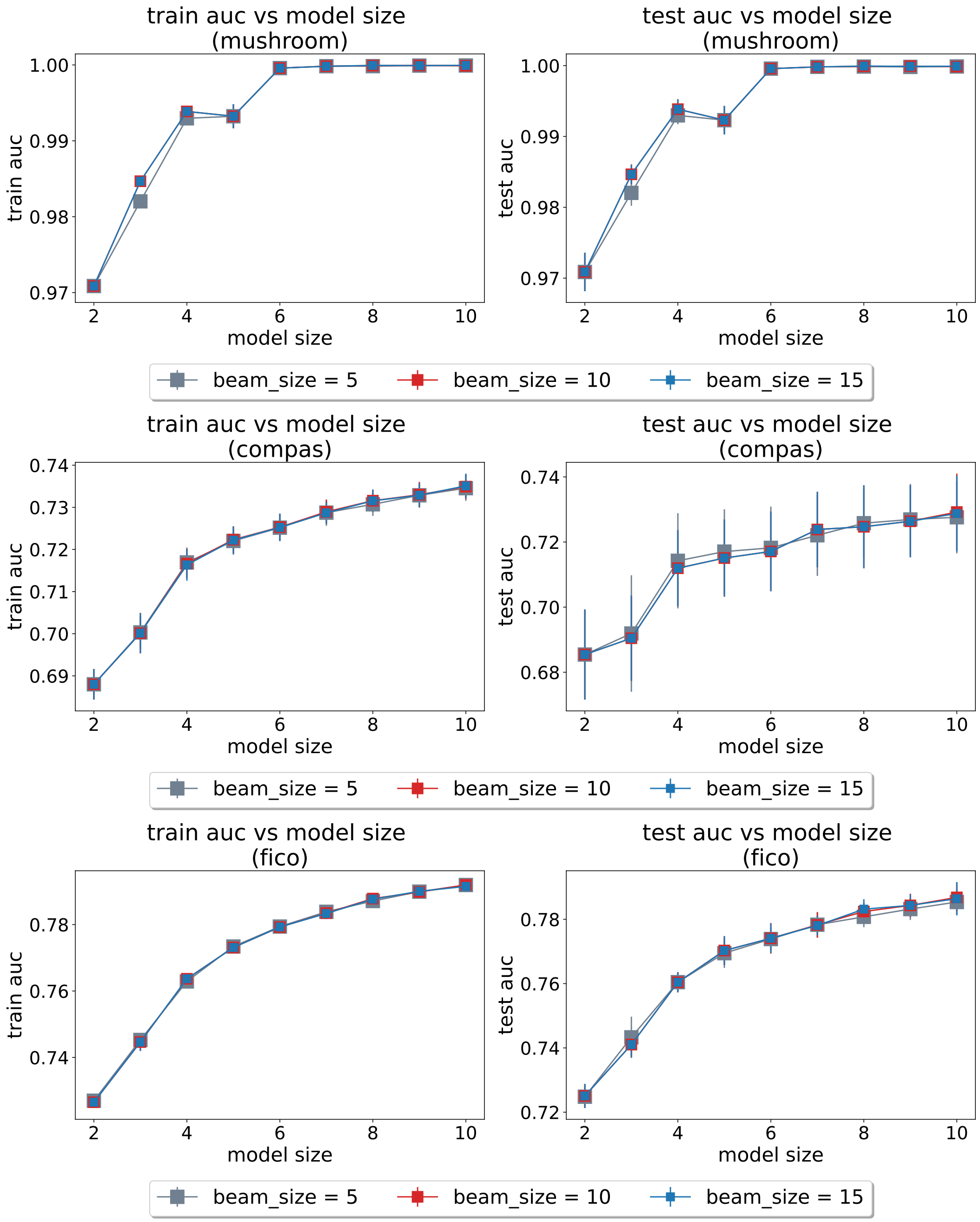

Figure 28: Perturbation study for beam size, B, on the mushroom, COMPAS, and FICO datasets. The default value used in the paper is 10.

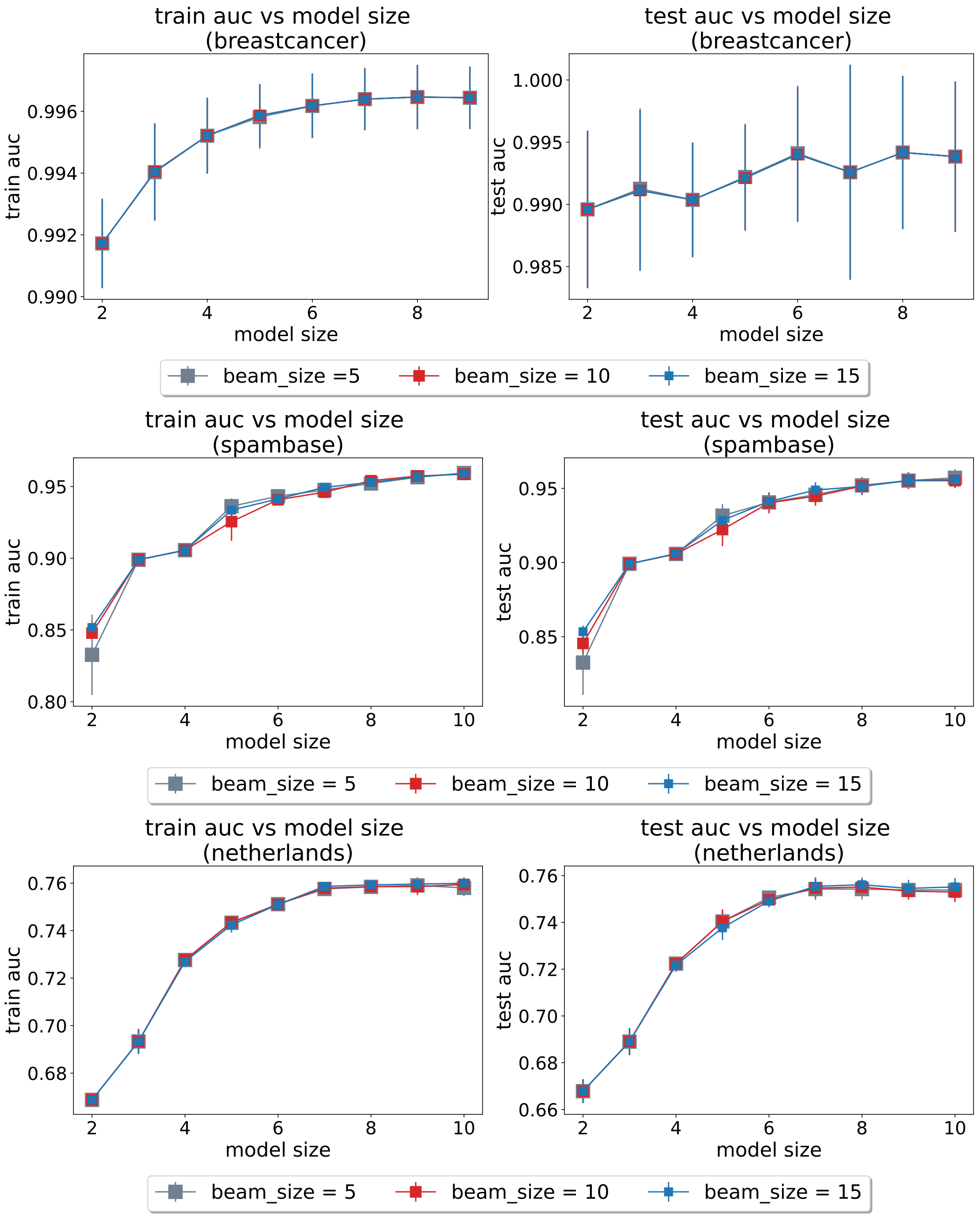

Figure 29: Perturbation study for beam size, B, on the breastcancer, spambase, and Netherlands datasets. The default value used in the paper is 10.

E.10.2 Perturbation Study on Tolerance Level ϵ for Sparse Diverse Pool

We perform a perturbation study on the hyperparameter tolerance level, ϵ, as mentioned in Appendix D.4. We set the tolerance level to 0.1, 0.3, and 0.5, respectively. The results are shown in Figures 30-32. The curves greatly overlap, confirming our previous claim that the performance is not particularly sensitive to the choice of value.

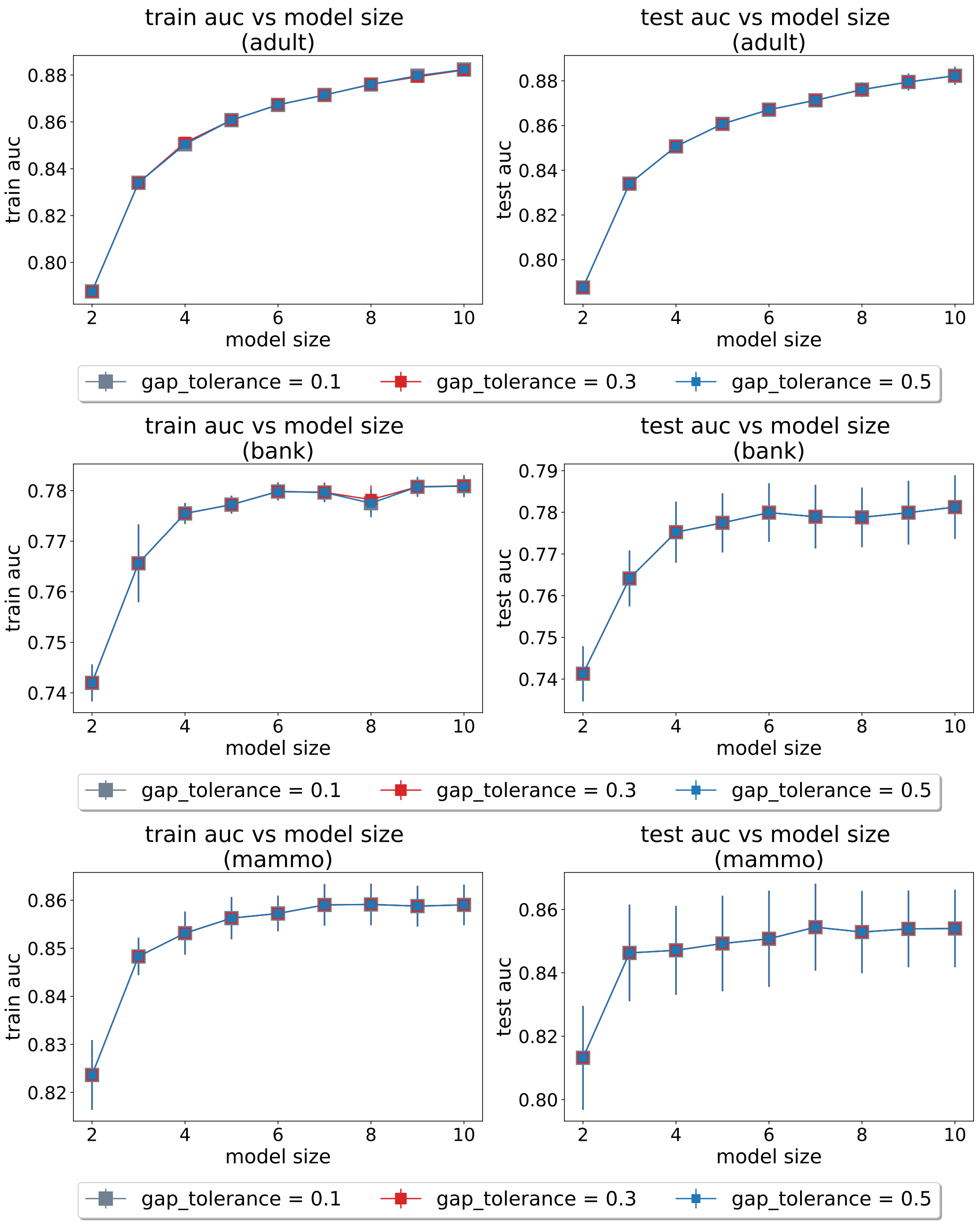

Figure 30: Perturbation study on tolerance level, ϵ, for sparse diverse pool on the adult, bank, and mammo datasets. The default value used in the paper is 0.3.

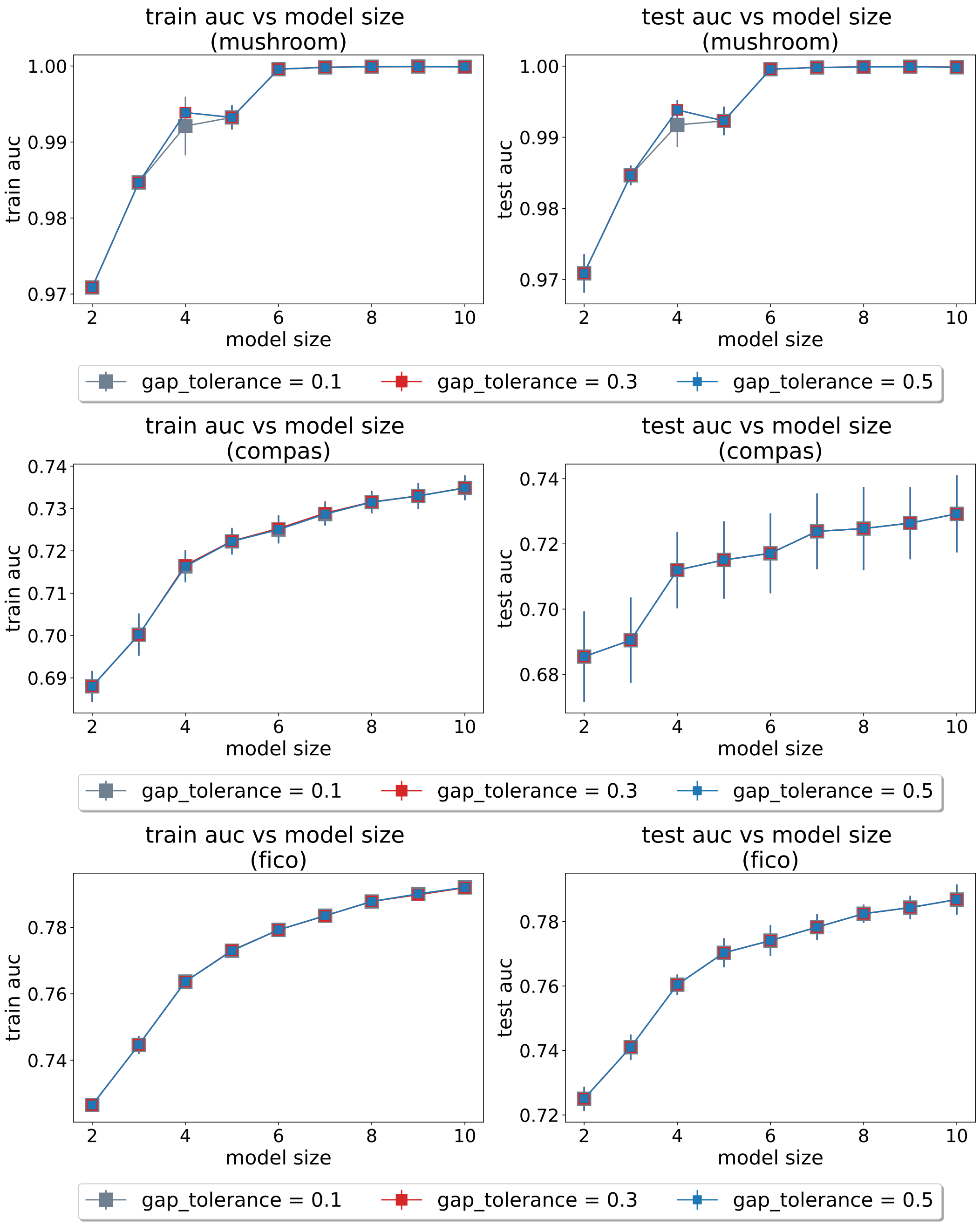

Figure 31: Perturbation study on tolerance level, ϵ, for sparse diverse pool on the mushroom, COMPAS and FICO datasets. The default value used in the paper is 0.3.

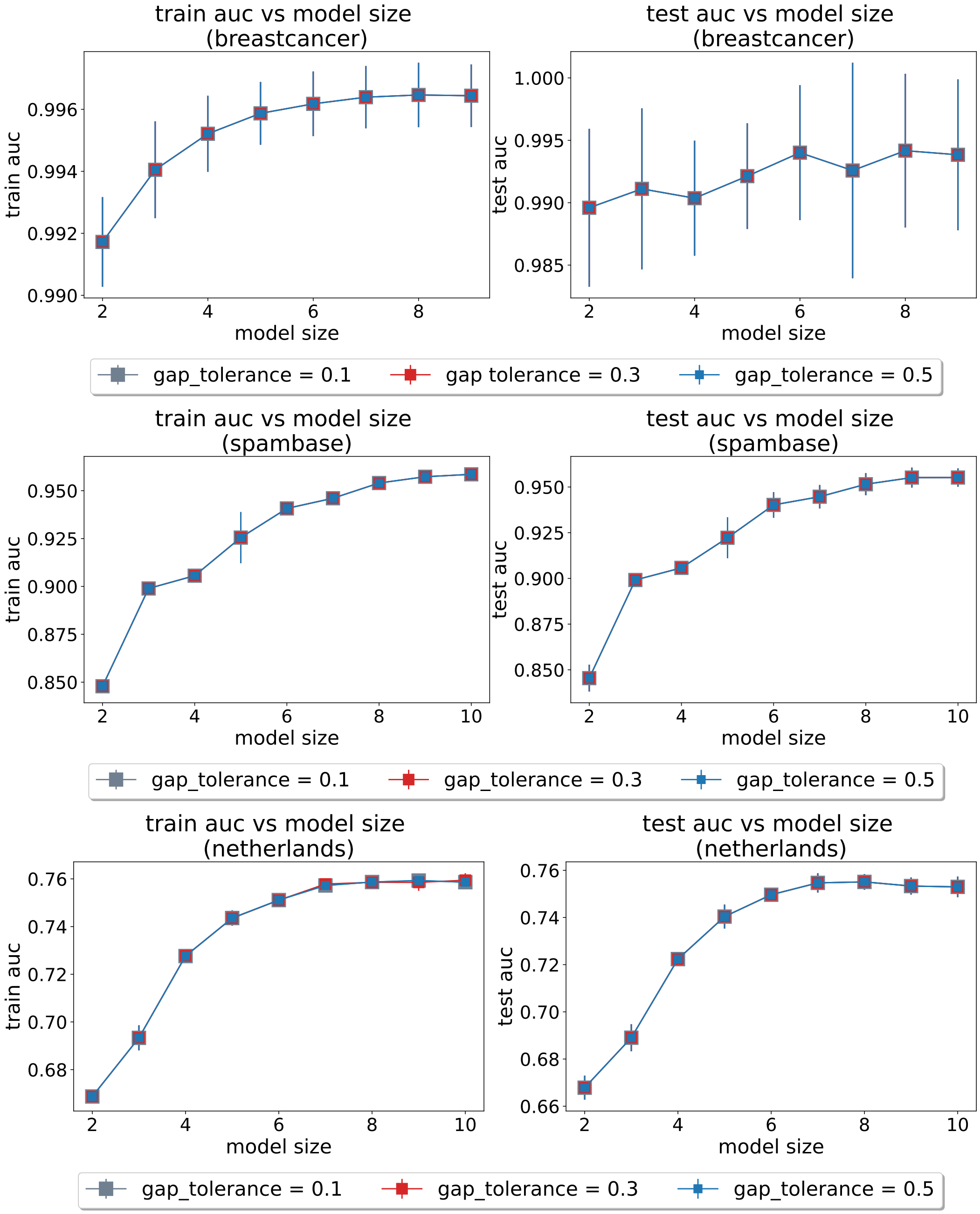

Figure 32: Perturbation study on tolerance level, ϵ, for sparse diverse pool on the breastcancer, spambase, and Netherlands datasets. The default value used in the paper is 0.3.

E.10.3 Perturbation Study on Number of Attempts T for Sparse Diverse Pool

We perform a perturbation study on the hyperparameter for the number of attempts, T, as mentioned in Appendix D.4. We have set the number of attempts to 35, 50, and 65, respectively. The results are shown in Figures 33-35. The curves greatly overlap, confirming our previous claim that the performance is not particularly sensitive to the choice of value for the hyperparameter.

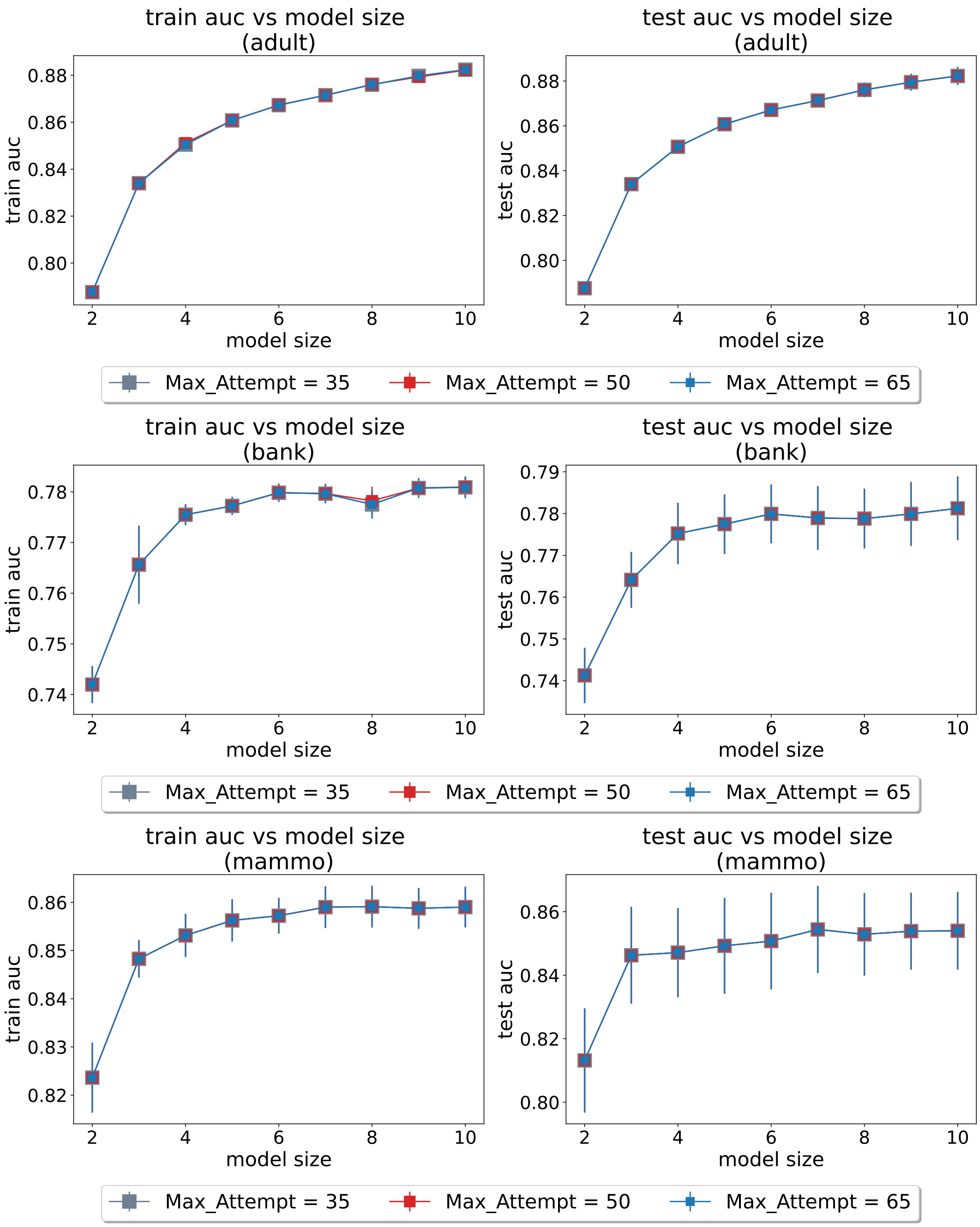

Figure 33: Perturbation study on number of attempts parameter, T, for sparse diverse pool on the adult, bank, and mammo datasets. The default value used in the paper is 50.

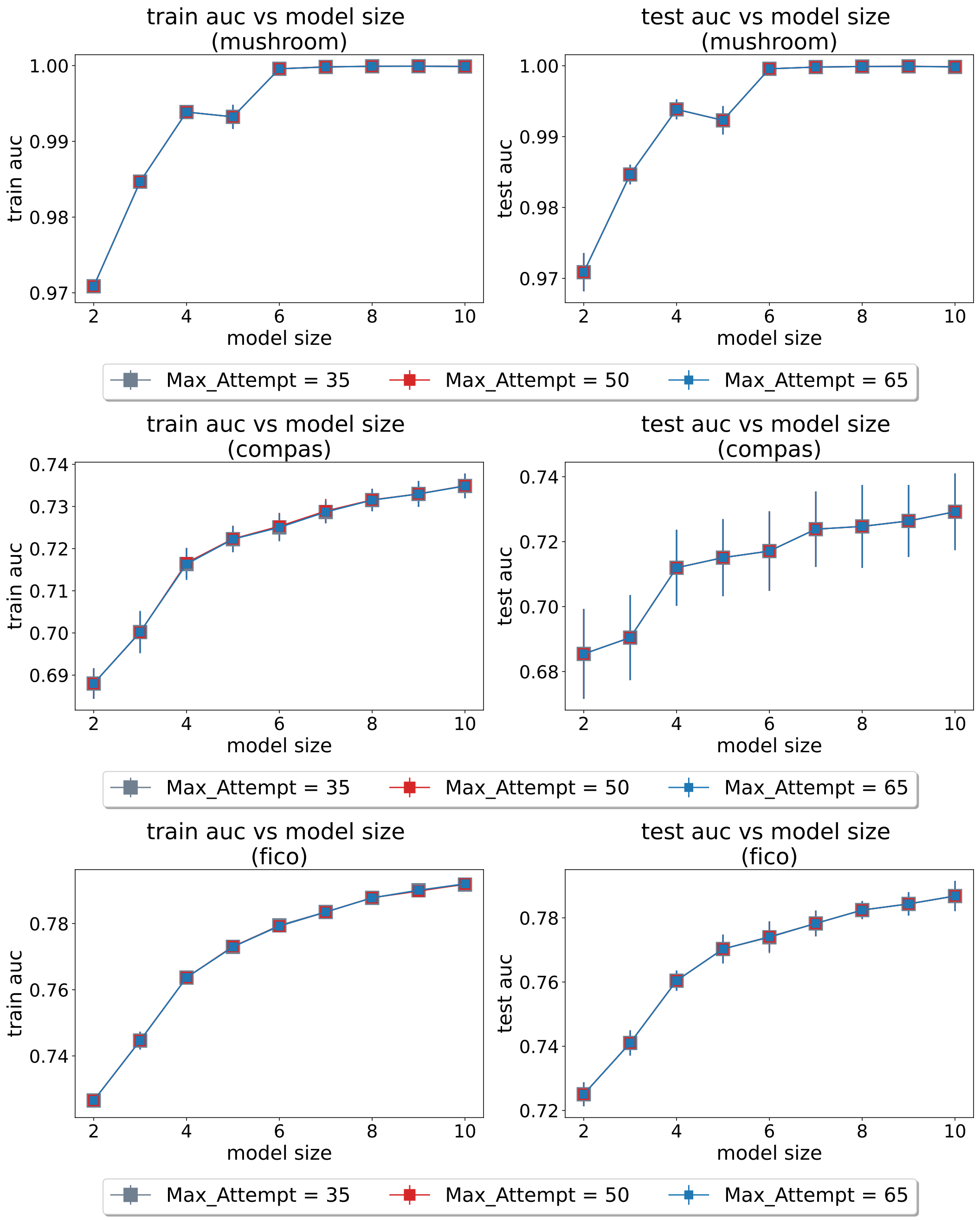

Figure 34: Perturbation study on number of attempts parameter, T, for sparse diverse pool on the mushroom, COMPAS, and FICO datasets. The default value used in the paper is 50.

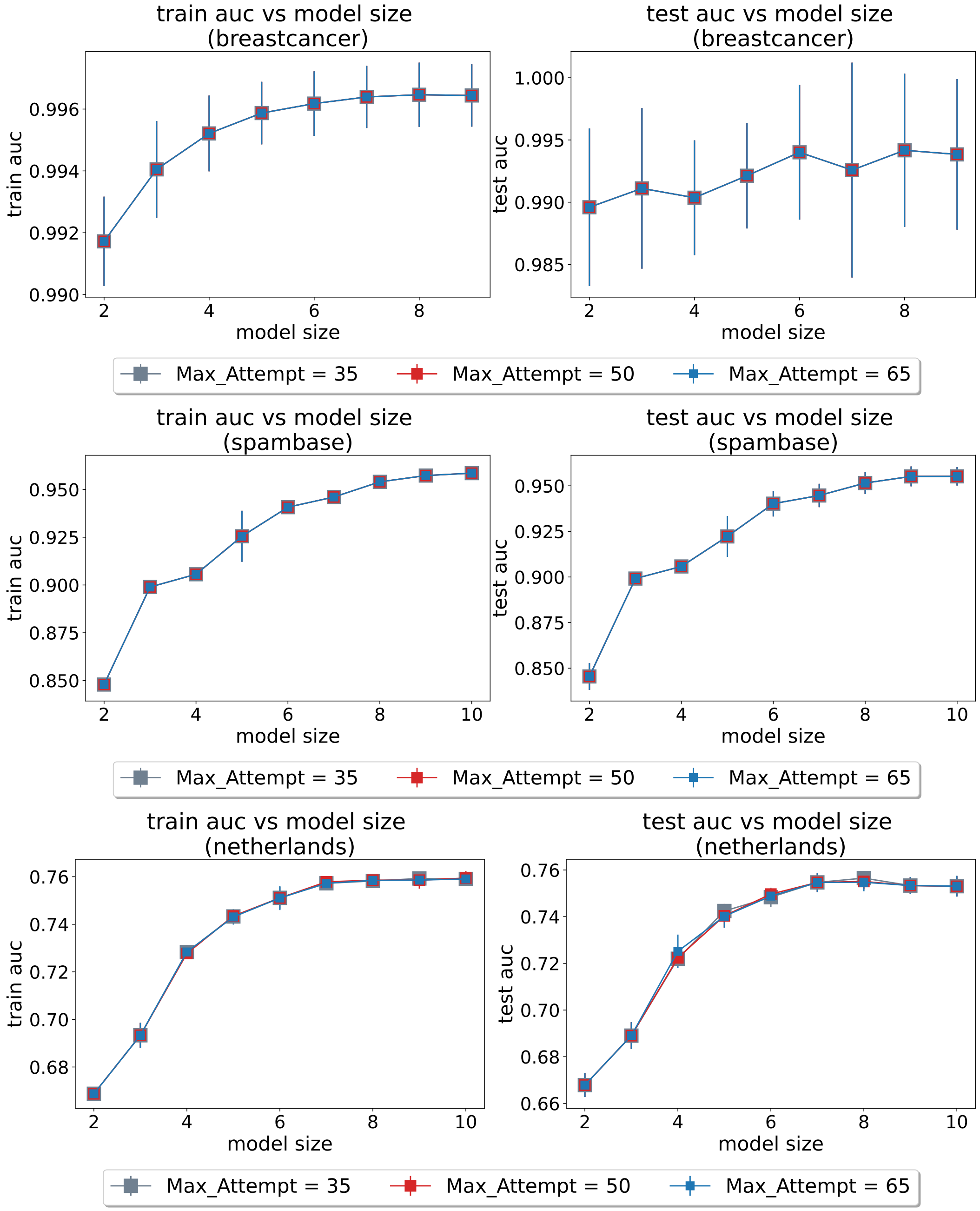

Figure 35: Perturbation study on number of attempts parameter, T, for sparse diverse pool on the breastcancer, spambase, and Netherlands datasets. The default value used in the paper is 50.

E.10.4 Perturbation Study on Number of Multipliers N_m

We perform a perturbation study on the hyperparameter for the number of multipliers, N_m, as mentioned in Appendix D.4. We have set the number of multipliers to 10, 20, and 30, respectively. The results are shown in Figures 36-38. The curves greatly overlap, confirming our previous claim that the performance is not particularly sensitive to the choice of values for N_m.

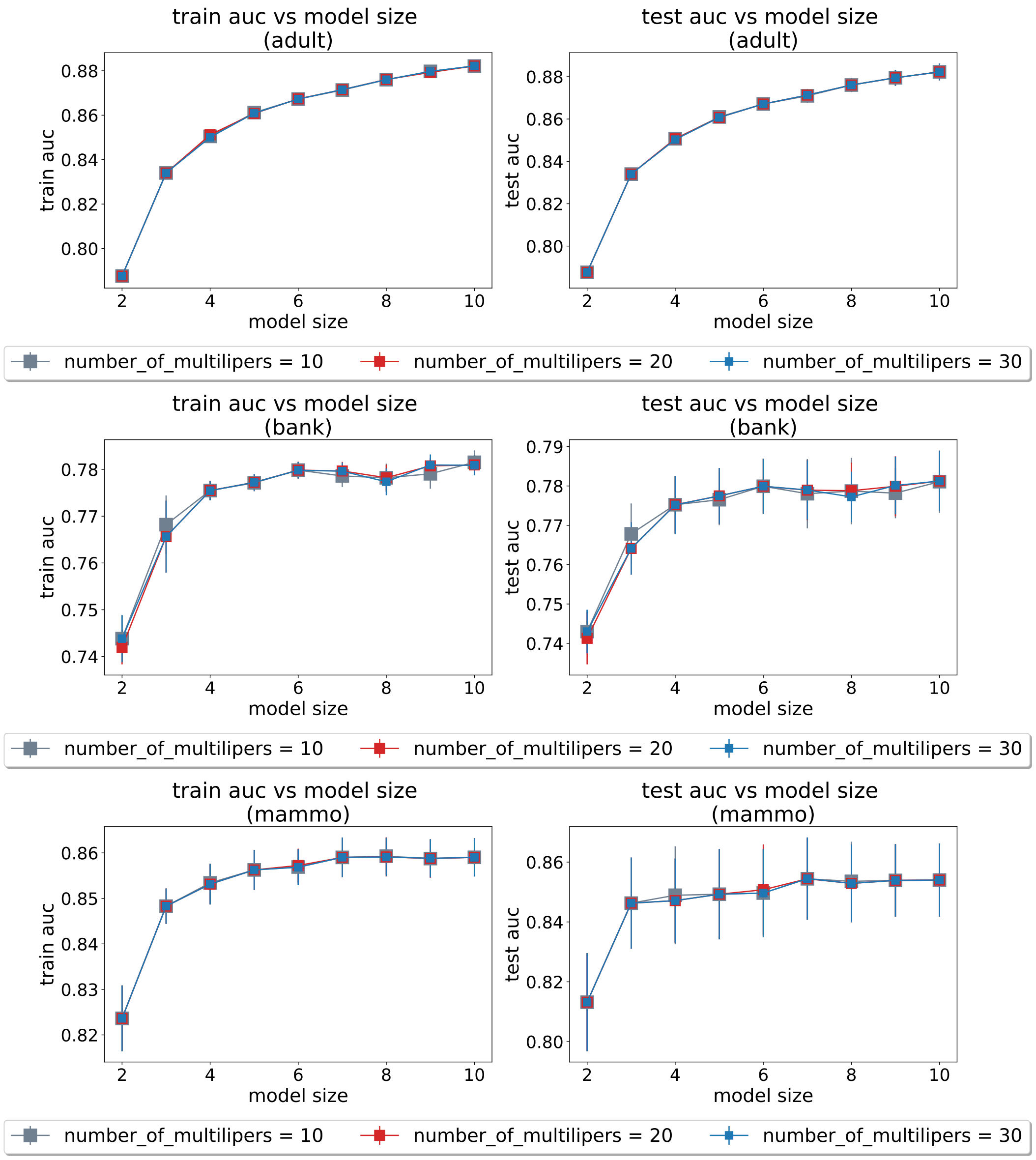

Figure 36: Perturbation study on number of multipliers, N_m, on the adult, bank, and mammo datasets. The default value used in the paper is 20.

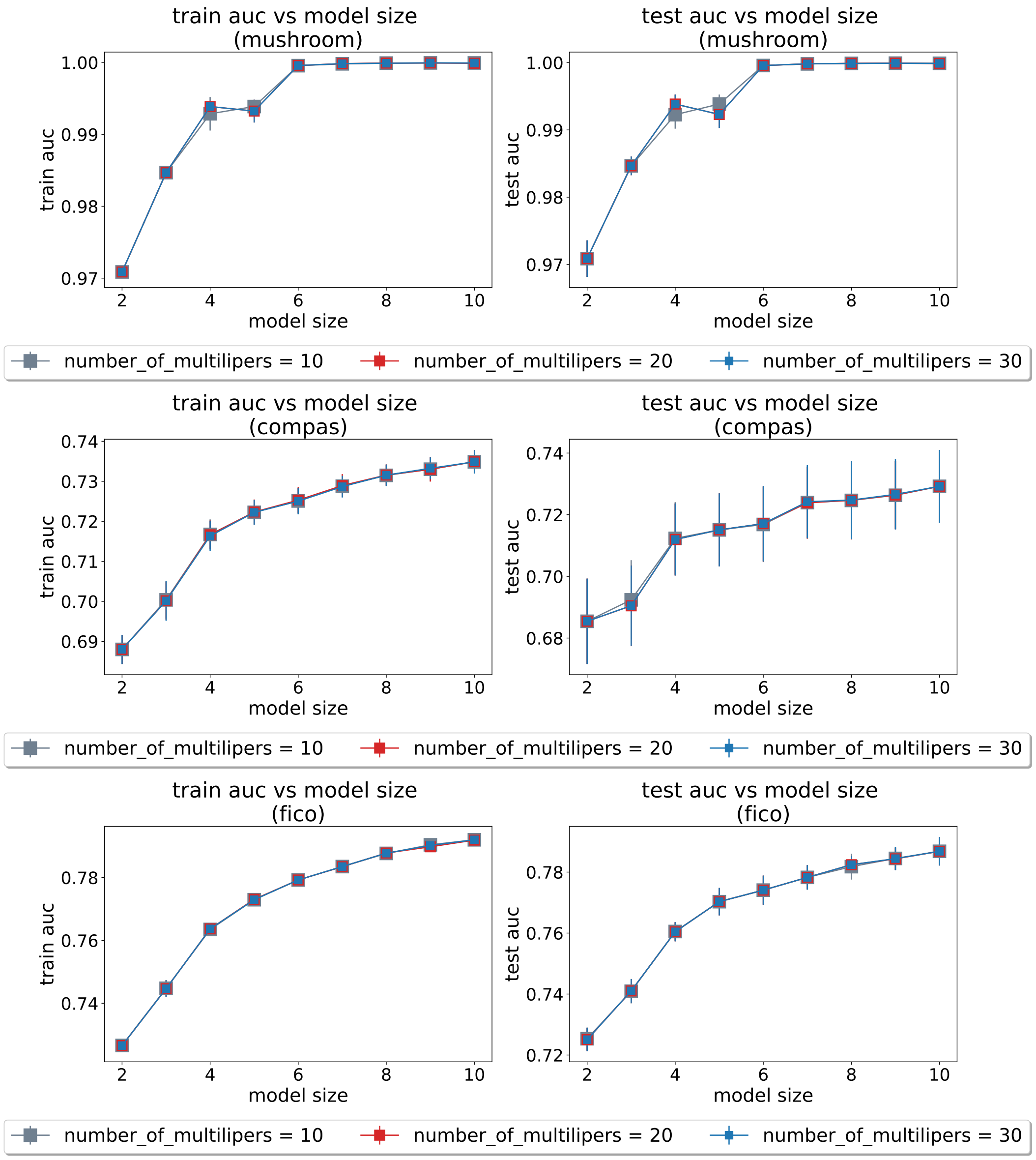

Figure 37: Perturbation study on number of multipliers, N_m, for sparse diverse pool on the mushroom, COMPAS and FICO datasets. The default value used in the paper is 20.

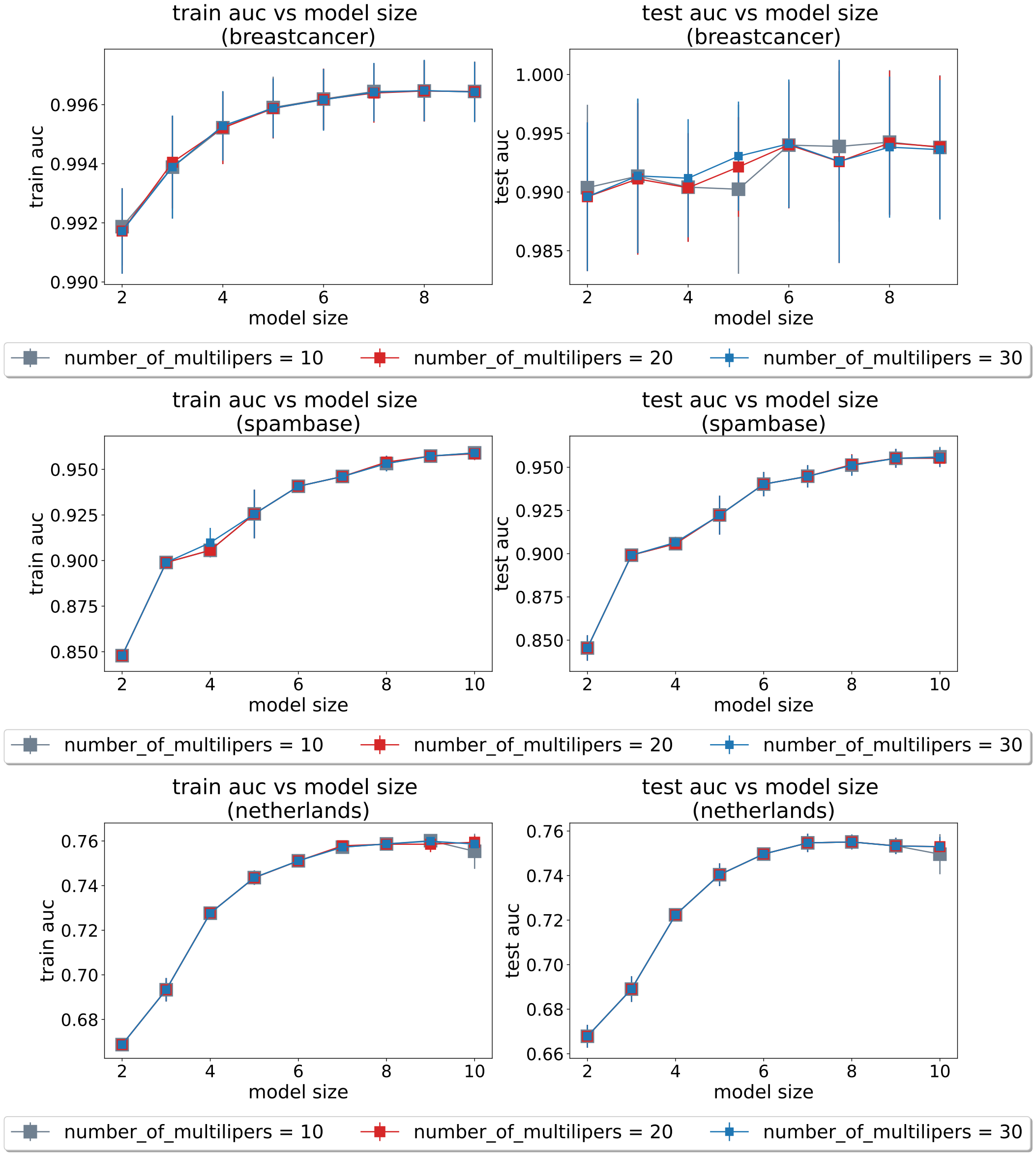

Figure 38: Perturbation study on number of multipliers, N_m, for sparse diverse pool on the breastcancer, spambase, and Netherlands datasets. The default value used in the paper is 20.

E.11 Comparison with Baseline AutoScore

We compare with the baseline AutoScore [44]. We set the number of features from 2 to 10 and use all other hyperparameters in the default setting. The results of training AUC and test AUC are shown in Figures 39-41. The plots of RiskSLIM are from experiments where we let RiskSLIM run for 1 hour. FasterRisk outperforms both RiskSLIM and AutoScore.

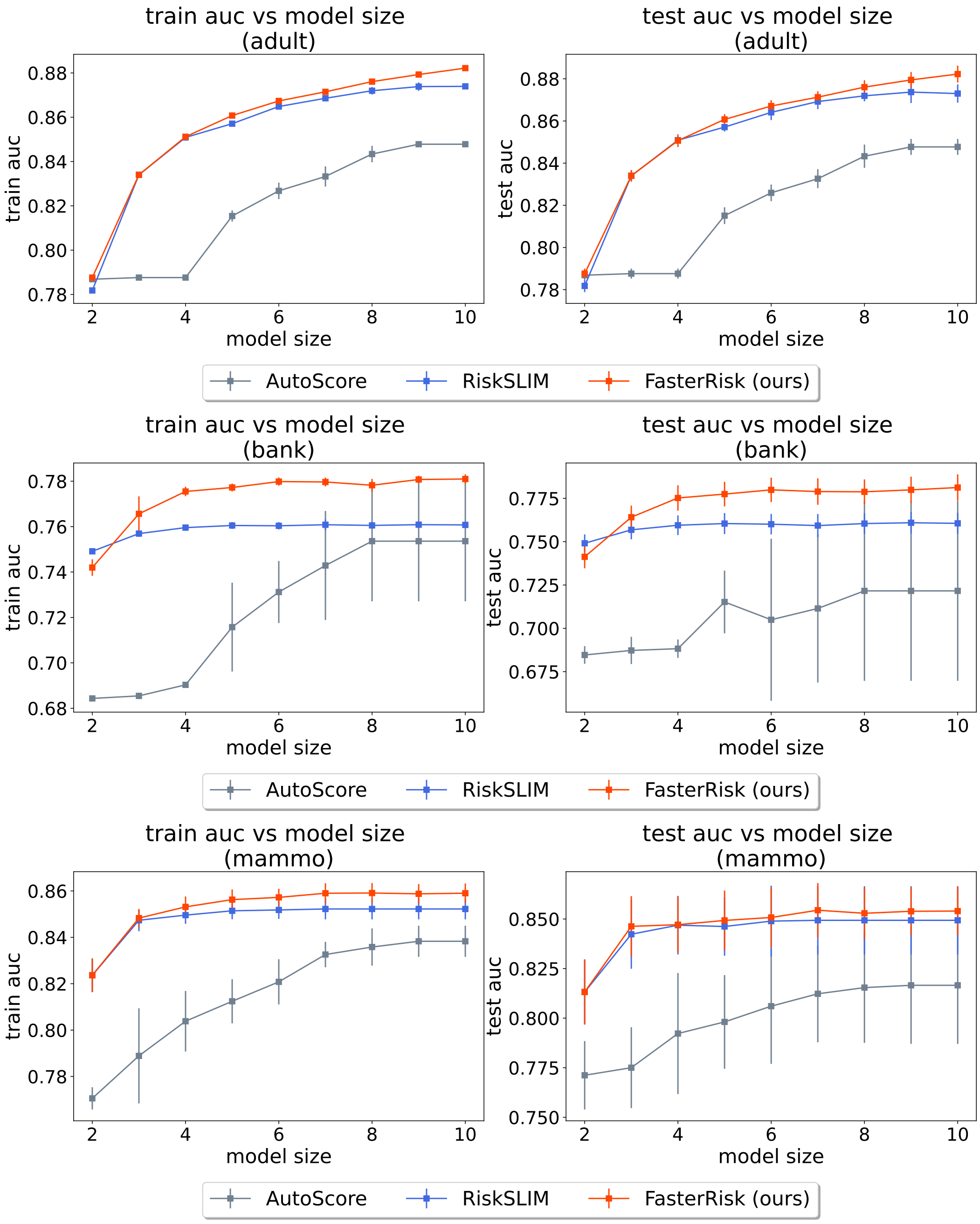

Figure 39: Comparison with the new baseline AutoScore on the adult, bank, and mammo datasets. The left column is training AUC (higher is better), and the right column is test AUC (higher is better).

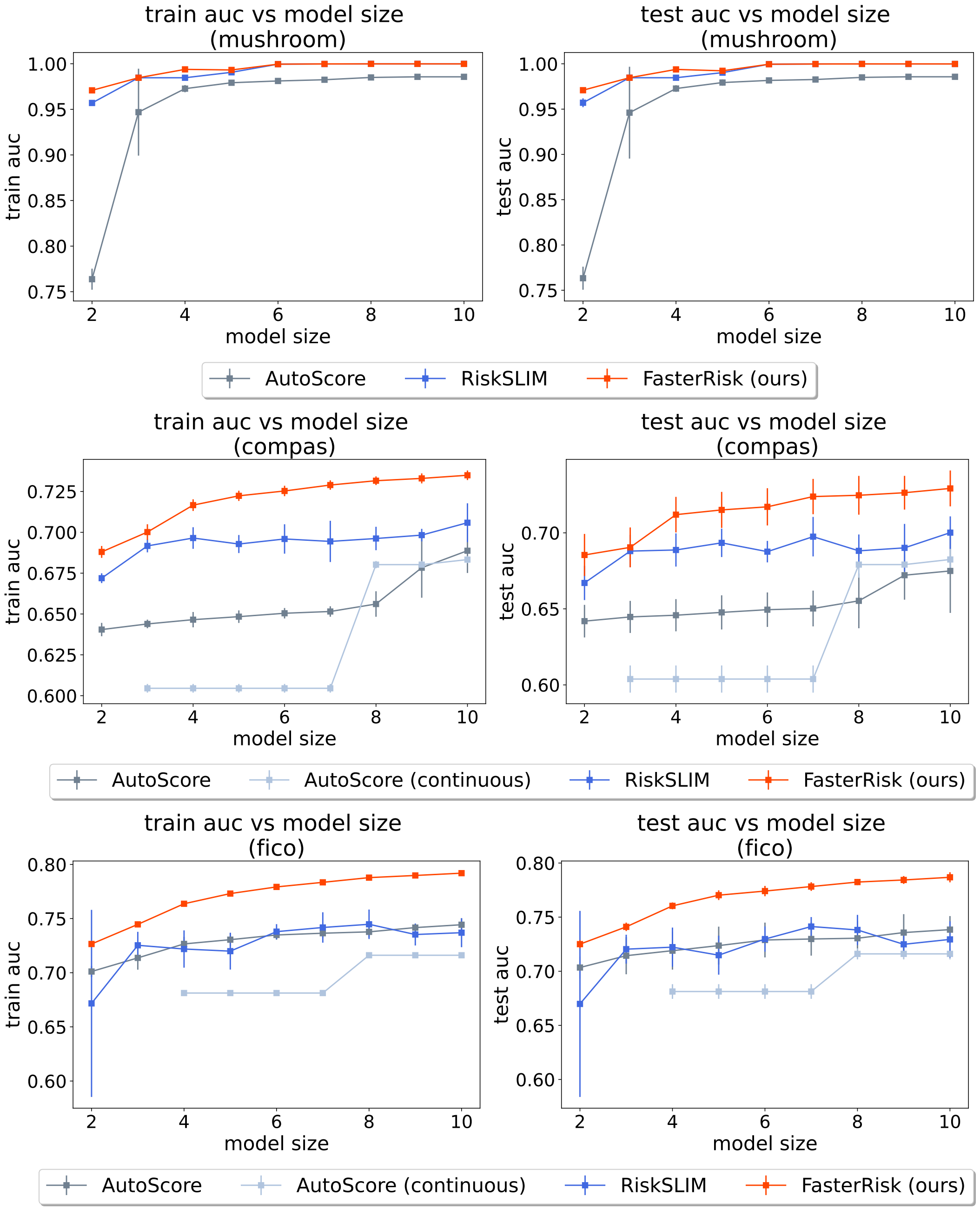

Figure 40: Comparison with the new baseline on the mushroom, Compas, and FICO datasets. The AutoScore (continuous) baseline is another method where AutoScore is applied to the original continuous features instead of the binary features as detailed in Appendix D.1. Not every model size can be obtained by the AutoScore (continuous) method. The left column is training AUC (higher is better), and the right column is test AUC (higher is better).

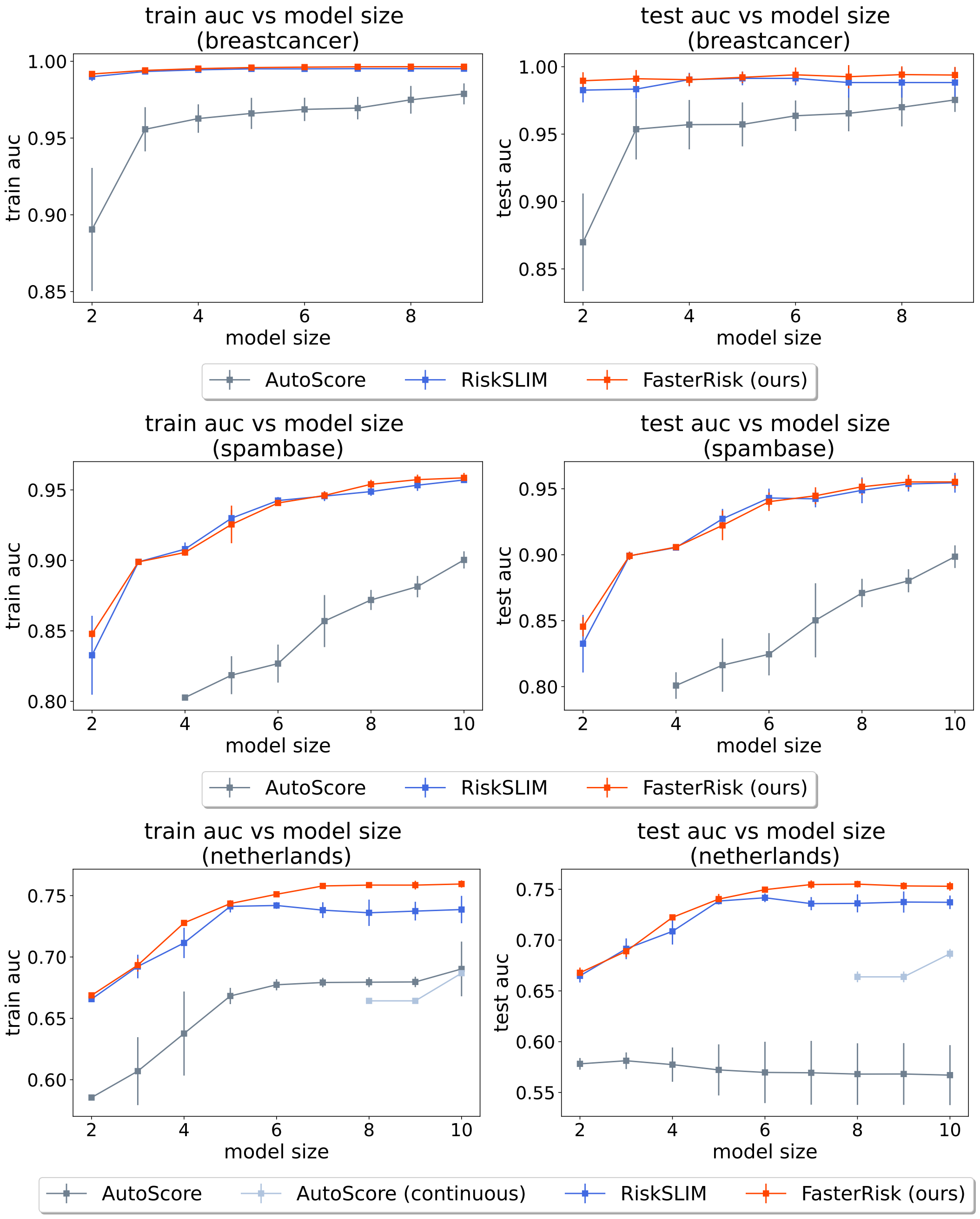

Figure 41: Comparison with the new baseline on the breastcancer, spambase, and Netherlands datasets. The AutoScore (continuous) baseline is another method where AutoScore is applied to the original continuous features instead of the binary features as detailed in Appendix D.1. Not every model size can be obtained by the AutoScore (continuous) method. The left column is training AUC (higher is better), and the right column is test AUC (higher is better).

F Additional Risk Score Models

We provide additional risk score models for the readers to inspect.

Appendix F.1 shows risk scores with different model sizes on different datasets.

Appendix F.2 shows different risk scores with the same size from the diverse pool of solutions.

Specifically, Appendix F.2.1 shows different risk scores on the bank dataset (financial application), Appendix F.2.2 shows different risk scores on the mammo dataset (medical application), and Appendix F.2.3 shows different risk scores on the Netherlands dataset (criminal justice application).

F.1 Risk Score Models with Different Sizes

For model size $= 3$, please see Tables 3-11.

For model size $= 5$, please see Tables 12-20.

For model size $= 7$, please see Tables 21-29.

We also include a large model with size $= 10$ on the FICO dataset, please see Table 30.

1.	no high school diploma	-4 points	...
2.	high school diploma only	-2 points	+ ...
3.	married	4 points	+ ...
		SCORE	=

SCORE	-4	-2	0	2	4
RISK	1.2%	4.1%	13.1%	34.7%	65.3%

Table 3: FasterRisk model for the adult dataset, predicting salary> 50K.

1.	Call in Second Quarter	-2 points	...
2.	Previous Call Was Successful	4 points	+ ...
3.	Employment Indicator < 5100	4 points	+ ...
		SCORE	=

SCORE	-2	0	2	6	8
RISK	2.8%	6.5%	14.5%	50.0%	70.8%

Table 4: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call.

1.	Irregular Shape	4 points	...
2.	Circumscribed Margin	-5 points	+ ...
3.	Age ≥ 60	3 points	+ ...
		SCORE	=

SCORE	-5	-2	-1	2
RISK	8.2%	20.1%	26.2%	50.0%

SCORE	3	4	7
RISK	58.5%	66.6%	84.9%

Table 5: FasterRisk model for the mammo dataset, predicting malignancy of a breast lesion.

1.	odor=almond	-5 points		...
2.	odor=anise	-5 points	+	...
3.	odor=none	-5 points	+	...
		SCORE	=	

SCORE	-5	0
RISK	10.8%	96.0%

Table 6: FasterRisk model for the mushroom dataset, predicting whether a mushroom is poisonous.

1.	prior_counts ≤ 2	-4 points		...
2.	prior_counts ≤ 7	-4 points	+	...
3.	age ≤ 31	4 points	+	...
		SCORE	=	

SCORE	-8	-4	0	4
RISK	23.6%	44.1%	67.0%	83.9%

Table 7: FasterRisk model for the COMPAS dataset, predicting whether individuals are arrested within two years of release.

1.	MSinceMostRecentInqexcl7days≤ 0	3 points		...
2.	ExternalRiskEstimate≤ 70	5 points	+	...
3.	ExternalRiskEstimate≤ 79	5 points	+	...
		SCORE	=	

SCORE	0	3	5	8	$\geq$ 10
RISK	13.7%	24.0%	33.4%	50.0%	$\geq$ 61.3%

Table 8: FasterRisk model for the FICO dataset, predicting whether an individual will default on a loan.

1.	Clump Thickness	×3 points		...
2.	Uniformity of Cell Size	×5 points	+	...
3.	Bare Nuclei	×3 points	+	...
		SCORE	=	

SCORE	$\leq$ 33	36	39	42	45
RISK	$\leq$ 3.3%	6.1%	10.8%	18.6%	30.1

SCORE	48	51	54	57	$\geq$ 60
RISK	67.0%	77.6%	85.5%	91.0	$\geq$ 94.5%

Table 9: FasterRisk model for the breastcancer dataset, predicting whether there is breast cancer using a biopsy.

1.	WordFrequency_Remove	×5 points		...
2.	WordFrequency_HP	×-2 points	+	...
3.	CharacterFrequency_$	×5 points	+	...
		SCORE	=	

SCORE	$\leq$ -4	-3	-2	-1	0
RISK	$\leq$0.4%	1.3%	3.7%	10.2%	25.2%

SCORE	1	2	3	4	$\geq$ 5
RISK	50.0%	74.8%	89.8%	96.3%	$\geq$ 98.7%

Table 10: FasterRisk model for the spambase dataset, predicting if an e-mail is spam.

1.	previous case ≤ 20	-5 points		...
2.	previous case ≤ 10 or previous case ≥ 21	-4 points	+	...
3.	# of previous penal cases≤ 3	-2 points	+	...
		SCORE	=	

SCORE	≤ -9	-7	-6	0
RISK	$\leq$ 50%	74.6%	83.4%	99.2%

Table 11: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years.

1.	no high school diploma	-4 points		...
2.	high school diploma only	-2 points	+	...
3.	age 22 to 29	-2 points	+	...
4.	any capital gains	3 points	+	...
5.	married	4 points	+	...
		SCORE	=	

SCORE	<-4	-3	-2	-1	0
RISK	<1.3%	2.4%	4.4%	7.8%	13.6%

SCORE	1	2	3	4	7
RISK	22.5%	35.0%	50.5%	65.0%	92.2%

Table 12: FasterRisk model for the adult dataset, predicting salary> 50K. This table has already been shown in the main paper.

1.	Call in Second Quarter	-2 points		...
2.	Previous Call Was Successful	4 points	+	...
3.	Previous Marketing Campaign Failed	-1 points	+	...
4.	Employment Indicator > 5100	-5 points	+	...
5.	3 Month Euribor Rate ≥ 100	-2 points	+	...
		SCORE	=	

SCORE	≤-5	-4	-3	-2	-1
RISK	$\leq$ 11.2%	15.1%	20.1%	26.2%	33.4%
SCORE	0	1	2	3	4
RISK	41.5%	50.0%	58.5%	66.6%	73.8%

Table 13: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call.

1.	Oval Shape	-2 points		…
2.	Irregular Shape	4 points	+	…
3.	Circumscribed Margin	-5 points	+	…
4.	Spiculated Margin	2 points	+	…
5.	Age ≥ 60	3 points	+	
		SCORE	=	

SCORE	-7	-5	-4	-3	-2	-1
RISK	6.0%	10.6%	13.8%	17.9%	22.8%	28.6%

SCORE	0	1	2	3	4	≥ 5
RISK	35.2%	42.4%	50.0%	57.6%	64.8%	71.4%

Table 14: FasterRisk model for the mammo dataset, predicting malignancy of a breast lesion. This table has already been shown in the main paper.

1.	odor=almond	-5 points	...
2.	odor=anise	-5 points	+ ...
3.	odor=none	-5 points	+ ...
4.	odor=foul	5 points	+ ...
5.	gill size=broad	-3 points	+ ...
		SCORE	=

SCORE	-8	-5	-3	$\geq$2
RISK	1.62%	26.4%	73.6%	>99.8%

Table 15: FasterRisk model for the mushroom dataset, predicting whether a mushroom is poisonous. This table has already been shown in the main paper.

1.	prior_counts ≤ 7	-5 points	...
2.	prior_counts ≤ 2	-5 points	+ ...
3.	prior_counts ≤ 0	-3 points	+ ...
4.	age ≤ 33	4 points	+ ...
5.	age ≤ 23	5 points	+ ...
		SCORE	=

SCORE	$\leq$ -10	-9	-6	-5	-4
RISK	$\leq$25.9%	29.4%	41.3%	45.6%	50.0%

SCORE	-2	-1	3	4	9
RISK	58.7%	62.8%	77.3%	80.2%	$\geq$ 90.7%

Table 16: FasterRisk model for the COMPAS dataset, predicting whether individuals are arrested within two years of release.

1.	MSinceMostRecentInqexcl7days ≤ -8	-4 points	...
2.	MSinceMostRecentInqexcl7days ≤ 0	2 points	+ ...
3.	NumSatisfactoryTrades ≤ 12	2 points	+ ...
4.	ExternalRiskEstimate ≤ 70	3 points	+ ...
5.	ExternalRiskEstimate ≤ 79	3 points	+ ...
		SCORE	=

SCORE	-2	0	1	2	3
RISK	6.7%	13.2%	18.2%	24.4%	32.0%

SCORE	4	5	6	8	10
RISK	40.7%	50.0%	59.3%	75.5%	86.8%

Table 17: FasterRisk model for the FICO dataset, predicting whether an individual will default on a loan. -8 means a missing value on the FICO dataset.

1.	Clump Thickness	×5 points	...
2.	Uniformity of Cell Size	×4 points	+ ...
3.	Marginal Adhesion	×3 points	+ ...
4.	Bare Nuclei	×4 points	+ ...
5.	Normal Nucleoli	×3 points	+ ...
		SCORE	=

SCORE	$\leq$ 55	60	65	70	75
RISK	$\leq$ 8.6%	14.6%	23.5%	35.7%	50.0

SCORE	80	85	90	95	$\geq$ 100
RISK	64.3%	76.5%	85.4%	91.4	$\geq$ 95.0%

Table 18: FasterRisk model for the breastcancer dataset, predicting whether there is breast cancer using a biopsy.

1.	WordFrequency_Remove	×5 points		...
2.	WordFrequency_Free	×2 points		...
3.	WordFrequency_0	×5 points	+	...
4.	WordFrequency_HP	×-2 points	+	...
5.	WordFrequency_George	×-2 points	+	...
		SCORE	=	

SCORE	$\leq$ -4	-3	-2	-1	0	1	2	3	4	$\geq$ 5
RISK	$\leq$0.6%	1.6%	4.4%	11.4%	26.4%	50.0%	73.6%	88.6%	95.6%	$\geq$ 98.4%

Table 19: FasterRisk model for the spambase dataset, predicting if an e-mail is spam.

1.	previous case ≤ 20	-5 points		...
2.	previous case ≤ 10 or previous case ≥ 21	-3 points	+	...
3.	# of previous penal cases ≤ 2	-2 points	+	...
4.	age in years ≤ 38.06	1 points	+	...
5.	age at first penal case ≤ 22.63	1 points	+	...
		SCORE	=	

SCORE	$\leq$-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	$\leq$ 23.8%	35.8%	50.0%	64.2%	76.2%	85.1%	91.1%	94.8%	97.0%	98.3%	99.1%	99.5%

Table 20: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years.

1.	Age 22 to 29	-2 points		...
2.	High School Diploma Only	-2 points	+	...
3.	No High school Diploma	-4 points		...
4.	Married	4 points	+	...
5.	Work Hours Per Week < 50	-2 points	+	...
6.	Any Capital Gains	3 points	+	...
7.	Any Capital Loss	2 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1	0	2	3	4	7
RISK	$\leq$0.8%	1.4%	2.6%	4.6%	8.1%	14.0%	35.3%	50.0%	64.7%	91.9%

Table 21: FasterRisk model for the adult dataset, predicting salary$>$ 50K.

1.	Blue Collar Job	-1 points		...
2.	Call in Second Quarter	-2 points	+	...
3.	Previous Call Was Successful	3 points	+	...
4.	Previous Marketing Campaign Failed	-1 points	+	...
5.	Employment Indicator > 5100	-5 points	+	...
6.	Consumer Price Index ≥ 93.5	1 points	+	...
7.	3 Month Euribor Rate ≥ 100	-1 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1
RISK	$\leq$ 7.9%	11.5%	16.3%	22.7%	30.6%
SCORE	0	1	2	3	4
RISK	39.9%	50.0%	60.1%	69.4%	77.3%

Table 22: FasterRisk model for the bank dataset, predicting if a person opens bank account after marketing call.

1.	Lobular Shape	2 points		...
2.	Irregular Shape	5 points	+	...
3.	Circumscribed Margin	-4 points	+	...
4.	Obscured Margin	-1 points	+	...
5.	Spiculated Margin	1 points	+	...
6.	Age < 30	-5 points	+	...
7.	Age ≥ 60	3 points	+	...
		SCORE	=	

SCORE	$\leq$-1	0	1	2	3
RISK	19.8%	25.9%	33.2%	41.3%	50.0%
SCORE	4	5	6	8	9
RISK	58.7%	66.8%	74.1%	85.2%	89.1%

Table 23: FasterRisk model for the mammo dataset, predicting malignancy of a breast lesion.

1.	odor=anise	-5 points		...
2.	odor=none	-5 points	+	...
3.	odor=foul	5 points	+	...
4.	gill size=narrow	4 points	+	...
5.	stalk surface above ring=grooves	2 points	+	...
6.	spore print color=green	5 points	+	...
		SCORE	=	

SCORE	-5	0	2	4	$\geq$ 5
RISK	0.5%	50.0%	89.2%	98.6%	99.5%

Table 24: FasterRisk model for the mushroom dataset, predicting whether a mushroom is poisonous.

1.	prior_counts ≤ 7	-3 points		...
2.	prior_counts ≤ 2	-3 points	+	...
3.	prior_counts ≤ 0	-2 points	+	...
4.	age ≤ 52	2 points	+	...
5.	age ≤ 33	2 points	+	...
6.	age ≤ 23	2 points	+	...
7.	age ≤ 20	4 points	+	...
		SCORE	=	

SCORE	-8	-6	-4	-3	-2	-1	0	1	2	3	4	6	7	10
RISK	11.3%	18.7%	29.3%	35.7%	42.7%	50.0%	57.3%	64.3%	70.7%	76.4%	81.3%	88.7%	91.3%	96.2%

Table 25: FasterRisk model for the COMPAS dataset, predicting whether individuals are arrested within two years of release.

1.	MSinceMostRecentInqexcl7days ≤ -8	-4 points		...
2.	MSinceMostRecentInqexcl7days ≤ 0	2 points	+	...
3.	NetFractionRevolvingBurden ≤ 37	-2 points	+	...
4.	ExternalRiskEstimate ≤ 70	2 points	+	...
5.	ExternalRiskEstimate ≤ 78	2 points	+	...
6.	AverageMInFile ≤ 60	2 points	+	...
7.	PercentTradesNeverDelq ≤ 85	2 points	+	...
		SCORE	=	

SCORE	-4	-2	0	2	4	6	8	10
RISK	8.0%	14.9%	26.0%	41.4%	58.6%	74.0%	85.1%	92.0%

Table 26: FasterRisk model for the FICO dataset, predicting whether an individual will default on a loan. -8 means a missing value on the FICO dataset.

1.	Clump Thickness	×4 points		...
2.	Uniformity of Cell Shape	×3 points	+	...
3.	Marginal Adhesion	×3 points	+	...
4.	Bare Nuclei	×3 points	+	...
5.	Bland Chromatin	×3 points	+	...
6.	Normal Nucleoli	×2 points	+	...
7.	Mitoses	×4 points	+	...
		SCORE	=	

SCORE	$\leq$ 55	60	65	70	75	80	85	90	95	$\geq$ 100
RISK	$\leq$ 5.1%	9.3%	16.2%	26.6%	40.6	56.3%	70.8%	82.1%	89.6	$\geq$ 94.2%

Table 27: FasterRisk model for the breastcancer dataset, predicting whether there is breast cancer using a biopsy.

1.	WordFrequency_Remove	×4 points		...
2.	WordFrequency_Free	×2 points		...
3.	WordFrequency_Business	×1 points	+	...
4.	WordFrequency_0	×4 points	+	...
5.	WordFrequency_HP	×-2 points	+	...
6.	WordFrequency_George	×-2 points	+	...
7.	CharacterFrequency_$	×5 points	+	...
		SCORE	=	

SCORE	$\leq$ -4	-3	-2	-1	0
RISK	$\leq$0.4%	1.3%	3.7%	10.2%	25.2%

SCORE	1	2	3	4	$\geq$ 5
RISK	50.0%	74.8%	89.8%	96.3%	$\geq$ 98.7%

Table 28: FasterRisk model for the spambase dataset, predicting if an e-mail is spam.

1.	previous case ≤ 20	-5 points		...
2.	previous case ≤ 10 or previous case ≥ 21	-4 points	+	...
3.	# of previous penal cases ≤ 1	-1 points	+	...
4.	# of previous penal cases ≤ 3	-1 points	+	...
5.	# of previous penal cases ≤ 5	-1 points	+	...
6.	age in years ≤ 21.80	1 points	+	...
7.	age in years ≤ 38.05	1 points	+	...
		SCORE	=	

SCORE	$\leq$-10	-9	-8	-7	-6	-5
RISK	$\leq$ 33.1%	50.0%	66.9%	80.3%	89.2%	94.3%

SCORE	-4	-3	-2	-1	0	$\geq$ 1
RISK	97.1%	98.6%	99.3%	99.6%	99.8%	99.9%

Table 29: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years.

1.	ExternalRiskEstimate$\leq$63	1 points		...
2.	ExternalRiskEstimate$\leq$70	2 points	+	...
3.	ExternalRiskEstimate$\leq$79	2 points	+	...
4.	AverageMInFile$\leq$59	2 points	+	...
5.	NumSatisfactoryTrades$\leq$13	2 points	+	...
6.	PercentTradesNeverDelq$\leq$95	1 points	+	...
7.	PercentInstallTrades$\leq$46	-1 points	+	...
8.	MSinceMostRecentInqexcl7days$\leq$-8	-5 points	+	...
9.	MSinceMostRecentInqexcl7days$\leq$0	2 points	+	...
10.	NetFractionRevolvingBurden$\leq$37	-2 points	+	...
		SCORE	=	

SCORE	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	2.7%	3.7%	5.0%	6.8%	9.2%	12.4%	16.4%	21.3%	27.3%	34.2%	41.9%

SCORE	3	4	5	6	7	8	9	10	11	12
RISK	50.0%	58.1%	65.8%	72.7%	78.7%	83.6%	87.6%	90.8%	93.2%	95.0%

Table 30: FasterRisk model for the FICO dataset, predicting whether an individual will default on a loan. -8 means a missing value on the FICO dataset.

F.2 Risk Score Models from the Pool of Solutions

F.2.1 Examples from the Pool of Solutions (Bank Dataset)

The extra risk score examples from the pool of solutions are shown in Tables 31-42. All models were from the pool of the third fold on the bank dataset, and we show the top 12 models, provided in ascending order of the logistic loss on the training set (the model with the smallest logistic loss comes first).

1.	Call in Second Quarter	-2 points		...
2.	Previous Call Was Successful	4 points	+	...
3.	Previous Marketing Campaign Failed	-1 points	+	...
4.	Employment Indicator $>$ 5100	-5 points	+	...
5.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1
RISK	$\leq$ 11.2%	15.1%	20.1%	26.2%	33.4%
SCORE	0	1	2	3	4
RISK	41.5%	50.0%	58.5%	66.6%	73.8%

Table 31: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9352.39. The AUCs on the training and test sets are 0.779 and 0.770, respectively.

1.	Call in Second Quarter	-2 points		...
2.	Previous Call Was Successful	4 points	+	...
3.	Previous Marketing Campaign Failed	-1 points	+	...
4.	Employment Variation Rate $<$ -1	5 points	+	...
5.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
		SCORE	=	

SCORE	$\leq$0	1	2	3	4
RISK	$\leq$ 11.2%	15.1%	20.1%	26.2%	33.4%
SCORE	5	6	7	8	9
RISK	41.5%	50.0%	58.5%	66.6%	73.8%

Table 32: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9352.39. The AUCs on the training and test sets are 0.779 and 0.770, respectively.

1.	Call in Second Quarter	-2 points		...
2.	Previous Call Was Successful	4 points	+	...
3.	Previous Marketing Campaign Failed	-1 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-5 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1
RISK	$\leq$ 11.2%	15.2%	20.1%	26.3%	33.4%
SCORE	0	1	2	3	4
RISK	41.5%	50.0%	58.5%	66.6%	73.7%

Table 33: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9352.86. The AUCs on the training and test sets are 0.779 and 0.769, respectively.

1.	Call in Second Quarter	-2 points		...
2.	Previous Marketing Campaign Failed	-1 points	+	...
3.	Previous Marketing Campaign Succeeded	4 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-5 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1
RISK	$\leq$ 11.3%	15.3%	20.2%	26.3%	33.5%
SCORE	0	1	2	3	4
RISK	41.5%	50.0%	58.5%	66.5%	73.6%

Table 34: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9363.40. The AUCs on the training and test sets are 0.779 and 0.769, respectively. Note that some customers do not have previous marketing campaigns, so for these customers, neither of conditions 2 nor 3 are satisfied.

1.	Call in Second Quarter	-2 points		...
2.	Previous Call Was Successful	4 points	+	...
3.	Consumer Price Index > 93.5	1 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-1 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-5 points	+	...
		SCORE	=	

SCORE	$\leq$-4	-3	-2	-1	0
RISK	$\leq$ 9.6%	13.4%	18.4%	24.6%	32.2%
SCORE	1	2	3	4	5
RISK	40.8%	50.0%	59.2%	67.8%	75.4%

Table 35: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9365.51. The AUCs on the training and test sets are 0.778 and 0.769, respectively.

1.	Call in First Quarter	2 points		...
2.	Call in Second Quarter	-1 points		...
3.	Previous Call Was Successful	3 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-3 points	+	...
		SCORE	=	

SCORE	$\leq$-4	-3	-2	-1	0
RISK	$\leq$ 8.8%	13.4%	19.8%	28.2%	38.5%
SCORE	1	2	3	4	5
RISK	50.0%	61.5%	71.8%	80.2%	86.6%

Table 36: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9365.57. The AUCs on the training and test sets are 0.776 and 0.766, respectively.

1.	Call in Second Quarter	-2 points		...
2.	Previous Call Was Successful	3 points	+	...
3.	Previous Marketing Campaign Failed	-1 points	+	...
4.	Consumer Price Index $\geq$ 93.5	1 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-4 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1
RISK	$\leq$ 3.0%	5.2%	9.0%	15.0%	23.9%
SCORE	0	1	2	3	4
RISK	35.9%	50.0%	64.1%	76.1%	85.0%

Table 37: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9367.20. The AUCs on the training and test sets are 0.781 and 0.772, respectively.

1.	Call in Second Quarter	-1 points		...
2.	Previous Call Was Successful	5 points	+	...
3.	Calls Before Campaign Succeeded	-1 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-4 points	+	...
		SCORE	=	

SCORE	$\leq$-4	-3	-2	-1	0
RISK	$\leq$ 11.4%	16.3%	22.6%	30.6%	39.9%
SCORE	1	2	3	4	5
RISK	50.0%	60.1%	69.4%	77.4%	83.7%

Table 38: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9367.93. The AUCs on the training and test sets are 0.780 and 0.769, respectively.

1.	Call in Second Quarter	-1 points		...
2.	Any Prior Calls Before Campaign	4 points	+	...
3.	Previous Marketing Campaign Failed	-5 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-4 points	+	...
		SCORE	=	

SCORE	$\leq$-5	-4	-3	-2	-1
RISK	$\leq$ 7.8%	11.4%	16.2%	22.6%	30.5%
SCORE	0	1	2	3	4
RISK	39.9%	50.0%	60.1%	69.5%	77.4%

Table 39: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9371.75. The AUCs on the training and test sets are 0.779 and 0.769, respectively.

1.	Called via Landline Phone	-2 points		...
2.	Previous Call Was Successful	5 points	+	...
3.	Previous Marketing Campaign Failed	-2 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-4 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-4 points	+	...
		SCORE	=	

SCORE	≤-6	-5	-4	-3	-2
RISK	≤ 10.9%	14.2%	18.3%	23.2%	28.9%
SCORE	-1	0	1	3	5
RISK	35.4%	42.6%	50.0%	64.6%	76.8%

Table 40: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9376.52. The AUCs on the training and test sets are 0.776 and 0.765, respectively.

1.	Job Is Retired	1 points		...
2.	Call in Second Quarter	-2 points	+	...
3.	Previous Call Was Successful	5 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-5 points	+	...
		SCORE	=	

SCORE	≤-3	-2	-1	0	1
RISK	≤ 17.2%	22.2%	28.1%	34.8%	42.2%
SCORE	2	3	4	5	6
RISK	50.0%	57.8%	65.2%	72.0%	77.8%

Table 41: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9378.18. The AUCs on the training and test sets are 0.777 and 0.766, respectively.

1.	Age $\geq$ 60	1 points		...
2.	Call in Second Quarter	-2 points	+	...
3.	Previous Call Was Successful	5 points	+	...
4.	3 Month Euribor Rate $\geq$ 100	-2 points	+	...
5.	3 Month Euribor Rate $\geq$ 200	-5 points	+	...
		SCORE	=	

SCORE	≤-3	-2	-1	0	1
RISK	≤ 17.3%	22.2%	28.1%	34.8%	42.2%
SCORE	2	3	4	5	6
RISK	50.0%	57.8%	65.2%	71.9%	77.8%

Table 42: FasterRisk model for the bank dataset, predicting if a person opens a bank account after a marketing call. The logistic loss on the training set is 9378.68. The AUCs on the training and test sets are 0.777 and 0.767, respectively.

F.2.2 Examples from the Pool of Solutions (Mammo Dataset)

The extra risk score examples from the pool of solutions are shown in Tables 43-54. All models were from the pool of the third fold on the mammo dataset, and we show the top 12 models, provided in ascending order of the logistic loss on the training set (the model with the smallest logistic loss comes first).

1.	Oval Shape	-2 points	...
2.	Irregular Shape	4 points	+ ...
3.	Circumscribed Margin	-5 points	+ ...
4.	Spiculated Margin	2 points	+ ...
5.	Age ≥ 60	3 points	+ ...
		SCORE	=

SCORE	-7	-5	-4	-3	-2	-1
RISK	6.0%	10.6%	13.8%	17.9%	22.8%	28.6%
SCORE	0	1	2	3	4	≥ 5
RISK	35.2%	42.4%	50.0%	57.6%	64.8%	71.4%

Table 43: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 357.77. The AUCs on the training and test sets are 0.854 and 0.853, respectively.

1.	Lobular Shape	1 point	...
2.	Irregular Shape	3 points	+ ...
3.	Circumscribed Margin	-3 points	+ ...
4.	Spiculated Margin	1 point	+ ...
5.	Age ≥ 60	2 points	+ ...
		SCORE	=

SCORE	-3	-2	-1	0	1
RISK	7.5%	11.8%	18.1%	26.8%	37.7%
SCORE	2	3	4	5	6
RISK	50.0%	62.3%	73.2%	81.9%	88.2%

Table 44: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 357.86. The AUCs on the training and test sets are 0.854 and 0.857, respectively.

1.	Lobular Shape	2 points	...
2.	Irregular Shape	5 points	+ ...
3.	Circumscribed Margin	-4 points	+ ...
4.	Age < 30	-5 points	+ ...
5.	Age ≥ 60	3 points	+ ...
		SCORE	=

SCORE	-9	-7	-6	-5	-4	-3	-2	-1	0
RISK	1.3%	2.6%	3.7%	5.2%	7.3%	10.1%	14.0%	18.9%	25.1%
SCORE	1	2	3	4	5	6	7	8	10
RISK	32.6%	41.0%	50.0%	59.0%	67.4%	74.9%	81.1%	86.0%	92.7%

Table 45: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 358.24. The AUCs on the training and test sets are 0.852 and 0.854, respectively.

1.	Lobular Shape	2 points	...
2.	Irregular Shape	5 points	+ ...
3.	Circumscribed Margin	-4 points	+ ...
4.	Age ≥ 30	5 points	+ ...
5.	Age ≥ 60	3 points	+ ...
		SCORE	=

SCORE	-4	-2	-1	0	1	2	3	4	5
RISK	1.3%	2.6%	3.7%	5.2%	7.3%	10.1%	14.0%	18.9%	25.1%
SCORE	6	7	8	9	10	11	12	13	15
RISK	32.6%	41.0%	50.0%	59.0%	67.4%	74.9%	81.1%	86.0%	92.7%

Table 46: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 358.24. The AUCs on the training and test sets are 0.852 and 0.854, respectively.

1.	Irregular Shape	2 points	...
2.	Circumscribed Margin	-2 points	+ ...
3.	Spiculated Margin	1 point	+ ...
4.	Age ≥ 30	2 points	+ ...
5.	Age ≥ 60	1 point	+ ...
		SCORE	=

SCORE	-2	-1	0	1	2	3	4	5	6
RISK	2.3%	4.7%	9.5%	18.2%	32.0%	50.0%	68.0%	81.8%	90.5%

Table 47: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 358.59. The AUCs on the training and test sets are 0.852 and 0.857, respectively.

1.	Irregular Shape	2 points	...
2.	Circumscribed Margin	-2 points	+ ...
3.	Spiculated Margin	1 point	+ ...
4.	Age < 30	-2 points	+ ...
5.	Age ≥ 60	1 point	+ ...
		SCORE	=

SCORE	-4	-3	-2	-1	0	1	2	3	4
RISK	2.3%	4.7%	9.5%	18.2%	32.0%	50.0%	68.0%	81.8%	90.5%

Table 48: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 358.59. The AUCs on the training and test sets are 0.852 and 0.857, respectively.

1.	Lobular Shape	2 points		...
2.	Irregular Shape	5 points	+	...
3.	Circumscribed Margin	-4 points	+	...
4.	Obscure Margin	-1 point	+	...
5.	Age $\geq$ 60	3 points	+	...
		SCORE	=	

SCORE	-5	-4	-3	-2	-1	0	1	2
RISK	5.3%	7.4%	10.3%	14.1%	19.1%	25.3%	32.7%	41.1%
SCORE	3	4	5	6	7	8	9	10
RISK	50.0%	58.9%	67.3%	74.7%	80.9%	85.9%	89.7%	92.6%

Table 49: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 358.71. The AUCs on the training and test sets are 0.852 and 0.857, respectively.

1.	Irregular Shape	5 points		...
2.	Circumscribed Margin	-5 points	+	...
3.	Microlobulated Margin	2 points	+	...
4.	Spiculated Margin	2 points	+	...
5.	Age $\geq$ 60	3 points	+	...
		SCORE	=	

SCORE	-5	-3	-2	-1	0	2	3
RISK	8.6%	14.6%	18.6%	23.5%	29.2%	42.7%	50.0%
SCORE	4	5	7	8	9	10	12
RISK	57.3%	64.3%	76.5%	81.4%	85.4%	88.7%	93.4%

Table 50: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 358.98. The AUCs on the training and test sets are 0.852 and 0.852, respectively.

1.	Irregular Shape	4 points		...
2.	Circumscribed Margin	-5 points	+	...
3.	SpiculatedMargin	2 points	+	...
4.	Age $\geq$ 45	1 point	+	...
5.	Age $\geq$ 60	3 points	+	...
		SCORE	=	

SCORE	-5	-4	-3	-2	-1	0	1	2
RISK	7.3%	9.7%	12.9%	16.9%	21.9%	27.8%	34.6%	42.1%
SCORE	3	4	5	6	7	8	9	10
RISK	50.0%	57.9%	65.4%	72.2%	78.1%	83.1%	87.1%	90.3%

Table 51: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 359.10. The AUCs on the training and test sets are 0.855 and 0.859, respectively.

1.	Irregular Shape	4 points		...
2.	Circumscribed Margin	-5 points	+	...
3.	Obscure Margin	-1 points	+	...
4.	Spiculated Margin	2 points	+	...
5.	Age $\geq$ 60	3 points	+	...
		SCORE	=	

SCORE	-6	-5	-4	-3	-2	-1	0	1
RISK	6.8%	9.2%	12.3%	16.3%	21.3%	27.3%	34.2%	41.9%
SCORE	2	3	4	5	6	7	8	9
RISK	50.0%	58.1%	65.8%	72.7%	78.7%	83.7%	87.7%	90.8%

Table 52: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 359.34. The AUCs on the training and test sets are 0.852 and 0.862, respectively.

1.	Oval Shape	-1 point		...
2.	Lobular Shape	1 point	+	...
3.	Irregular Shape	4 points	+	...
4.	Circumscribed Margin	-4 points	+	...
5.	Age $\geq$ 60	3 points	+	...
		SCORE	=	

SCORE	-5	-4	-3	-2	-1	0	1
RISK	7.0%	9.8%	13.6%	18.5%	24.8%	32.3%	40.8%
SCORE	2	3	4	5	6	7	8
RISK	50.0%	59.2%	67.7%	75.2%	81.5%	86.4%	90.2%

Table 53: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 359.53. The AUCs on the training and test sets are 0.850 and 0.849, respectively.

1.	Lobular Shape	1 point		...
2.	Irregular Shape	4 points	+	...
3.	Circumscribed Margin	-3 points	+	...
4.	Age $\geq$ 45	1 point	+	...
5.	Age $\geq$ 60	2 points	+	...
		SCORE	=	

SCORE	-3	-2	-1	0	1	2
RISK	6.3%	9.5%	14.1%	20.5%	28.9%	38.9%
SCORE	3	4	5	6	7	8
RISK	50.0%	61.1%	71.1%	79.5%	85.9%	90.5%

Table 54: FasterRisk model for the mammo dataset, predicting the risk of malignancy of a breast lesion. The logistic loss on the training set is 359.53. The AUCs on the training and test sets are 0.852 and 0.850, respectively.

F.2.3 Examples from the Pool of Solutions (Netherlands Dataset)

The extra risk score examples from the pool of solutions are shown in Tables 55-66. All models were from the pool of the third fold on the Netherlands dataset, and we show the top 12 models, provided in ascending order of the logistic loss on the training set (the model with the smallest logistic loss comes first).

1.	# of previous penal cases ≤ 2	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.633	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4
RISK	14.9%	23.8%	35.8%	50.0%	64.2%	76.2%	85.1%

SCORE	-3	-2	-1	0	1	2
RISK	91.1%	94.8%	97.0%	98.3%	99.1%	99.5%

Table 55: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9226.84. The AUCs on the training and test sets are 0.743 and 0.742, respectively.

1.	# of previous penal cases ≤ 1	-1 point		...
2.	# of previous penal cases ≤ 3	-1 point	+	...
3.	age in years ≤ 38.052	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-3 points	+	...
		SCORE	=	

SCORE	-8	-7	-6	-5	-4
RISK	12.4%	27.4%	50.0%	72.6%	87.6%

SCORE	-3	-2	-1	0	1
RISK	94.9%	98.0%	99.2%	99.7%	99.9%

Table 56: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9232.51. The AUCs on the training and test sets are 0.744 and 0.739, respectively.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 23.265	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4
RISK	16.1%	24.9%	36.6%	50.0%	63.4%	75.1%	83.9%

SCORE	-3	-2	-1	0	1	2
RISK	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 57: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9250.94. The AUCs on the training and test sets are 0.739 and 0.739, respectively.

1.	# of previous penal cases ≤ 3	-2 points	...
2.	age in years ≤ 38.052	1 point	+ ...
3.	age at first penal case ≤ 22.989	1 point	+ ...
4.	previous case ≤ 10 or > 20	-3 points	+ ...
5.	previouse case ≤ 20	-5 points	+ ...
		SCORE	=

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.1%	25.0%	36.6%	50.0%	63.4%	75.0%	83.9%	90.0%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 58: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9250.95. The AUCs on the training and test sets are 0.738 and 0.739, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points	...
2.	age in years ≤ 38.052	1 point	+ ...
3.	age at first penal case ≤ 23.283	1 point	+ ...
4.	previous case ≤ 10 or > 20	-3 points	+ ...
5.	previouse case ≤ 20	-5 points	+ ...
		SCORE	=

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.0%	24.9%	36.6%	50.0%	63.4%	75.1%	84.0%	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 59: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9251.14. The AUCs on the training and test sets are 0.739 and 0.739, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points	...
2.	age in years ≤ 38.052	1 point	+ ...
3.	age at first penal case ≤ 22.934	1 point	+ ...
4.	previous case ≤ 10 or > 20	-3 points	+ ...
5.	previouse case ≤ 20	-5 points	+ ...
		SCORE	=

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.1%	25.0%	36.6%	50.0%	63.4%	75.0%	83.9%	90.0%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 60: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9251.39. The AUCs on the training and test sets are 0.739 and 0.740, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.907	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.1%	24.9%	36.6%	50.0%	63.4%	75.1%	83.9%	90.0%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 61: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9251.53. The AUCs on the training and test sets are 0.739 and 0.740, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 23.328	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.0%	24.9%	36.5%	50.0%	63.5%	75.1%	84.0%	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 62: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9252.07. The AUCs on the training and test sets are 0.738 and 0.739, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.965	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.1%	24.9%	36.6%	50.0%	63.4%	75.1%	83.9%	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 63: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9252.13. The AUCs on the training and test sets are 0.738 and 0.740, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.830	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.1%	24.9%	36.6%	50.0%	63.4%	75.1%	83.9%	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 64: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9252.19. The AUCs on the training and test sets are 0.739 and 0.740, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.870	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.1%	24.9%	36.6%	50.0%	63.4%	75.1%	83.9%	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 65: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9252.25. The AUCs on the training and test sets are 0.739 and 0.740, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

1.	# of previous penal cases ≤ 3	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 23.233	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	16.0%	24.9%	36.5%	50.0%	63.5%	75.1%	84.0%	90.1%	94.0%	96.5%	97.9%	98.8%	99.3%

Table 66: FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. The logistic loss on the training set is 9252.27. The AUCs on the training and test sets are 0.738 and 0.739, respectively. Note that this risk score is slightly different from that of Table 57 in Condition 3.

G Model Reduction

G.1 Reducing Models to Relatively Prime Coefficients

If the coefficients of a model are not relatively prime, one can divide all the coefficients by any common prime factors without changing any of the predicted risks. Table 67(left), copied from Table 6, is reduced in this way to produce Table 67(right). Table 68(left), copied from Table 7, is reduced in this way to produce Table 68(right).

1.	odor=almond	-5 points		...
2.	odor=anise	-5 points	+	...
3.	odor=none	-5 points	+	...
		SCORE	=	

SCORE	-5	0
RISK	10.8%	96.0%

1.	odor=almond	-1 points		...
2.	odor=anise	-1 points	+	...
3.	odor=none	-1 points	+	...
		SCORE	=	

SCORE	-1	0
RISK	10.8%	96.0%

Table 67: *Left:* FasterRisk model for the Mushroom dataset, predicting whether a mushroom is poisonous. Copy of Table 6. *Right:* Reduction to have relatively prime coefficients.

1.	prior_counts ≤ 2	-4 points		...
2.	prior_counts ≤ 7	-4 points	+	...
3.	age ≤ 31	4 points	+	...
		SCORE	=	

SCORE	-8	-4	0	4
RISK	23.6%	44.1%	67.0%	83.9%

1.	prior_counts ≤ 2	-1 points		...
2.	prior_counts ≤ 7	-1 points	+	...
3.	age ≤ 31	1 points	+	...
		SCORE	=	

SCORE	-2	-1	0	1
RISK	23.6%	44.1%	67.0%	83.9%

Table 68: *Left:* FasterRisk model for the COMPAS dataset, predicting whether individuals are arrested within two years of release. Copy of Table 7. *Right:* Reduction to have relatively prime coefficients.

G.2 Transforming Features for Better Interpretability

Sometimes the original features are not as interpretable as they could be with some minor postprocessing. For example, Table 69 has features "previous case $\leq$ 10 or > 20" and "previous case $\leq$ 20". We can transform them into more interpretable and user-friendly features as "previous case $\leq$ 10", "10 < previous case $\leq$ 20", and "previous case > 20". The transformed model is shown in Table 70.

1.	# of previous penal cases ≤ 2	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.633	1 point	+	...
4.	previous case ≤ 10 or > 20	-3 points	+	...
5.	previouse case ≤ 20	-5 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	14.9%	23.8%	35.8%	50.0%	64.2%	76.2%	85.1%	91.1%	94.8%	97.0%	98.3%	99.1%	99.5%

Table 69: Original FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years.

1.	# of previous penal cases ≤ 2	-2 points		...
2.	age in years ≤ 38.052	1 point	+	...
3.	age at first penal case ≤ 22.633	1 point	+	...
4.	previous case ≤ 10	-8 points	+	...
5.	$10 <$ previouse case ≤ 20	-5 points	+	...
6.	previouse case > 20	-3 points	+	...
		SCORE	=	

SCORE	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2
RISK	14.9%	23.8%	35.8%	50.0%	64.2%	76.2%	85.1%	91.1%	94.8%	97.0%	98.3%	99.1%	99.5%

Table 70: Postprocessed FasterRisk model for the Netherlands dataset, predicting whether defendants have any type of charge within four years. We have transformed the "previous case" feature for better interpretability. Note that in the original model, samples with previous case values less than 10 accumulate -8 points, -3 for the 4th line and -5 for the 5th line. In the transformed model, this case is more clearly stated in line 4.

H Discussion of Limitations

FasterRisk does not provide provably optimal solutions to an NP-hard problem, which is how it is able to perform in reasonable time. FasterRisk's models should not be interpreted as causal. FasterRisk creates very sparse, generalized, additive models, and thus has limited model capacity. FasterRisk's models inherit flaws from data on which it was trained. FasterRisk is not yet customized to a given application, which can be done in future work. We note that even if a model is interpretable, it can still have negative societal bias. (Generally, it is easier to check for such biases with scoring systems than with black box models). Looking at a variety of models from the diverse pool can help users to find models that are more fair.