A Summary of datasets

We show experimental results on four datasets: a recidivism dataset (COMPAS) [29], the Fair Isaac (FICO) credit risk dataset [30] used for the Explainable ML Challenge, the Diabetes dataset [31], and an ICU dataset MIMIC-II [32]. Table 1 summarizes all the datasets. We note that these are real-world datasets using features that are known to be used in systems of this type. Specifically, the COMPAS dataset is a criminal recidivism dataset and we use features that are based on criminal history, age, etc. For FICO, the data were provided by FICO itself based on how credit scores are usually constructed. MIMIC-II is based on features observed in ICU patients that are predictive of death, and the Diabetes dataset also uses typical measurements taken during pregnancy that are indicative of diabetes. The features for each dataset are generally available for the decision and they are interpretable.

Dataset	Samples	Features	Classification task
COMPAS	6907	7	Predict if someone will be arrested ≤ 2 years of release
FICO	10459	23	Predict if someone will default on a loan
Diabetes	768	9	Predict whether a pregnant woman has diabetes
MIMIC-II	24508	17	Predict whether a patient dies in the ICU

Table 1: Summary of datasets

B Sampling Uniformly from Ellipsoid

Algorithm 1 *SampleFromEllipsoid*$(\mathbf{Q}, \boldsymbol{\omega}_c)$

Input: parameters of the ellipsoid $\mathbf{Q} \in \mathbb{R}^{m\times m}$, $\boldsymbol{\omega}_c \in \mathbb{R}^m$
Output: a point inside the ellipsoid $\boldsymbol{\omega} \in \mathbb{R}^m$
1: $\mathbf{u} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ *// sample an m dimensional vector from standard multivariate Gaussian distribution*
2: $\mathbf{u} \leftarrow \mathbf{u}/\|\mathbf{u}\|_2$ *// normalize it to get a unit-vector*
3: $r \sim U(0, 1)$ *// get a sample from uniform distribution*
4: $r \leftarrow r^{1/m}$ *// rescale to get the radius of a sample in a unit sphere*
5: $\mathbf{y} \leftarrow r\mathbf{u}$ *// $\mathbf{y}$ is a random point inside a unit sphere*
6: $\boldsymbol{\Lambda}, \mathbf{V} = \text{Eig}(\mathbf{Q})$ *// Eigen-decomposition, diagonal of $\boldsymbol{\Lambda}$ are the eigenvalues, and columns of $\mathbf{V}$ are eigenvectors*
7: $\mathbf{x} \leftarrow \boldsymbol{\Lambda}^{-\frac{1}{2}}\mathbf{V}^T\mathbf{y}$ *// get a point in the ellipsoid $\mathbf{x}^T\mathbf{Q}\mathbf{x} \leq 1$*
8: $\boldsymbol{\omega} = \mathbf{x} + \boldsymbol{\omega}_c$ *// shift it so that the center is $\boldsymbol{\omega}_c$*
return $\boldsymbol{\omega}$

Algorithm 1 describes the algorithm to uniformly sample from the ellipsoid $\{\boldsymbol{\omega} \in \mathbb{R}^m : (\boldsymbol{\omega} - \boldsymbol{\omega}_c)^T\mathbf{Q}(\boldsymbol{\omega} - \boldsymbol{\omega}_c) \leq 1\}$. The algorithm first samples a random point inside a high dimensional unit sphere (line 1-5), and applies a linear transformation (calculated from $\mathbf{Q}$ and $\boldsymbol{\omega}_c$) to get the point in the target ellipsoid (line 6-8). The whole process can be decomposed into a purely stochastic part, i.e., sample in the unit sphere, and a deterministic part, which is differentiable. Using this sampling algorithm, we can get samples for objective (6) and use gradient-based methods to optimize $\mathbf{Q}$ and $\boldsymbol{\omega}_c$. In addition, the algorithm is also used to sample data for the problem in Section 4.3 and to calculate precisions in Section 5.1.

C Precision and volume of the approximated Rashomon set

To run our method, we set the learning rate to 0.0001 and run 1000 iterations. C is set to 500. For the logistic regression and EBM baselines, we sample 2000 coefficient vectors by fitting GAMs on the bootstrap sampled subsets of data. We run logistic regression with ℓ_2 penalty and EBM with no interaction terms. We then find the minimum volume ellipsoid that can cover most coefficient vectors. Given a set of coefficient vectors $\boldsymbol{\omega}_{\text{samples}}$, we solve the following problem:

$$\min_{\mathbf{Q},\boldsymbol{\omega}_c} -\det(\mathbf{Q})^{\frac{1}{2m}} + \xi \cdot \frac{1}{2000}\sum_{i=1}^{2000} \max(\|\mathbf{Q}^{1/2}(\boldsymbol{\omega}_{\text{sample}_i} - \boldsymbol{\omega}_c)\|^2 - 1, 0). \tag{13}$$

We solve this problem via gradient descent. ξ is set to 1000. The number of iterations and learning rate in GD are set to 1000 and 0.01, respectively. We initialize $\mathbf{Q}$ by the ZCA whitening matrix and $\boldsymbol{\omega}_c$ by the average of $\boldsymbol{\omega}_{\text{samples}}$.

After rescaling the Rashomon set approximated by baselines, we sample 10,000 points from our $\hat{R}$ and rescaled baseline Rashomon sets, calculate the loss, and get the precision. We include more figures in this appendix that are similar to Figure 1 to compare with baselines using different values of λ_s and θ.

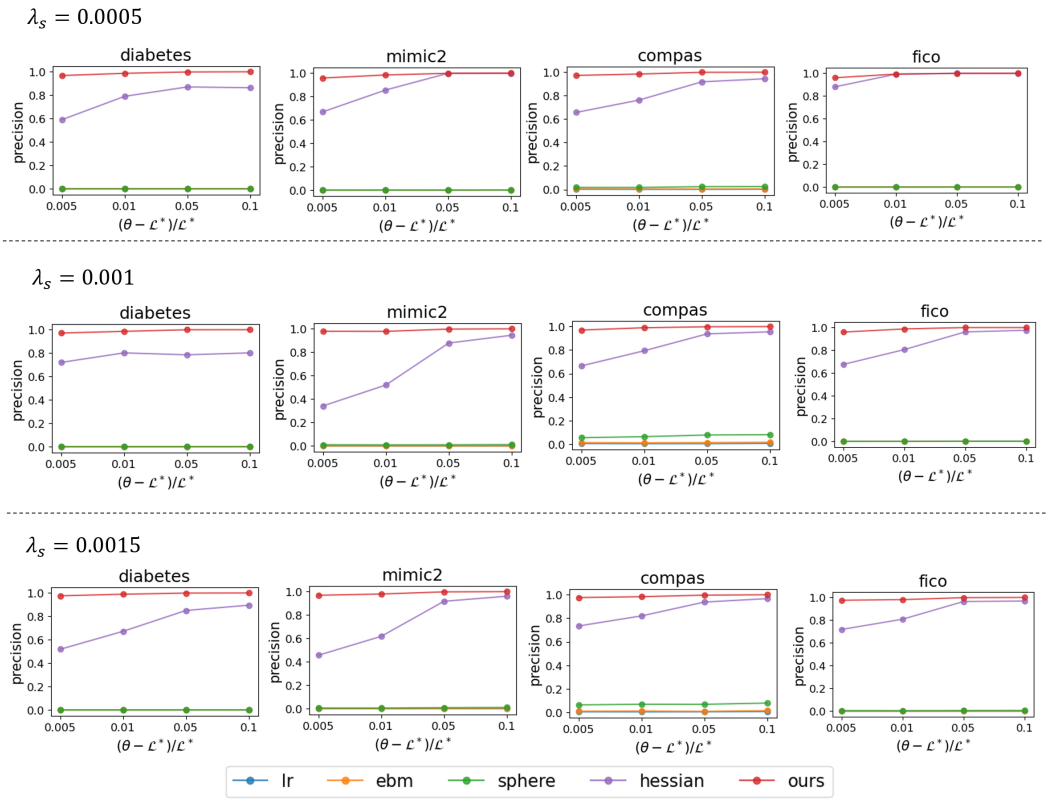


Figure 6: Precision of the approximated Rashomon sets as a function of θ. Our method always dominates other baselines. hessian is our starting point.

Figure 6 compares the precision of our method and baselines when the volume is fixed. The Rashomon set approximated by our method has the largest intersection with the true Rashomon set. This pattern is consistent across all datasets and values of λ_s. The Rashomon set approximated by the Hessian has lower precision but is always better than the other baselines. As θ becomes larger, the Hessian method becomes better and sometimes comes close to the result after optimization.

Figure 7 shows the tradeoff between the size and precision of the approximated Rashomon set for each method. The Rashomon set approximated by our optimization method is better than baselines given different values of θ.

We also report the optimization time of our method and baselines in Table 2. In most cases, our proposed method has a run time slightly longer than logistic regression with bootstrapping but shorter than the EBM baseline. Baselines "hessian" and "sphere" do not require the optimization step, so they finish instantaneously. For this table, we ran the gradient descent on a CPU, whereas had we used GPU, it would be at least 10x faster.

Though we show that our method can find the ellipsoid with high precision, one may still wonder how well the ellipsoid approximates the true Rashomon set using other metrics. To answer this question, we show the scaling ratio that is needed to ensure points sampled from the surface of the ellipsoid are outside the true Rashomon set in Table 3. On average, each dimension needs to scale by only a

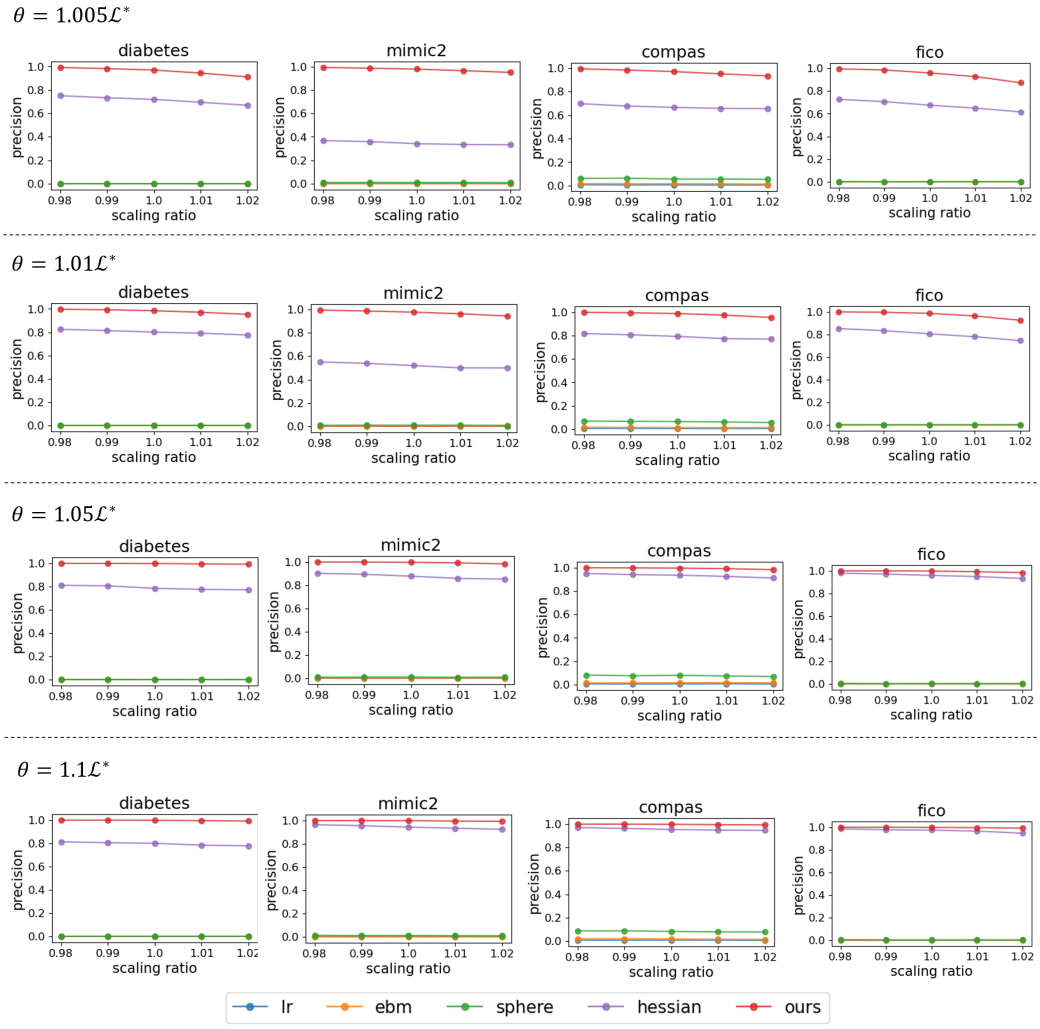


Figure 7: Tradeoff between precision and the size of the approximated Rashomon set given different θ. Note that the scaling factor to the power m is volume, which is proportional to recall. Our method dominates baselines on all datasets. As the scaling ratio increases, the precision starts to decrease.

small ratio to cover the true Rashomon set. For example, $\sim$5% is needed on FICO to cover the true Rashomon set. A small scaling ratio means our ellipsoid captures almost the whole Rashomon set, so that scaling by a small amount can cover the whole set. (Note that in practice we would not want to do this, it would be better to use our optimized approximation to the Rashomon set to avoid false positives inside the approximation.)

GAMs with different Support Sets: In these experiments, we keep 90%, 80%, and 70% of bins trained with $\lambda_s = 0.0005, \lambda_2 = 0.001$. For the baseline method, C is set to 3000, and the learning rate and the number of iterations are set to 0.001 and 2500, respectively. Since $\binom{K-p}{K-\tilde{K}}$ could be very large, we first sample 10,000 different merging strategies and compare at most 100 $R_{P_{\tilde{S}}}$. Table 4 shows more detailed results. Merging 30% of bins for the MIMIC-II dataset leads to empty $R_{P_{\tilde{S}}}$ for 10,000 merging strategies and merging bins for the Diabetes dataset also leads to empty $R_{P_{\tilde{S}}}$.

D Gradient-based optimization for log determinant

As we discussed in Section 3.1, we find the maximum volume inscribed ellipsoid by optimizing Eq (6). Eq (6) is not guaranteed to be convex, but using the log determinant, we can construct a

Dataset	λ_s	Ours	LR+sampling	EBM+sampling	Hessian (our initialization)	Sphere
Diabetes	5e-4	17.11	10.25	458.61	instant	instant
	1e-3	13.27	7.01	288.87	instant	instant
	1.5e-3	12.17	6.31	229.76	instant	instant
MIMIC-II	5e-4	437.87	786.85	3142.36	instant	instant
	1e-3	390.49	576.31	1862.03	instant	instant
	1.5e-3	383.91	572.8	1556.97	instant	instant
COMPAS	5e-4	94.92	22.65	246.14	instant	instant
	1e-3	84.61	18.92	155.77	instant	instant
	1.5e-3	81.05	18.38	170.49	instant	instant
FICO	5e-4	131.71	89.61	1169.73	instant	instant
	1e-3	119.08	73.02	975.37	instant	instant
	1.5e-3	117.14	68.28	901.0	instant	instant

Table 2: running time in seconds of our method compared to baselines using logistic regression and explainable boosting machine. Baselines "hessian" and "sphere" do not require the optimization step, so they finish instantaneously.

θ	scaling ratio (normalized)			
	Diabetes	MIMIC-II	COMPAS	FICO
$1.005\mathcal{L}^*$	1.081	1.145	1.096	1.014
$1.01\mathcal{L}^*$	1.079	1.111	1.087	1.020
$1.05\mathcal{L}^*$	1.108	1.076	1.089	1.041
$1.1\mathcal{L}^*$	1.119	1.078	1.095	1.048

Table 3: Scaling ratio needed to ensure points sampled from the surface of the ellipsoid are outside the true Rashomon set ($\lambda_s = 0.001, \lambda_2 = 0.001$).

convex objective function. In this appendix, we explore how performance changes if we optimize a convex objective.

Let us first define the new objective function:

$$\min_{Q,\boldsymbol{\omega}_c} -\frac{1}{2m}\log(\det(Q^{-1})) + C \cdot \mathbb{E}_{\boldsymbol{\omega}\sim\hat{R}(\mathbf{Q},\boldsymbol{\omega}_c)}[\max(\mathcal{L}(\boldsymbol{\omega}) - \theta, 0)]. \tag{14}$$

Similar to Eq (6), the first term is used to maximize the volume of the ellipsoid. The volume of an ellipsoid is proportional to $\det(Q^{-\frac{1}{2}})$. We normalize it by m, i.e. $(\det(Q^{-\frac{1}{2}}))^{\frac{1}{m}}$. Maximizing this term is equivalent to minimizing $(\det(Q^{-\frac{1}{2}}))^{-\frac{1}{m}}$.

Q is positive definite since it is the quadratic form for an ellipsoid. Then $Q^{\frac{1}{2}}$ is also positive definite and $\det(Q^{-1}) = (\det(Q))^{-1}$. We also know that $\det(Q^{\frac{1}{2}}) = (\det(Q))^{\frac{1}{2}}$. Therefore,

$$\begin{aligned}(\det(Q^{-\frac{1}{2}}))^{-\frac{1}{m}} &= (\det(Q^{\frac{1}{2}}))^{\frac{1}{m}} \\ &= (\det(Q))^{\frac{1}{2m}} \\ &= (\det(Q^{-1}))^{-\frac{1}{2m}}.\end{aligned}$$

We can take the log on the right-hand side term, and the objective is to minimize is thus $-\frac{1}{2m}\log(\det(Q^{-1}))$. It is well known that the log determinant is concave. After multiplying by $-\frac{1}{2m}$, this term is convex.

The second term penalizes the points sampled from the ellipsoid if they are outside $R(\theta)$. $\mathcal{L}(\boldsymbol{\omega})$ is convex w.r.t $\boldsymbol{\omega}$. Then $\mathcal{L}(\boldsymbol{\omega}) - \theta$ is also convex. Finding the maximum between a convex function and a constant is convex and the expectation is known to be convex, so the second term is also convex. Therefore, the objective function is convex and we then use gradient descent to find the minimizer.

The results obtained by minimizing Eq (14) are shown in Figure 8. They are almost the same as the results shown in Figures 6 and 7. And Table 5 shows similar results compared to Table 4. These

Dataset	$\tilde{K}$	precision ratio	volume ratio	time Method 2 (sec)	time Method 1 (sec)
COMPAS	15	1.01 ± 0.03	1.05 ± 0.07	4.84e-4 $\pm$ 6.25e-5	324.85 ± 8.70
COMPAS	14	1.01 ± 0.05	1.12 ± 0.12	4.76e-4 $\pm$ 9.33e-5	296.51 ± 41.46
COMPAS	13	0.99 ± 0.01	1.20 ± 0.12	4.77e-4 $\pm$ 1.13e-4	272.64 ± 6.64
FICO	28	1.01 ± 0.04	0.90 ± 0.06	9.99e-4 $\pm$ 4.54e-3	377.10 ± 14.95
FICO	26	0.99 ± 0.06	1.10 ± 0.13	5.59e-4 $\pm$ 1.12e-4	374.28 ± 10.24
FICO	24	0.99 ± 0.02	1.29 ± 0.13	5.55e-4 $\pm$ 4.04e-5	297.15 ± 8.26
MIMIC-II	35	1.00 ± 0.20	0.93 ± 0.10	5.89e-4 $\pm$ 1.72e-4	1127.10 ± 141.42
MIMIC-II	32	0.99 ± 0.01	1.10 ± 0.06	5.29e-4 $\pm$ 2.76e-5	1067.50 ± 12.90

Table 4: Precision, volume, and time comparison between blocking method (Method 2) and optimization (Method 1). This table shows that Method 2 is faster for the same task while performing equally well.

results indicate that optimizing the convex function doesn't bring us better results and during the experiments, we find hyperparameter tuning is even harder. Therefore, we use the previous results (optimizing the determinant, not the log determinant) in the remaining experiments.

Dataset	$\tilde{K}$	precision ratio	volume ratio	time method 2 (sec)	time method 1 (sec)
COMPAS	15	1.00 ± 0.02	1.09 ± 0.06	4.33e-4 $\pm$ 7.14e-5	158.12 ± 4.16
COMPAS	14	1.00 ± 0.03	1.12 ± 0.08	3.23e-4 $\pm$ 6.79e-5	136.90 ± 3.73
COMPAS	13	0.99 ± 0.01	1.20 ± 0.12	5.33e-4 $\pm$ 1.28e-4	324.19 ± 8.71
FICO	28	1.00 ± 0.01	0.95 ± 0.04	5.04e-4 $\pm$ 8.99e-4	467.49 ± 5.02
FICO	26	1.00 ± 0.04	1.09 ± 0.08	3.44e-4 $\pm$ 1.04e-4	465.45 ± 5.05
FICO	24	0.99 ± 0.01	1.25 ± 0.10	3.02e-4 $\pm$ 2.67e-5	449.33 ± 15.93
MIMIC-II	35	0.99 ± 0.01	0.98 ± 0.06	6.42e-4 $\pm$ 1.91e-4	1683.11 ± 17.64
MIMIC-II	32	0.99 ± 0.02	1.05 ± 0.03	5.77e-4 $\pm$ 4.21e-5	1666.22 ± 6.57

Table 5: Precision, volume, and time comparison between blocking method (Method 2) and optimization (Method 1) trained by optimizing Eq (14). The results are almost the same as those shown in Table 4.

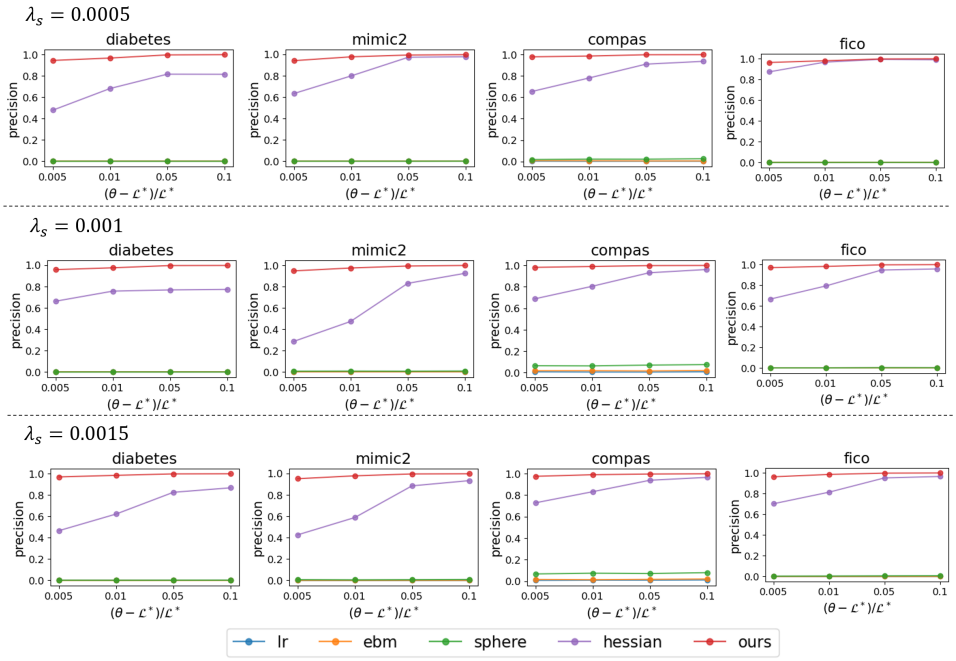
E Variable importance range

We first show more results on variable importance range and then compare the time taken to compute VI_+ for MIP-based formulation and LP formulation.

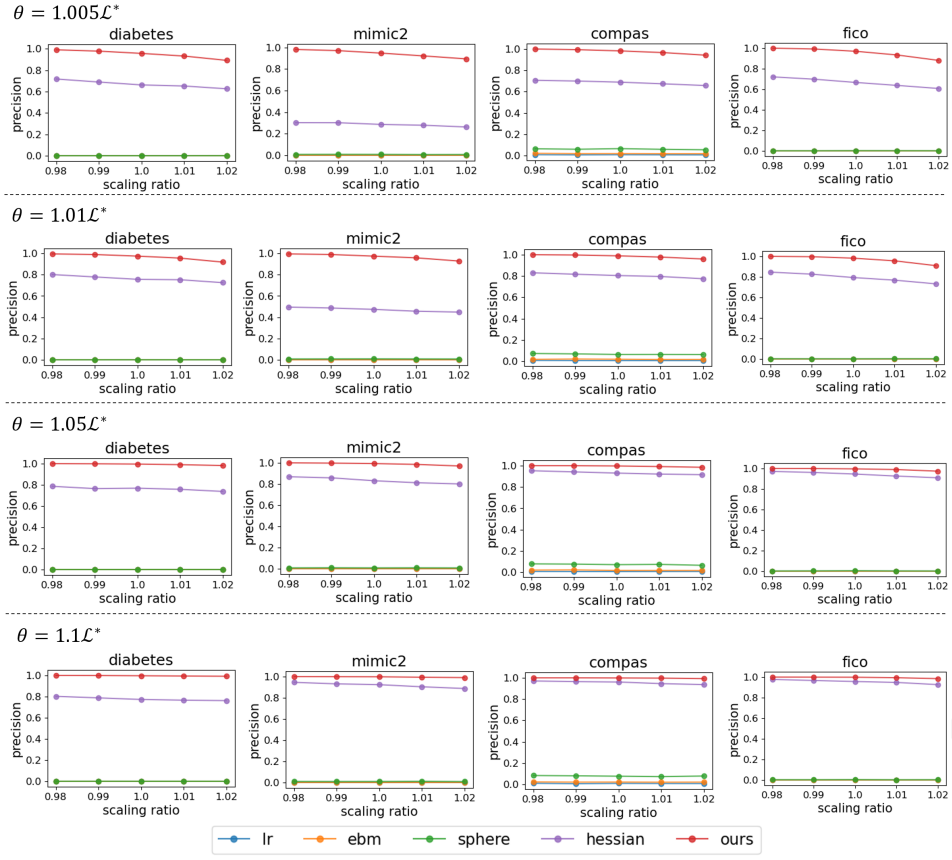
We show the shape functions of "Glucose" when the lowest and highest variable importance are achieved in Figure 3. When the importance of "Glucose" is minimized or maximized, one might be interested in how the shape function changes for other features. We show such variations in Figure 9. Most features keep the trend as $\boldsymbol{\omega}_c$ with some variation in magnitude.

Figure 10 shows the variable importance range on the MIMIC-II, FICO, and COMPAS datasets. For the MIMIC-II dataset (left subfigure), features "PFratio", "GCS", and "Age" have relatively higher VI_-, which means these features are, in general, important for GAMs in the Rashomon set. For the FICO dataset (mid subfigure), features "ExternalRiskEstimate" and "MSinceMostRecentInqexcl7days" have higher VI_- either fixing or not fixing other coefficients, indicating that these two features are important. Feature "prior_count" in the COMPAS dataset has slightly higher VI_- than feature "age."

As we mentioned in Section 4.1, the lower bound of variable importance VI_- is obtained by solving a linear programming problem with a quadratic constraint, while to get the upper bound of variable importance we need to solve a mixed integer programming problem. We use Python CVXPY package [35, 36] for solving LP problems and Cplex 12.8 for MIP problems. Note that as long as $M \geq 2|\boldsymbol{\omega}_{j,k}|$, the MIP formulation is valid. We set M to 200. This is usually large enough to bound the absolute value of each coefficient given that we have the ℓ_2 penalty. Since solving a MIP problem is usually time-consuming, we also propose another way to get VI_+. Note that it's the absolute value in the objective that restricts us to use LP solver. Therefore, we enumerate all positive-negative



(a) Precision of the approximated Rashomon set as a function of θ.



(b) Tradeoff between precision and the size of the approximated Rashomon set given different θ. The scaling factor to the power m is volume, which is proportional to recall.

Figure 8: Precision and volume of the approximated Rashomon set obtained by optimizing Eq (14). They are almost the same as the results shown in Figure 6 and 7.

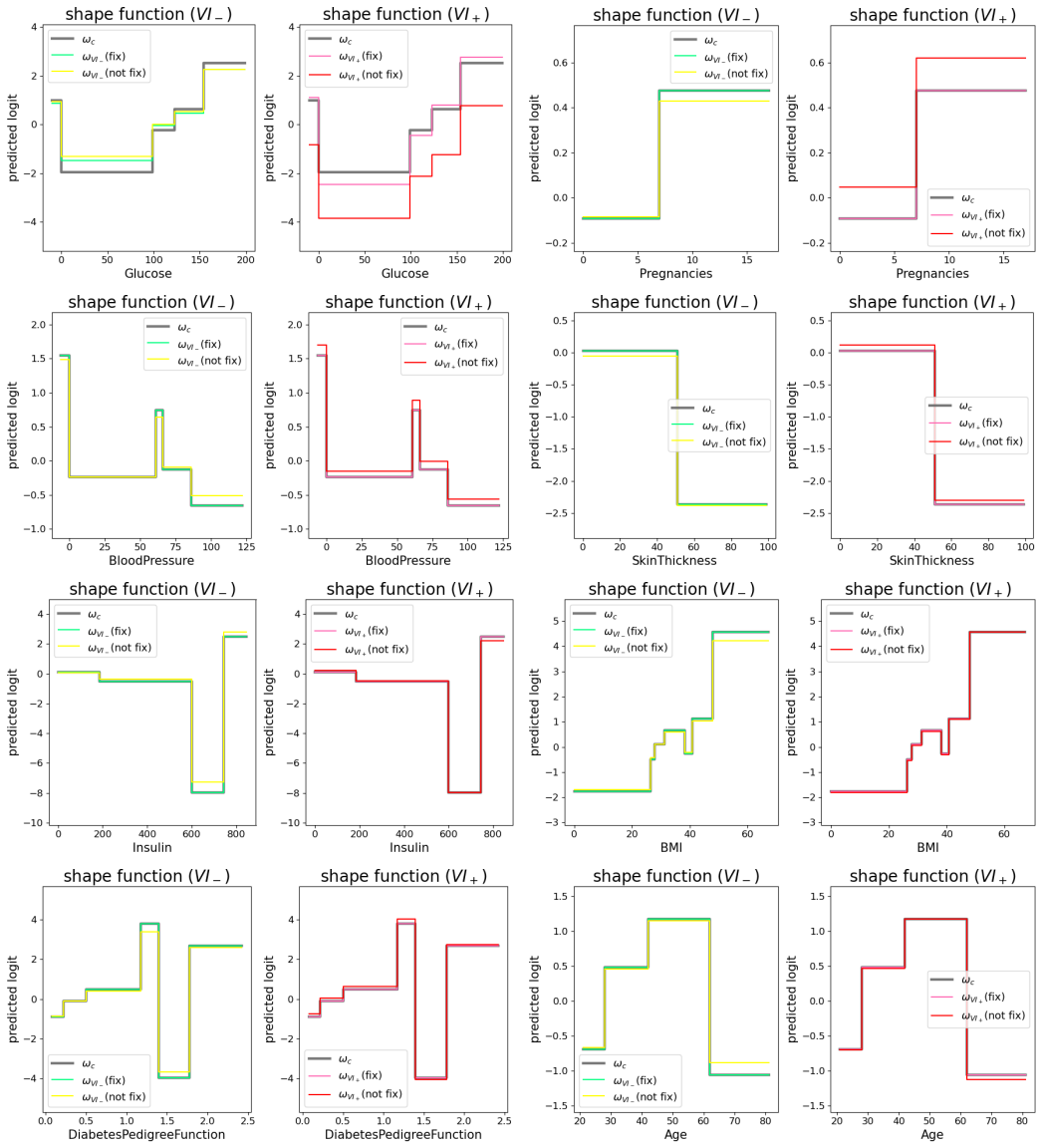


Figure 9: Shape functions of other features in the Diabetes dataset when the importance of "Glucose" is minimized or maximized.

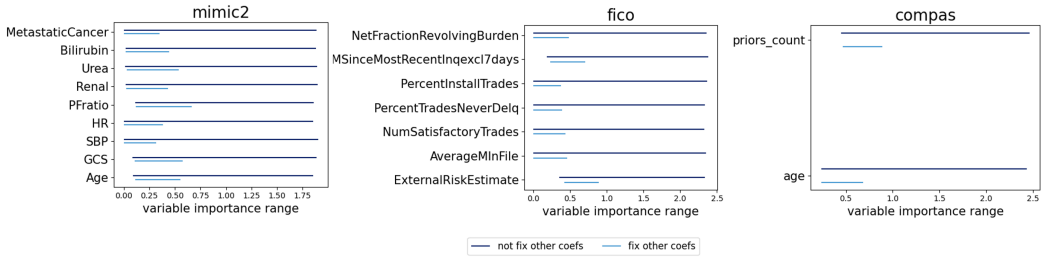


Figure 10: The variable importance range of the MIMIC-II, FICO, and COMPAS datasets. Each line connects VI_- and VI_+. ($\lambda_s = 0.001, \lambda_2 = 0.001, \theta = 1.01\mathcal{L}^*$)

Algorithm 2 *GetVI$_+$FromLP*$(\boldsymbol{\pi}_j, \mathbf{Q}, \boldsymbol{\omega}_c)$

Input: proportion of samples in each bin of feature j $\boldsymbol{\pi}_j \in \mathbb{R}^{B_j}$, parameters of the ellipsoid $\mathbf{Q} \in \mathbb{R}^{m \times m}$,$\boldsymbol{\omega}_c \in \mathbb{R}^m$
Output: a point inside the ellipsoid $\boldsymbol{\omega} \in \mathbb{R}^m$
$Obj^* \leftarrow -\infty$
for $I \in \{-1, 1\}^{B_j}$ *// try all positive-negative combinations for* $\omega_{j,k}, k \in \{0, ...B_j - 1\}$
  // solve the LP problem
  $Obj, \boldsymbol{\omega} \leftarrow \max_{\boldsymbol{\omega}_j} (\boldsymbol{\pi}_j \odot I)^T \boldsymbol{\omega}_j$ such that $(\boldsymbol{\omega} - \boldsymbol{\omega}_c)^T Q (\boldsymbol{\omega} - \boldsymbol{\omega}_c) \leq 1$
  if $Obj > Obj^*$:
    $\boldsymbol{\omega}^* \leftarrow \boldsymbol{\omega}, Obj^* \leftarrow Obj$
return $\boldsymbol{\omega}^*$

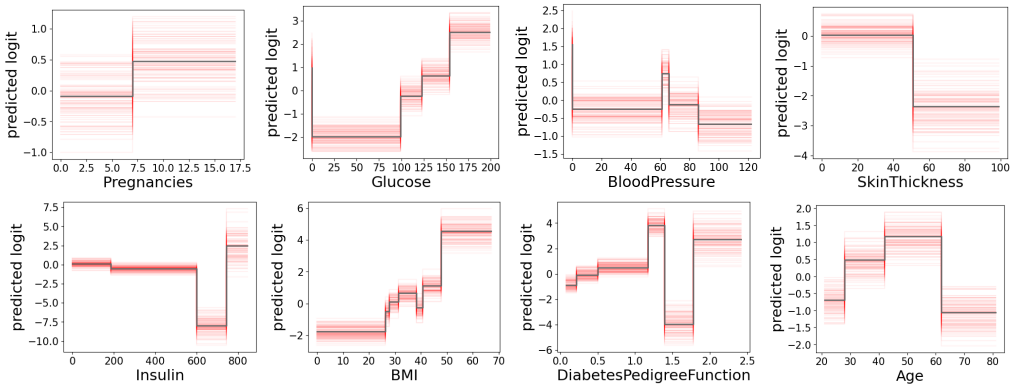


Figure 11: 100 different shape functions of the Diabetes dataset. The shape function at $\boldsymbol{\omega}_c$ is colored in gray. ($\lambda_s = 0.001, \lambda_2 = 0.001, \theta = 1.01\mathcal{L}^*$)

combinations of $\omega_{j,k}, k \in \{0, B_j - 1\}$, and then solve LP problem with the sign constraint enforced (see Algorithm 2). Table 6 compares the running time by solving both MIP and LP problems for VI_+. The runtime required to solve LP problems with the sign constraint enforced is usually less than that required to solve a MIP problem.

Dataset	Fix other coefs		Not fix other coefs	
	MIP	LP	MIP	LP
Diabetes	18.876 ± 10.025	4.780 ± 4.996	35.597 ± 17.413	1.961 ± 2.164
MIMIC-II	11.450 ± 6.259	0.497 ± 0.166	14.977 ± 12.921	0.265 ± 0.139
COMPAS	23.193 ± 0.176	4.138 ± 0.033	46.782 ± 5.709	3.043 ± 0.017
FICO	11.749 ± 7.841	0.816 ± 0.766	30.133 ± 7.11	0.425 ± 0.457

Table 6: Time comparison in seconds between solving MIP and LP problem with the sign constraint enforced for VI_+.

F Shape functions of GAMs in the Rashomon set

Next, we show a diverse set of coefficient vectors sampled from the approximated Rashomon set. Figure 11 and Figure 12 depict 100 coefficient vectors (in red) sampled from the ellipsoid and $\boldsymbol{\omega}_c$ (in gray), the center of the optimized ellipsoid of the Diabetes and MIMIC-II dataset, respectively. Various different red lines in each subfigure indicate that the approximated Rashomon set contains many different coefficient vectors. And these models are actually within the true Rashomon sets. Many of them may generally follow similar patterns as we can see from the trend of these shape functions, while some of them may have some variations (see Figure 5). In summary, using the approximated Rashomon set, we can easily get a diverse set of shape functions for each feature.

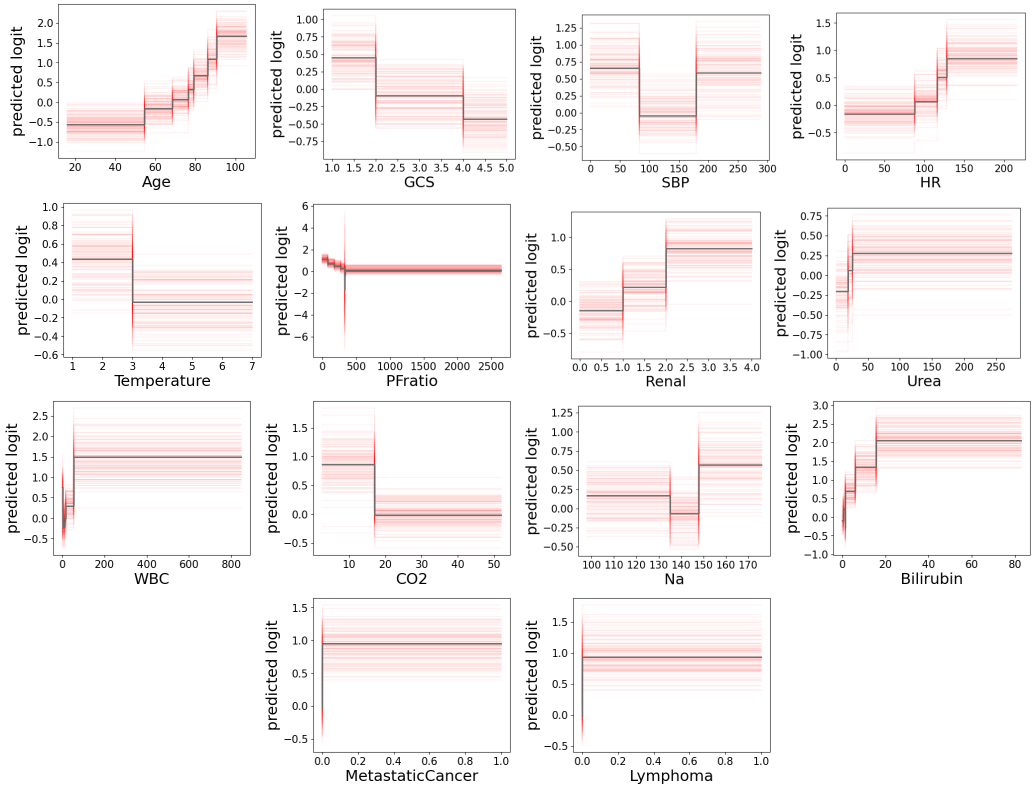


Figure 12: 100 different shape functions of the MIMIC-II dataset. The shape function at $\boldsymbol{\omega}_c$ is colored in gray. ($\lambda_s = 0.0002, \lambda_2 = 0.001, \theta = 1.01\mathcal{L}^*$)

G User preferred shape functions

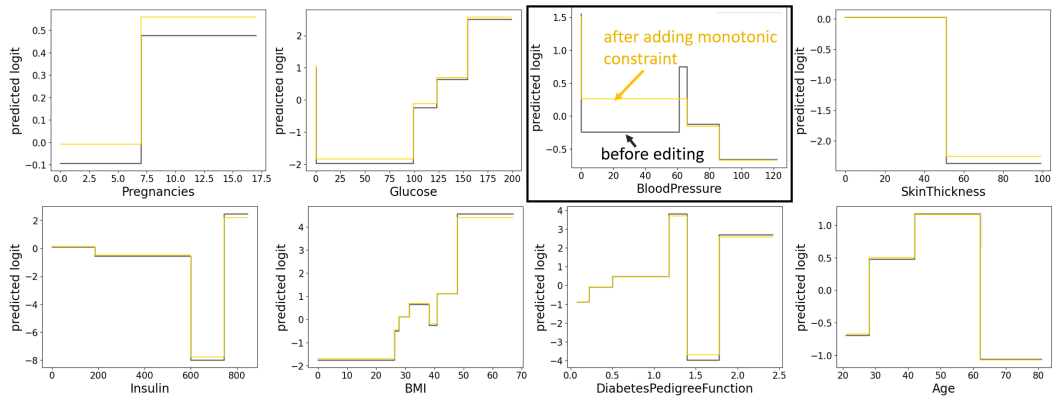
In real applications, users might have preference for shape functions that are consistent with their domain knowledge. Our approximated Rashomon set makes it easy to incorporate such user preferences by finding a model within the set that satisfies some constraints. We show two case studies here.

Diabetes: Figure 11 shows that a jump occurs when blood pressure is around 60. One possible user request might be making this blood pressure shape function monotonically decreasing. By solving the quadratic programming problem with linear constraints, we can get the shape functions colored in yellow in Figure 13a. The updated shape function for "BloodPressure" is monotonically decreasing. And the shape functions for other features are also updated. Almost all of them follow the trend in $\boldsymbol{\omega}_c$ (in gray), the center of the optimized ellipsoid, with small changes in magnitude. Note that this optimization problem is solved in 0.04 seconds.

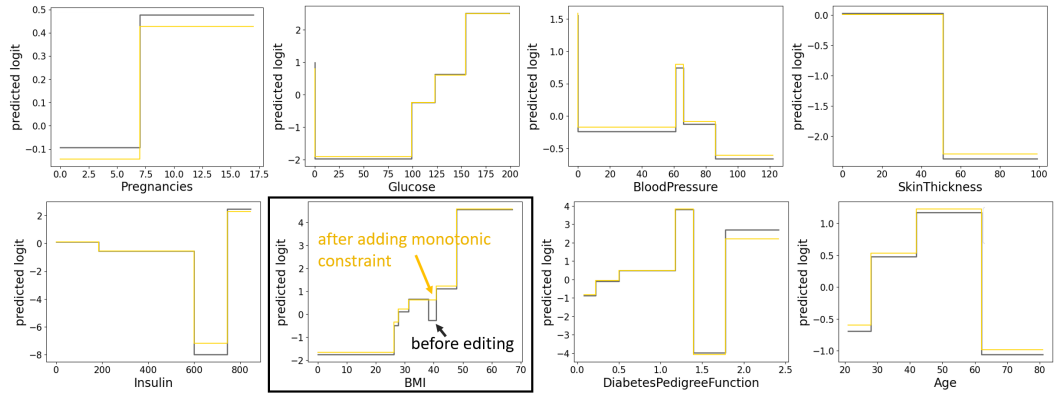
One might also be interested in making the shape function of "BMI" monotonically increasing. By solving the optimization problem, we can get the shape functions shown in Figure 13b. The updated shape function for "BMI" is monotonically increasing (in yellow). The sudden decrease that occurs at "BMI"=40 is removed by connecting the left step. Similar to Figure 13a, the updated shape functions of other features follow the trend in $\boldsymbol{\omega}_c$ (in gray), the center of the optimized ellipsoid, with small changes in magnitude. And this optimization problem is solved in 0.0004 seconds.

Sometimes a monotonic constraint is not what users want; they might have more specific preferences on the shape functions. Here we show some examples using hypothetical shape functions. Figure 14 extends the visualizations in Figure 4. It shows shape functions after imposing two different requests on "BloodPressure".

Figure 15 shows shape functions after imposing two different requests on "BMI." In Figure 15a, the requested shape function of "BMI" (colored in red in the top-left subfigure) shifts below the original shape function but maintains a monotonically increasing pattern. While the closest shape function



(a) Shape functions of the Diabetes dataset with the monotonic constraint on the "BloodPressure" (in yellow). The optimization problem is solved in 0.04 seconds.



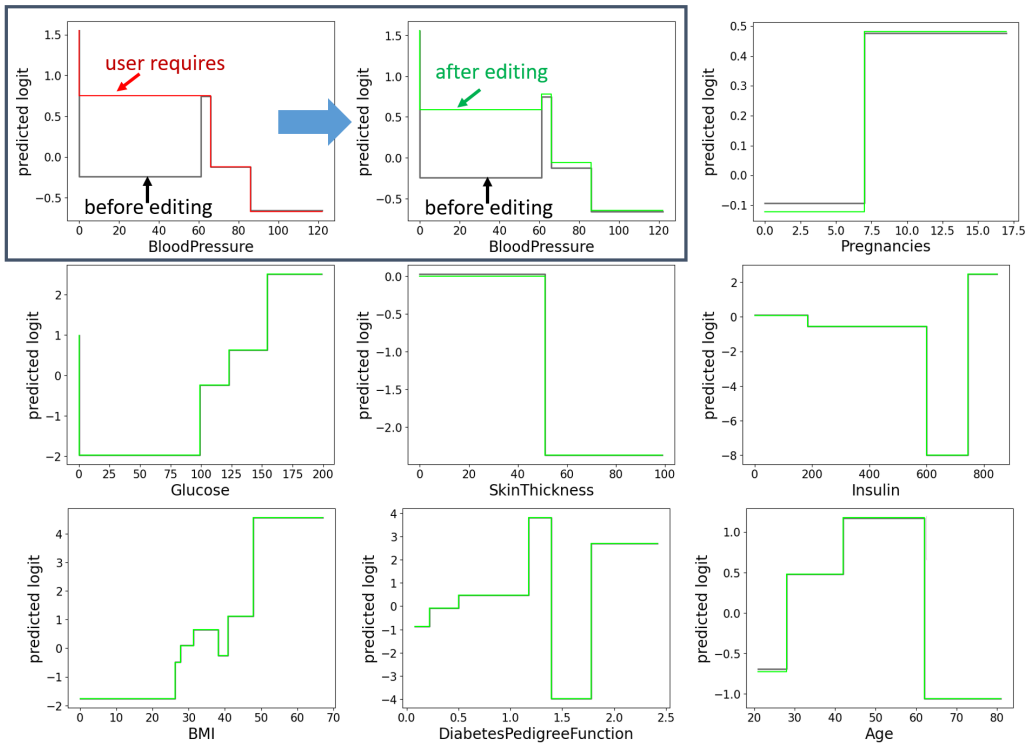
(b) Shape functions of the Diabetes dataset with the monotonic constraint on the "BMI" (in yellow). The optimization problem is solved in 0.0004 seconds.

Figure 13: Shape functions with monotonic constraints.

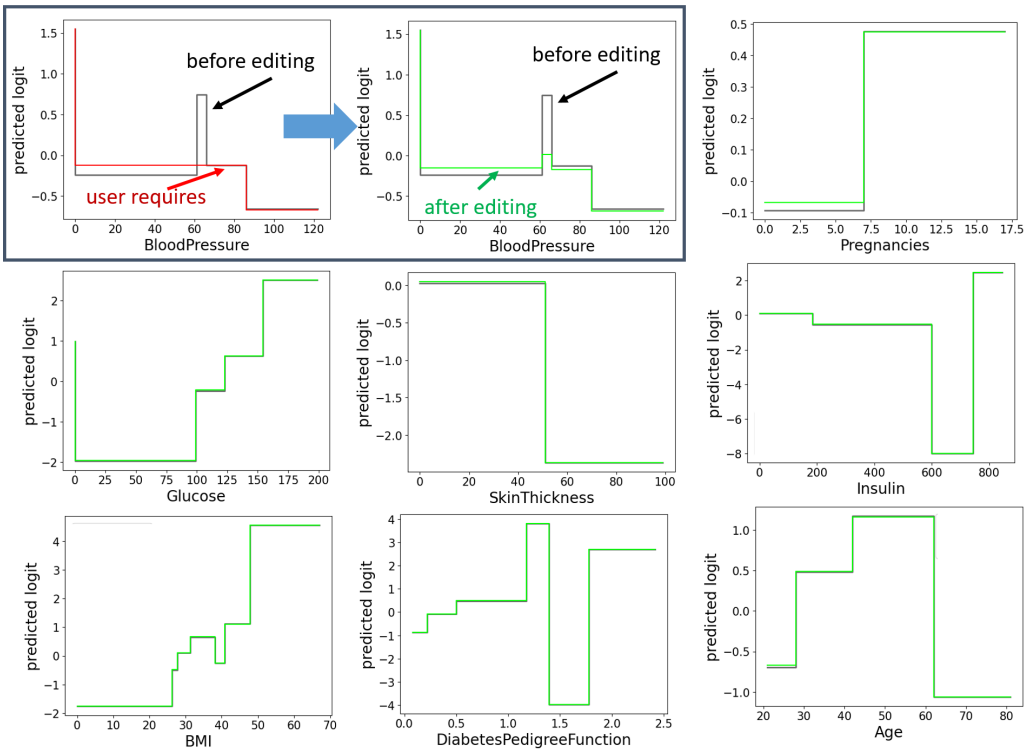
in the approximated Rashomon set is also below the original shape function, it is not necessarily monotonically increasing. Instead, it is more likely to follow the trend observed in the original shape function, as we aim to minimize the Euclidean distance between the requested shape function and the shape function in $\hat{R}$. Shape functions of other features change only slightly in magnitude. Figure 15b shows a different case. The requested shape function of "BMI" forces certain steps to have the same coefficient. However, after solving the QP problem, the updated shape function shown in green in the top-middle subfigure is a combination of theshape function before editing and the requested shape function.

MIMIC-II: In Figure 12, we can see jumps in several shape functions. For example, "PFratio" has a sudden jump around 330, and "Bilirubin" has a jump close to 0. PFratio is a measurement of lung function; it measures how well patients convert oxygen from the air into their blood. And bilirubin is a red-orange compound that breaks down heme. The bilirubin level reflects the balance between production and excretion. The elevated levels may indicate certain diseases. Missing values commonly exist in the real dataset and imputation is widely used. [28] shows these jumps are caused by mean imputation, and have no physical meaning. We can impose monotonic constraints on these two features simultaneously. We want the shape function of "PFratio" to be monotonically decreasing, while the shape function of "Bilirubin" is monotonically increasing. Figure 16 shows the shape functions after optimization. The shape functions of "PFratio" and "Bilirubin" satisfy the request as shown in the inset plots, and the shape functions of other features are only slightly changed.

Suppose a user prefers to remove the jump in the shape function of "PFratio." One way is to remove the jump while keeping the last step unchanged (top-left subfigure in Figure 17a). Fortunately, by solving Problem (12), we find that the specified shape function is within the Rashomon set (shown by

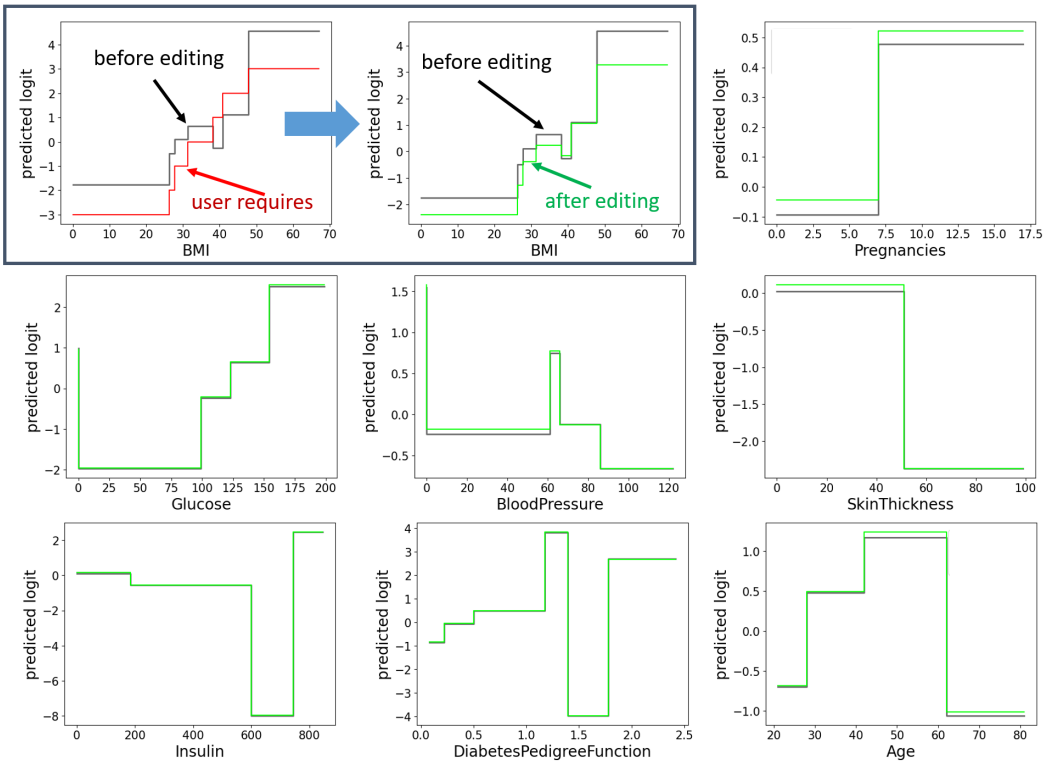


(a) Case 1

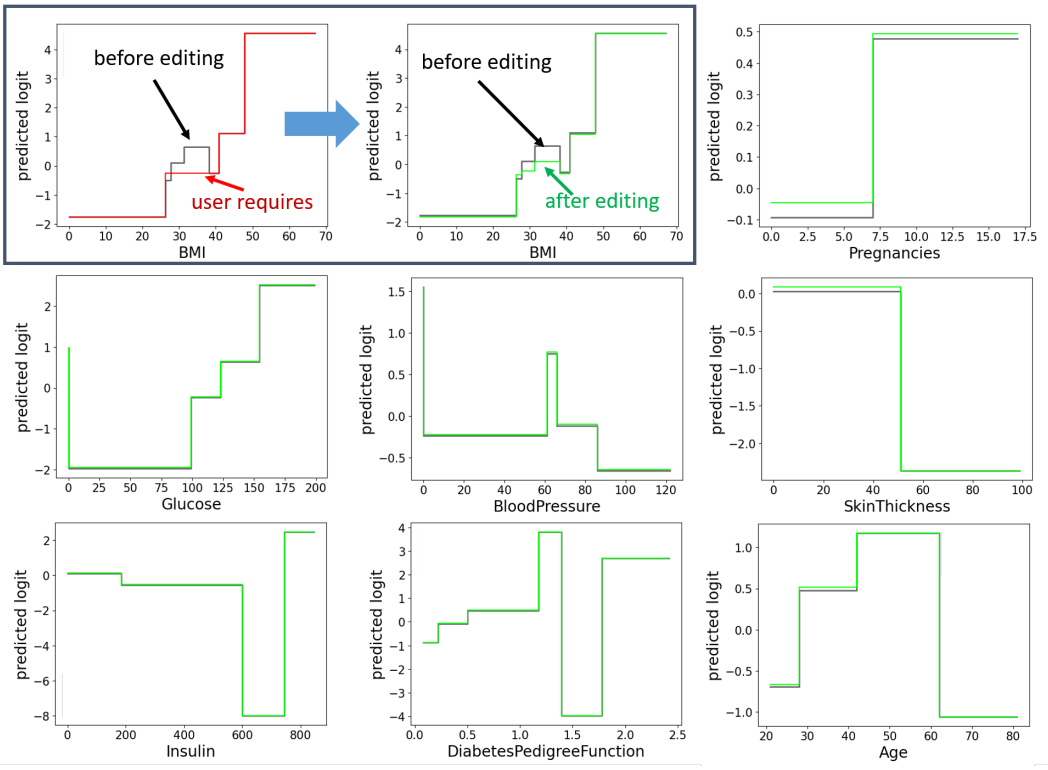


(b) Case 2

Figure 14: Shape functions on the Diabetes dataset after a hypothetical shape function on "Blood-Pressure" is requested. The red curve in the top-left subfigure is the required shape function. The shape function colored in green in the top-middle subfigure is the closest shape function within $\hat{R}$.



(a) Case 1



(b) Case 2

Figure 15: Shape functions on the Diabetes dataset after a hypothetical shape function on "BMI" is requested. The red curve in the top-left subfigure is the required shape function. The shape function colored in green in the top-middle subfigure is the closest shape function within $\hat{R}$.

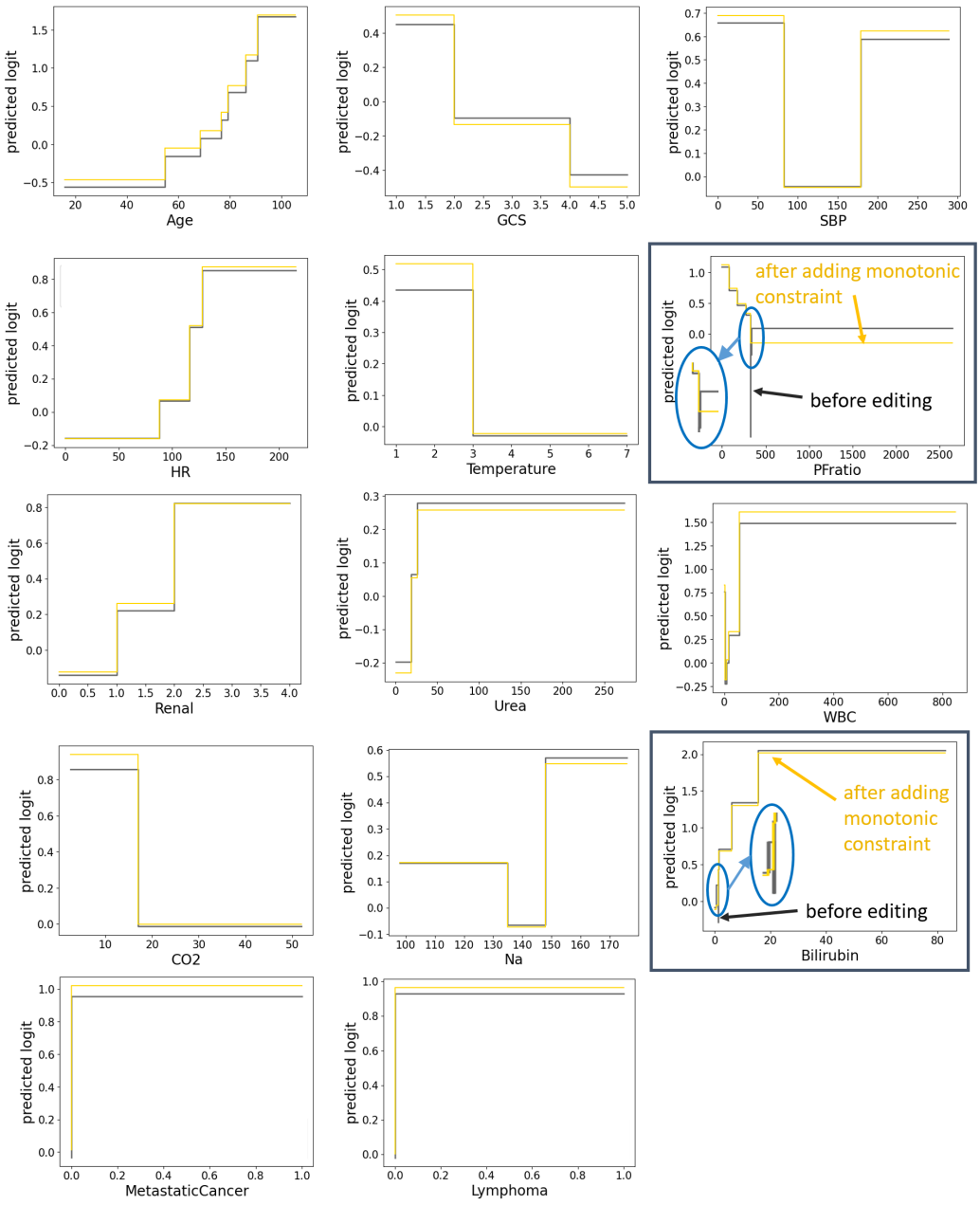


Figure 16: Shape functions of the MIMIC-II dataset with the monotonic constraints on the "PFratio" and "Bilirubin" (in yellow).

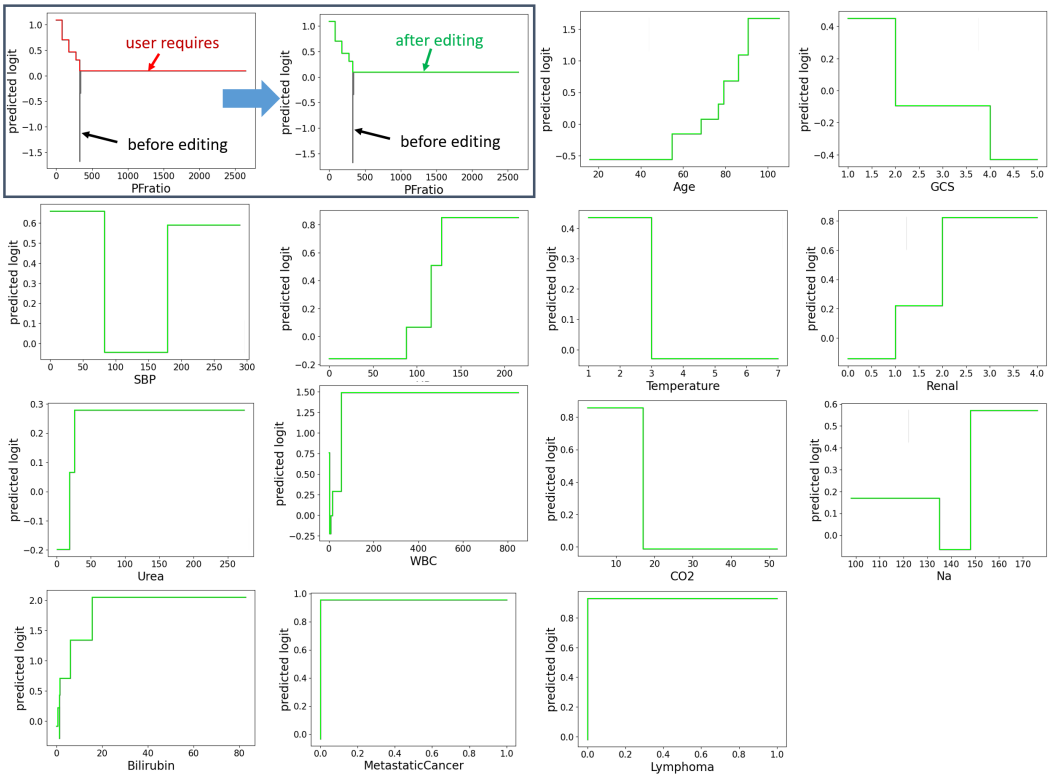
Dataset	θ (constant * $\mathcal{L}^*$)	$\boldsymbol{\omega}^*$		$\boldsymbol{\omega}$ sampled from $\hat{R}$	
		accuracy	auc	accuracy	auc
COMPAS	1.005	0.696	0.748	0.683 ± 0.010	0.744 ± 0.004
	1.01			0.683 ± 0.010	0.742 ± 0.005
	1.05			0.668 ± 0.017	0.724 ± 0.012
	1.1			0.649 ± 0.026	0.704 ± 0.021
FICO	1.005	0.720	0.792	0.717 ± 0.003	0.791 ± 0.001
	1.01			0.716 ± 0.004	0.790 ± 0.002
	1.05			0.708 ± 0.008	0.780 ± 0.006
	1.1			0.700 ± 0.010	0.770 ± 0.009
Diabetes	1.005	0.760	0.819	0.761 ± 0.004	0.818 ± 0.002
	1.01			0.760 ± 0.005	0.818 ± 0.003
	1.05			0.758 ± 0.011	0.816 ± 0.006
	1.1			0.755 ± 0.014	0.814 ± 0.009
MIMIC-II	1.005	0.886	0.803	0.886 ± 0.001	0.803 ± 0.002
	1.01			0.886 ± 0.001	0.802 ± 0.002
	1.05			0.885 ± 0.002	0.794 ± 0.005
	1.1			0.884 ± 0.003	0.784 ± 0.009

Table 7: Test accuracy and AUC comparison between $\boldsymbol{\omega}^*$ and $\boldsymbol{\omega}$ sampled from the approximated Rashomon set with respect to different θs.

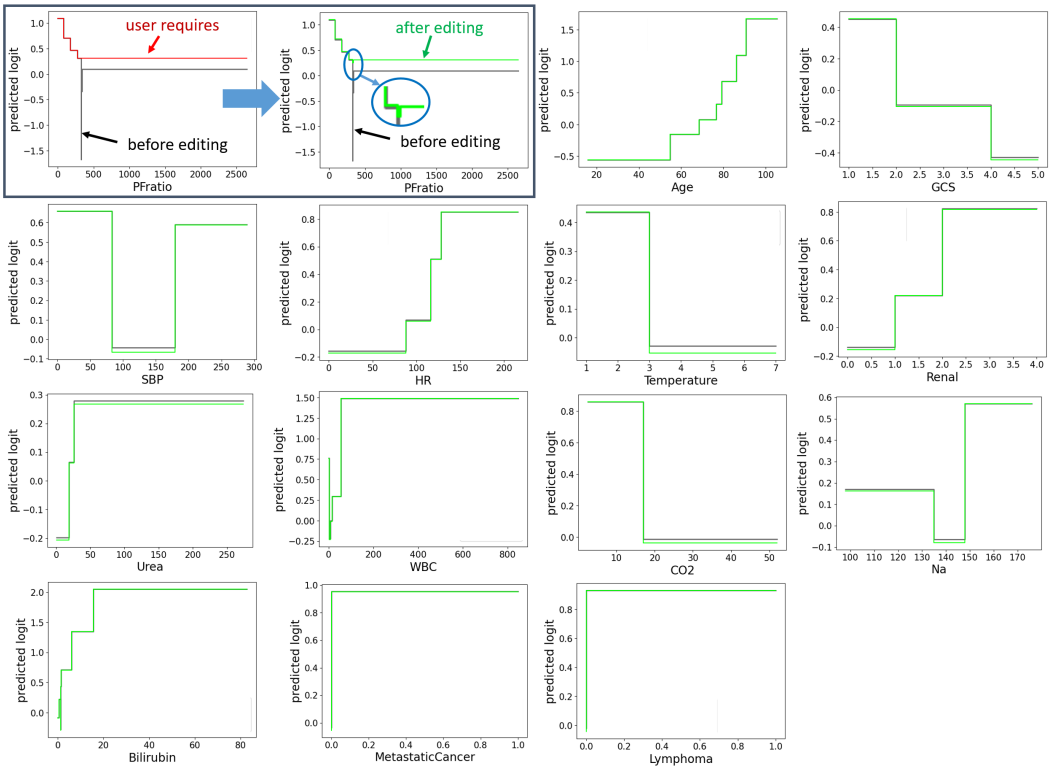
the green curve in the top-mid subfigure in Figure 17a). Another user might not like this idea and prefers to remove the jump by connecting to the left step (Figure 17b). However, this specified shape function is not within the Rashomon set, and we find the closest solution in green, which still has a small jump at 330 (see the inset plot). Different user-specified shape functions lead to different solutions. The Rashomon set can serve as a small but computationally efficient space in which users can find a model that is closest to their needs.

H Test performance

We now show the test performance of models sampled from our approximated Rashomon set. We compare the test accuracy and AUC between $\boldsymbol{\omega}^*$ and $\boldsymbol{\omega}$s sampled from $\hat{R}$ on the four datasets with different values of θ and results are shown in Table 7. We sample 1000 $\boldsymbol{\omega}$ from $\hat{R}$ and show the average and one standard deviation. The larger value of θ leads to a larger Rashomon set, which means we allow models with higher loss. Therefore, as θ increases, both accuracy and AUC decrease. But when the constant is slightly larger than 1, such as 1.005 and 1.01, the test performance of models sampled from $\hat{R}$ usually covers the performance achieved by $\boldsymbol{\omega}^*$ in one standard deviation. This means in general our approximated Rashomon set can return a diverse set of models without compromising the performance.



(a) Case 1



(b) Case 2

Figure 17: Shape functions on the MIMIC-II dataset after a hypothetical shape function on "PFratio" is requested. The red curve in the top-left subfigure is the requested shape function. The shape function colored in green in the top-middle subfigure is the closest shape function within $\hat{R}$.