Supplementary Material: Interpretable Regional Descriptors: Hyperbox-Based Local Explanations

Susanne Dandl[1,2], Giuseppe Casalicchio[1,2], Bernd Bischl[1,2], and Ludwig Bothmann ✉ [1,2]

[1] Department of Statistics, LMU Munich, Ludwigstr. 33, 80539 Munich, Germany
[2] Munich Center for Machine Learning (MCML), Munich, Germany
Ludwig.Bothmann@stat.uni-muenchen.de

A Application Examples

In addition to the credit application in Section 1, we show in the following a medical and jurisdictional application.

Medical Consider an ML model that predicts if a person will develop diabetes in the future. (For simplicity, we assume this model accurately approximates real world relationships.) In the following, we discuss two cases:

(1) A person that is predicted to develop diabetes wants to know why this is the case and what can be options to prevent this. There are different potential actions to take: more sport, less red meat, homeopathic medicine, etc. The IRD can tell which action is not promising, e.g., sports when all realistic amounts of sport are inside the box. However, changing the diet might be an option, because changing the diet by just eating meat one day a week is not part of the box (concrete strategies for prevention can reveal counterfactual explanations).

(2) A person that is predicted not to develop diabetes wants to know how flexible their life-style is without changing the prediction. It may be okay for a person to gain weight without having a higher risk of developing diabetes, as long as they do not change their diet towards including more red meat.

Jurisdiction Consider an ML model that predicts if a person will commit a crime in the next 2 years. A person that gets a high score wants to know why. IRDs that do not contain all groups of protected attributes, such as gender, can indicate unfair discrimination against these groups. Hence, IRDs can initiate further investigations on fairness and biases of an ML model.

B Proof of Theorem 1

Proof. Given a feature X_j that is not involved in the prediction model $\hat{f}$ such that $\forall \tilde{\mathbf{x}} \in \mathcal{X} \wedge \forall x_j \in \mathcal{X}_j$:

$$\hat{f}(\tilde{x}_1, ..., \tilde{x}_{j-1}, \tilde{x}_j, \tilde{x}_{j+1}, ..., \tilde{x}_p) = \hat{f}(\tilde{x}_1, ..., \tilde{x}_{j-1}, x_j, \tilde{x}_{j+1}, ..., \tilde{x}_p), \tag{1}$$

and given a box B for $\mathbf{x}'$ that is maximal according to Definition 1. We assume now that Theorem 1 does not hold such that $B_j = [l_j, u_j] \subset \mathcal{X}_j$. However, since Eq. (1) holds, either $(\exists\, x_j \in \mathcal{X}_j \wedge x_j < l_j : \mathit{precision}(B \cup [x_j, l_j]) = 1)$, or $(\exists\, x_j \in \mathcal{X}_j \wedge x_j > u_j : \mathit{precision}(B \cup [u_j, x_j]) = 1)$ for numeric X_j or $(\exists\, x_j \in \mathcal{X}_j \setminus B_j : \mathit{precision}(B \cup x_j) = 1)$ for categorical X_j holds which contradicts the maximality assumption of B.

C Proof of Theorem 2

Proof. Given a box B with $\mathit{precision}(B) = 1$ and $\mathbf{x}' \in B$, and given a feature X_j that is relevant for $\hat{f}(x')$ such that $\exists\, x_j \in \mathcal{X}_j \backslash B_j : \hat{f}(x'_1, ..., x'_{j-1}, x_j, x'_{j+1}, ..., x'_p) \notin Y'$. We assume now that Theorem 2 does not hold, such that $B_j = \mathcal{X}_j$. This contradicts the statement that $\mathit{precision}(B) = 1$ because x_j that leads to a prediction $\notin Y'$ for $\mathbf{x}'$ is also covered by the box.

D Proof of Theorem 3

Proof. Without loss of generality, we assume that we only have numeric features. Assume we computed $\bar{\underline{B}} = \bigcup_{j=1}^{p} [l_j, u_j]$ such that $\forall j \in \{1, ...p\}$:

$$\hat{f}(\underbrace{x'_1, .., x'_{j-1}, l_j, x'_{j+1}, ..., x'_p}_{:=\mathbf{x}'_l}) \notin Y' \wedge \hat{f}(\underbrace{x'_1, .., x'_{j-1}, u_j, x'_{j+1}, ..., x'_p}_{:=\mathbf{x}'_u}) \notin Y'.$$

We assume that $B \subset \bar{\underline{B}}$ is not true for now such that there is a homogeneous B with $min(B_j) < l_j$ or $max(B_j) > u_j$ and $\mathbf{x}' \in B$. However, then either $\mathbf{x}'_l$ or $\mathbf{x}'_u$ would also be part of B but for both $\hat{f}(\mathbf{x}'_u) \notin Y'$ or $\hat{f}(\mathbf{x}'_l) \notin Y'$ holds, which contradicts that B is homogeneous.

E Pseudocode and Illustrations of IRD Methods

E.1 Pseudocode

Algorithm 1 Adapted MaxBox approach [2]

Input: Targeted instance $\mathbf{x}'$, desired range Y', prediction model $\hat{f} : \mathcal{X} \to \mathbb{R}$, input dataset $\bar{\mathbf{X}}$, initial box B
Initialize candidates = [ ], upper_bound_coverage_best = -Inf, current_best = [ ]
if $\exists\, \mathbf{x} \in \bar{\mathbf{X}} \wedge \mathbf{x} \in B : \hat{f} \notin Y'$ **then**
  candidates = candidates.*append*(B)
  while *length*(candidates) > 0 **do**
    B^{best} = *choose_best*(candidates)
      ▷ if upper_bound_coverage_best < 0, B^{best} corresponds to the box with the most no. of shrinking steps done before (with the upper bound of the coverage as a tiebreaker), else, B^{best} corresponds to the box that maximizes $\left(\frac{|\{\mathbf{x} \in B | \hat{f}(\mathbf{x}) \in Y\}|}{|\{\mathbf{x} \in B | \hat{f}(\mathbf{x}) \notin Y\}|}\right)$.
    candidates = candidates.*remove*(B^{best})
    children = *create_new_candidates*(B^{best}) ▷ in Figure S. 1, C and D are new candidates created from the initial box
    for $B \in$ children **do**
      if $\forall \mathbf{x} \in B : \hat{f}(\mathbf{x}) \in Y'$ **then**
        coverage = *upper_bound_coverage*(B)
        if coverage > upper_bound_coverage_best **then**
          current_best = B
          upper_bound_coverage_best = coverage
        end if
      else
        if *upper_bound_coverage*(B) > upper_bound_coverage_best **then**
          candidates = candidates.*append*(B)
        end if
      end if
    end for
  end while
else
  current_best = B
end if
return current_best

Algorithm 2 Adapted PRIM approach [3]

Input: Targeted instance $\mathbf{x}'$, desired range Y', prediction model $\hat{f} : \mathcal{X} \to \mathbb{R}$, input dataset $\bar{\mathbf{X}}$, initial box B
while $\exists\, \mathbf{x} \in \bar{\mathbf{X}} \wedge \mathbf{x} \in B : \hat{f} \notin Y'$ **do**
  for $j \in \{1, ..., p\}$ **do**
    $C_j = [\,]$ ▷ create candidates for peeling
    if X_j numeric **then**
      $C_j = C_j.append(B_j^-, B_j^+)$ where $B_j^- = [l_j, min(X_{j(\alpha)}, x'_j)]$ and
$B_j^+ = [max(X_{j(1-\alpha)}, x'_j), u_j]$ with $x_{j(\alpha)}$ and $x_{j(1-\alpha)}$ as the α- and $(1-\alpha)$-quantiles of X_j in the current box B
    else if X_j categorical **then**
      $C_j = \{s \in B_j \mid s \neq x'_j\}$
    end if
  end for
  $b^{best} = \underset{b \in C_j,\, j \in \{1,...,p\}}{\arg\max} \; precision(B \setminus b)$
  $B = B \setminus b^{best}$
end while
homogeneous = TRUE
while homogeneous **do**
  for $j \in \{1, ..., p\}$ **do**
    $C_j = [\,]$ ▷ create candidates for pasting
    if X_j numeric **then**
      inbox $= \{\mathbf{x} \in \bar{\mathbf{X}} \mid x_k \in B_k\}$, for $k \in \{1, ..., j-1, j+1, ...p\}$
      number_added $= |\{\mathbf{x} \in \bar{\mathbf{X}} \mid \mathbf{x} \in B\}| \cdot \alpha$
      $C_j = C_j.append(B_j^-, B_j^+)$ with $B_j^- = [x_j^l, l_j]$ and $B_j^+ = [u_j, x_j^u]$ with
x_j^l as the jth feature value of the (number_added)th observation $\mathbf{x} \in$ inbox with a value x_j lower than l_j and
x_j^u as the jth feature value of the (number_added)th observation $\mathbf{x} \in$ inbox with a value x_j higher than u_j

    else if X_j categorical **then**
      $C_j = \{s \in X_j \mid s \notin B_j\}$
    end if
    $C_j = \{b \in C_j \mid precision(B \cup b) = 1\}$
  end for
  if $\exists\, j \in \{1, ..., p\} : |C_j| > 0$ **then**
    $b^{best} = \underset{b \in C_j,\, j \in \{1,...,p\}}{\arg\max} \; coverage(B \setminus b)$
    $B = B \cup b$
  else
    homogeneous = FALSE
  end if
end while
return B

Algorithm 3 Adapted MAIRE approach [7]

Input: Targeted instance $\mathbf{x}'$, desired range Y', prediction model $\hat{f} : \mathcal{X} \to \mathbb{R}$, input dataset $\bar{\mathbf{X}}$, initial box B, precision threshold τ (default 1), maximum number of iterations max_iterations (default 100)
Scale all feature values of $\mathbf{x} \in \bar{\mathbf{X}}$ and $\mathbf{x}'$ to 0-1 range
best_coverage = 0
converged = FALSE
best_candidate = B
$i = 0$
while $i \leq max\_iterations$ **do**
  $B = optimize\_with\_adam(B)$
    ▷ optimizes differentiable versions of coverage, precision and locality
  if $precision(B) \geq \tau \wedge coverage(B) \geq$ best_coverage **then**
    best_candidate = B
  else if $precision(B) < \tau$ **then**
    converged = TRUE
  end if
  if converged = TRUE **then**
    i = i + 1
  end if
end while
return best_candidate

E.2 Illustrations

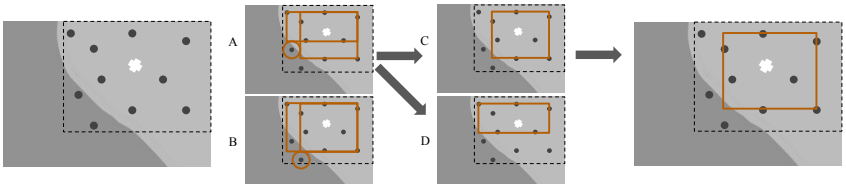


Fig. S. 1: Illustration of the adapted MaxBox algorithm. The algorithm starts with $\bar{\mathbf{B}}$ (dashed box). In the box are two data points with predictions $\notin Y'$ (called negative samples) and the box needs to be further optimized. First, a negative sample is chosen - either the one in A or B. Therefore, the number of samples with predictions $\in Y'$ after excluding the points in one feature dimension are inspected. The resulting boxes of both negative samples cover a maximum of seven samples. We chose the one of A (B is also fine). Its resulting boxes are the new subproblems/candidates (C and D). Both boxes in C and D only include samples with predictions $\in Y'$, but the box in C is chosen as an optimum because it includes more samples with predictions $\in Y'$. D is discarded because it has a lower number. Since C and D cannot be further split because no negative samples are within both boxes, the returned box by MaxBox is the box in C.

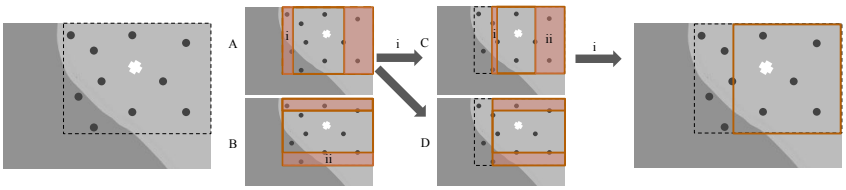


Fig. S. 2: Illustration of the adapted PRIM algorithm. The algorithm starts with $\bar{B}$. In the first iteration, there exist four potential subboxes (two in each feature dimension (A vs. B)) that could be removed. The subbox i is chosen because it has the highest precision but compared to ii it has a smaller size. In the next step (C & D), again four subboxes can be potentially removed. Again, we choose i for the same reason as before. After its removal, the resulting box is at the same time the final box because in the pasting step only one subbox could be added – i again. All other dimensions are maximal.

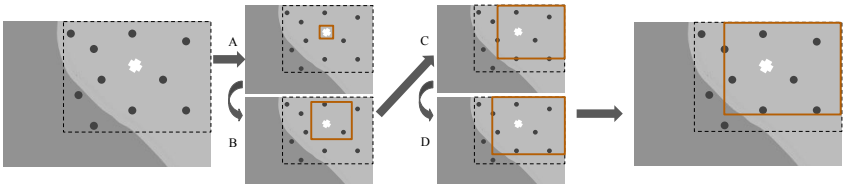


Fig. S. 3: Illustration of the adapted MAIRE algorithm. The algorithm starts with the smallest box possible. The box boundaries are then iteratively enlarged (A-D). The box boundaries are only updated if the precision of the new box = 1.

F Pseudocode of Post-Processing Approach

Algorithm 4 Post-processing algorithm - peeling (inspired by [3])

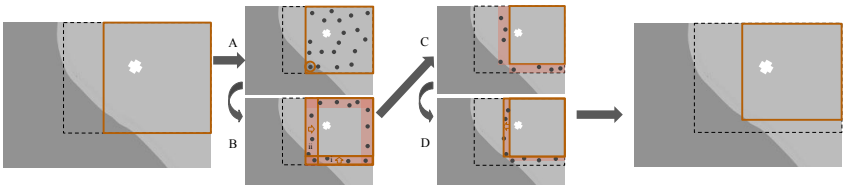
Input: Targeted instance $\mathbf{x}'$, desired range Y', prediction model $\hat{f} : \mathcal{X} \to \mathbb{R}$, initial box B, number of samples for evaluation M (default 100), relative subbox size of continuous features α (default 0.1)
for $j \in \{1, ..., p\}$ **do**
  if X_j numeric **then**
    $s_j = (max(\mathcal{X}_j) - min(\mathcal{X}_j)) \cdot \alpha$ ▷ derive subbox sizes for numeric features based on $\mathcal{X}$
    if X_j integer **then**
      $s_j = round(s_j)$
    end if
  end if
end for
$\bar{\mathbf{X}} = sample\_uniformly(B, n = M \cdot 5)$ ▷ sample new data to check if B homogeneous
if $\exists \mathbf{x} \in \bar{\mathbf{X}} \wedge \mathbf{x} \in B : \hat{f} \notin Y'$ **then**
  not_homogeneous = TRUE ▷ start peeling
  while not_homogeneous **do**
    for $j \in \{1, ..., p\}$ **do**
      $C_j = [\,]$ ▷ create candidates for peeling
      if X_j numeric **then**
        $C_j = C_j.append(B_j^-, B_j^+)$
where $B_j^- = [l_j, min(l_j + s_j, x'_j)]$ and $B_j^+ = [max(u_j - s_j, x'_j), u_j]$
      else if X_j categorical **then**
        $C_j = \{s \in B_j \mid s \neq x'_j\}$
      end if
      $C_j = \{b \in C_j \mid precision(B_j^b) < 1\}$ with $B_j^b = (B_1 \times ... \times B_{j-1} \times b \times B_{j+1} \times ... \times B_p)$
    end for
    if $\exists\, j \in \{1, ..., p\} : |C_j| > 0$ **then**
      $b^{best} = \underset{b \in C_j,\, j \in \{1,...,p\}}{\arg\max} precision\_to\_boxsize(B_j^b)$ ▷ evaluate on M new instances sampled within B_j^b
      $B^{best} = (B_1 \times ... \times B_{j-1} \times b^{best} \times B_{j+1} \times ... \times B_p)$ ▷ choose the one with lowest precision relative to size
      $B = B^{best}$
    else
      not_homogeneous = FALSE
    end if
  end while
end if
return B, $\mathbf{s} = \{s_j \mid X_j \text{ numeric}\}$

Algorithm 5 Post-processing algorithm - pasting (inspired by [3])

Input: Targeted instance $\mathbf{x}'$, desired range Y', prediction model $\hat{f} : \mathcal{X} \to \mathbb{R}$, initial box B (potentially peeled), number of samples for evaluation M (default 100), relative subbox size of continuous features α (default 0.1), lower threshold for relative subbox size α_0 (default 0.05), subbox sizes of numeric features $\mathbf{s}$

homogeneous = TRUE ▷ start pasting
stepsize = 1
while homogeneous **do**
  for $j \in \{1, ..., p\}$ **do**
    $C_j = [\,]$ ▷ create candidates/subboxes for pasting
    if X_j numeric **then**
      $C_j = C_j.append(B_j^-, B_j^+)$
where $B_j^- = [l_j - \text{stepsize} \cdot s_j, l_j]$ and $B_j^+ = [u_j, u_j + \text{stepsize} \cdot s_j]$
    else if X_j categorical **then**
      $C_j = \{s \in X_j \mid s \notin B_j\}$
    end if
    $C_j = \{b \in C_j \mid precision(B_j^b) = 1\}$ with $B_j^b = (B_1 \times ... \times B_{j-1} \times b \times B_{j+1} \times ... \times B_p)$
  end for
  if $\exists\, j \in \{1, ..., p\} : |C_j| > 0$ **then**
    $b^{best} = \underset{b \in C_j,\, j \in \{1,...,p\}}{\arg\max} \; size(B_j^b)$ ▷ evaluate on M new instances sampled within B_j^b
    $B = B \cup b$ ▷ choose largest one with precision 1
  else
    if stepsize $\geq \alpha_0$ **then**
      stepsize = stepsize/2 ▷ if no box with precision 1 exists, consider reducing the subbox sizes
    else
      homogeneous = FALSE
    end if
  end if
end while
return B

Fig. S. 4: Illustration of the post-processing algorithm. The algorithm starts with the box generated by another method (solid brown box, which is a subbox of the dashed box $\bar{\mathrm{B}}$). First, new points are sampled and it is assessed whether the box is homogeneous (A). If not, the subboxes with the lowest precision compared to their size are peeled iteratively (B). The precision is assessed based on newly sampled points within the subboxes. First subbox i is peeled then subbox ii (both contain a sample with a prediction $\notin Y'$). If no subbox with precision < 1 exists, it is assessed whether the box could be further enlarged (C). If all considered subboxes have precisions < 1, the subbox sizes are halved (D) as long as the relative subbox size does not fall below a threshold.



G Level Set Identification

The algorithm by Kuratomi et al. [5] starts at $\mathbf{x}'$ and tries to find a connection $\in Y'$ between the nominal, then the ordinal, and then the continuous features of $\mathbf{x}$ and $\mathbf{x}'$. If a path is found, $\mathbf{x}$ is part of $\mathcal{L}$. For categorical features, all permutations of feature orders are inspected.[1] For continuous features, the shortest linear path for a given number of equidistant steps is checked. Kuratomi et al. [5] used DBSCAN, for which the choice of the maximum distance threshold is ambiguous. The identification algorithm has a complexity of $O(c! \cdot c + o! \cdot \sum_{j=1}^{o} k_j + q)$ with c and o as the number of nominal and ordinal features, respectively, k_j as the number of possible values of an ordinal feature X_j and q as the number of inspected steps for continuous features.

The level set could be further enriched by attempting to find connections between the unconnected and connected points. For the comparison of IRD methods, however, a convex level set is sufficient, since the hyperbox itself is convex.

H Tuning of ML models

For hyperparameter tuning, we used random search (with 15 evaluations), and 5-fold cross-validation (CV) with the misclassification error (classification) or mean squared error (regression) as a performance measure. Table S. 1 shows the tuning search space of each model. The rather limited tuning setup should be sufficient

[1] If the number of permutations exceeds 100 permutations, 100 feature orders are randomly chosen.

for our task of explaining a prediction model – a less accurate model is not a hindrance. Unbalanced datasets such as *tic_tac_toe*, *diabetes* and *cmc* were balanced with the SMOTE algorithm [1]. For SMOTE, numeric features were standardized and categorical ones were one-hot encoded. The optimizer for the neural network was ADAM [4] with 500 epochs. For all other hyperparameters, the default values of the mlr3keras R package were used [6] (apart from the no. of layer units, see Table S. 1). Table S. 2 shows the accuracies of each model using nested resampling with 5-fold CV in the inner and outer loop).

Table S. 1: Tuning search space of each model. Hyperparameter values of *num.trees* were log-transformed.

Model	Hyperparameter	Range
random forest	num.trees	[1, 1000]
logistic regression	-	-
linear model	-	-
multi-nomial model	-	-
hyperbox/rpart	-	-
neural net	layer_units	[1, 20]

Table S. 2: Classification error or mean squared error (regression) of each model on each dataset. The performances were computed using nested resampling with 5-fold CV in the inner and outer loop. We did not measure the performance of the (terminal node) hyperbox model because the model differs for each $\mathbf{x}'$.

	Random forest	Linear model	Neural net	Hyperbox
diabetes	0.233	0.224	0.229	-
tic_tac_toe	0.036	0.019	0.094	-
cmc	0.466	0.495	0.389	-
vehicle	0.256	0.201	0.254	-
no2	33502.856	37678.319	77866.331	-
plasma_retinol	45391.218	59224.452	297481.249	-

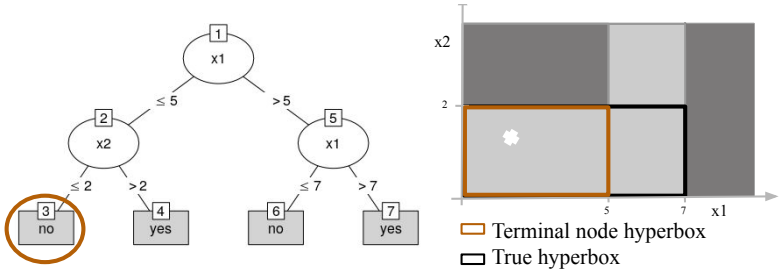


Fig. S. 5: True hyperbox vs. terminal node hyperbox for a CART tree. The white cross corresponds to $\mathbf{x}'$.

I Benchmark - Additional Results

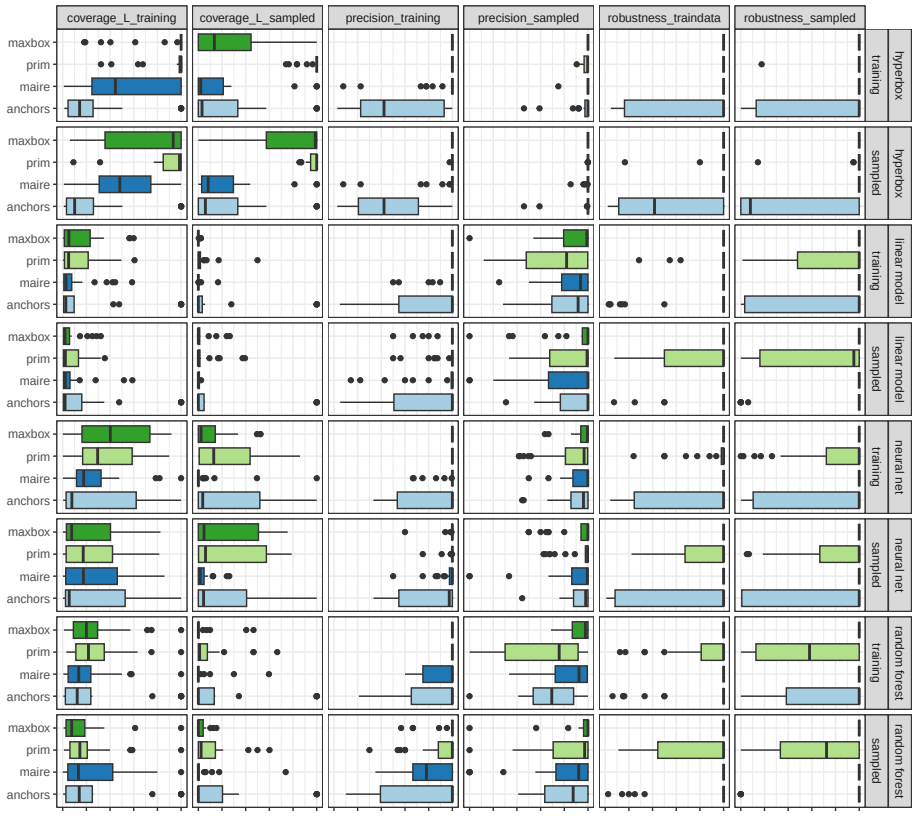


Fig. S. 6: Comparison of MaxBox, PRIM, Anchors, and MAIRE w.r.t. coverage and precision for each model separately. Each method was either run or evaluated on training data or uniformly sampled data from $\bar{\mathbb{B}}$ *without* post-processing. Higher values for precision and coverage are better.

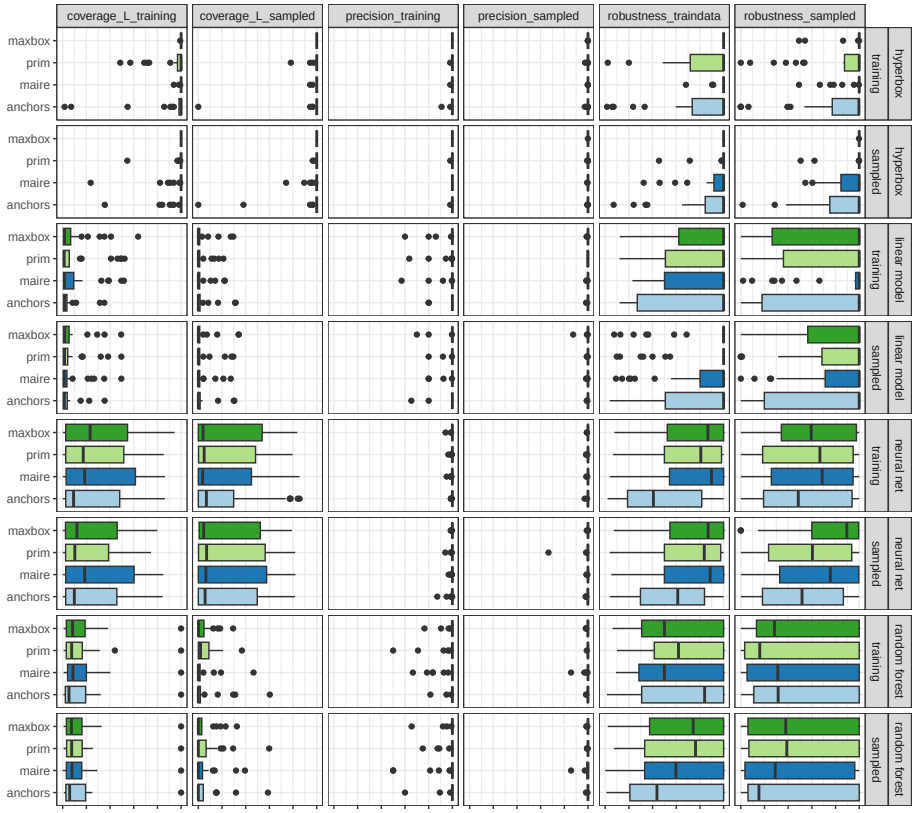


Fig. S. 7: Comparison of MaxBox, PRIM, Anchors, and MAIRE w.r.t. coverage and precision for each model separately. Each method was either run or evaluated on training data or uniformly sampled data from $\bar{\underline{\mathrm{B}}}$ *with* post-processing. Higher values for precision and coverage are better.

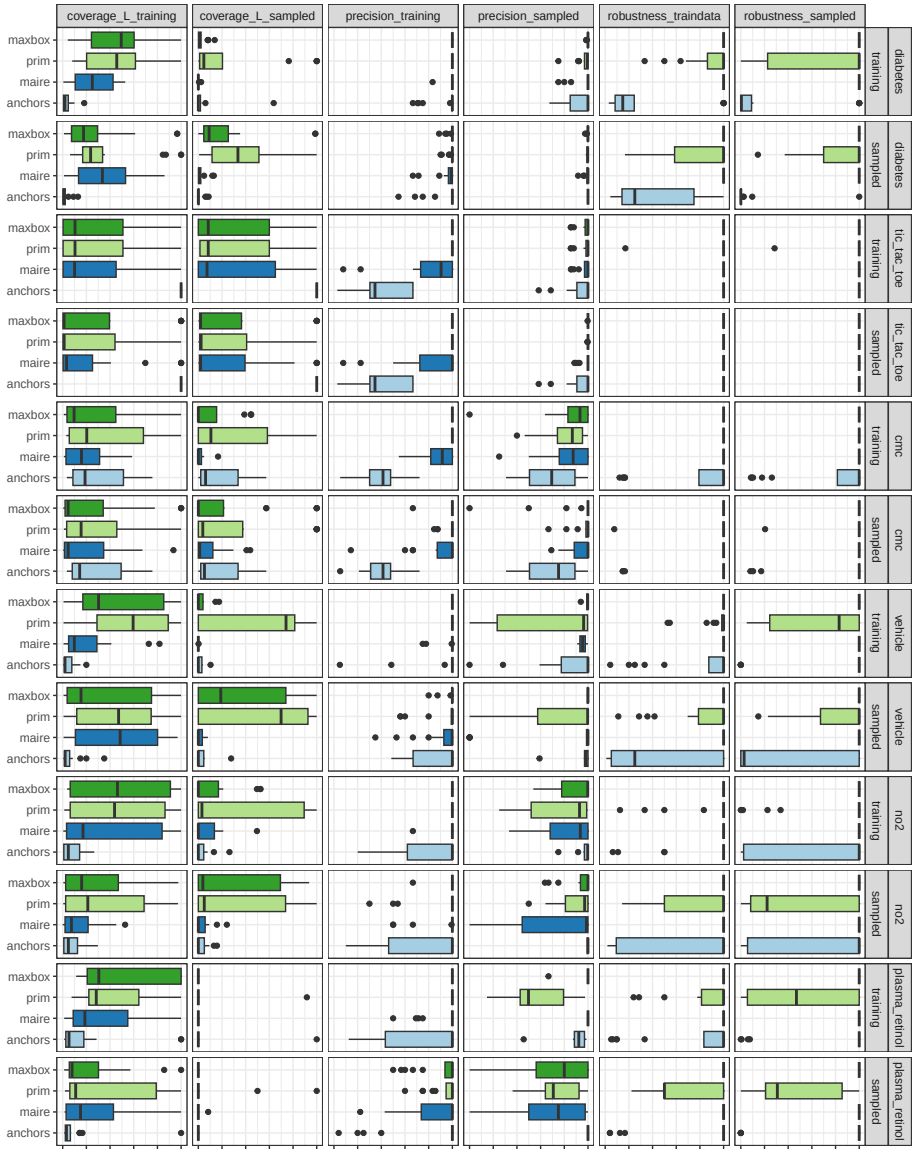


Fig. S. 8: Comparison of MaxBox, PRIM, Anchors, and MAIRE w.r.t. coverage and precision for each dataset separately. Each method was either run or evaluated on training data or uniformly sampled data from $\bar{\underline{B}}$ *without* post-processing. Higher values for precision and coverage are better.

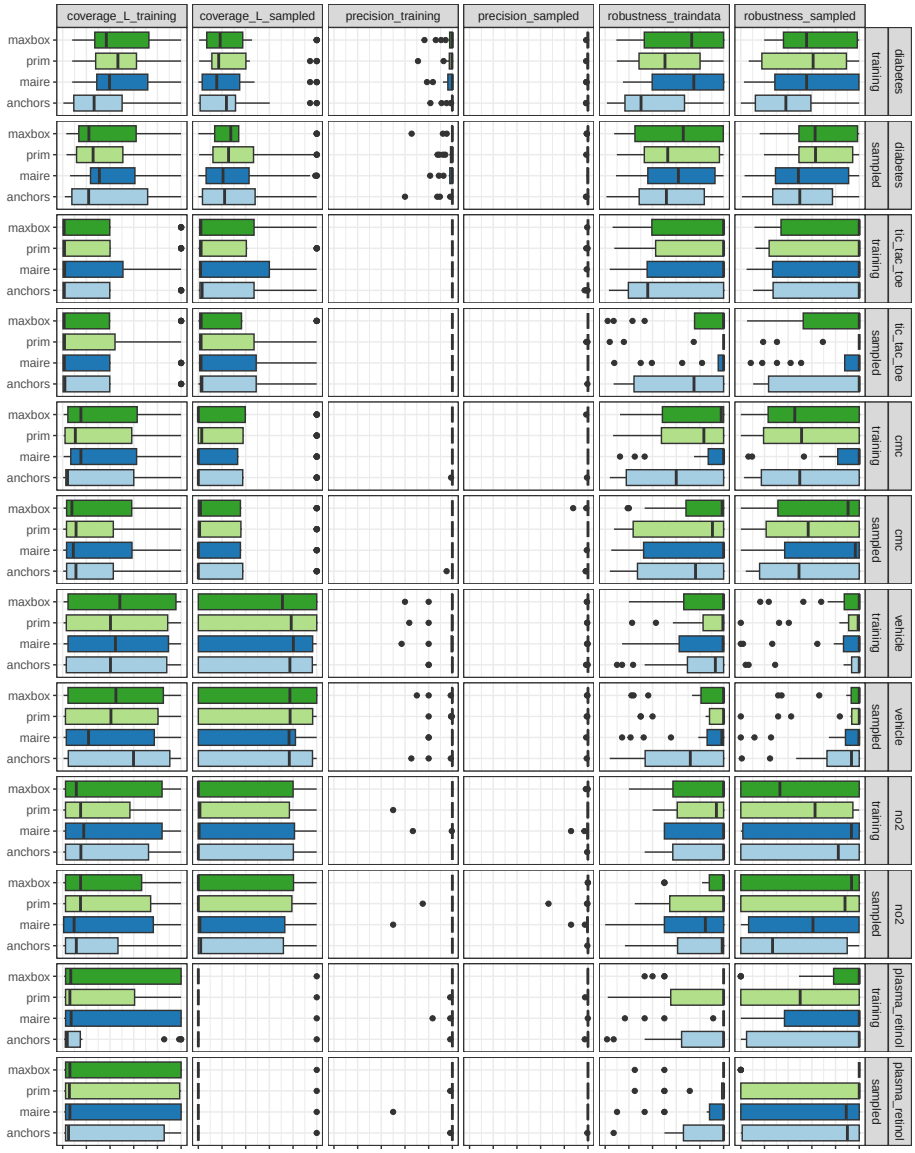


Fig. S. 9: Comparison of MaxBox, PRIM, Anchors, and MAIRE w.r.t. coverage and precision for each dataset separately. Each method was either run or evaluated on training data or uniformly sampled data from $\bar{\mathbb{B}}$ *with* post-processing. Higher values for precision and coverage are better.

Table S. 3: Statistical analysis of RQ 1. Pairwise comparison of MaxBox, PRIM, Anchors, and MAIRE w.r.t. coverage and precision. Each value corresponds to the p-value obtained for the Wilcoxon rank-sum test with H_0 that the performances do not differ. Cells printed in bold font correspond to p-values that are lower than $\alpha = 0.05/36$ (Bonferroni-adjustment) and indicate that one method outperforms the other. Only methods run on the training data without post-processing were compared.

measure	MaxBox = PRIM	MaxBox = Anchors	MaxBox = MAIRE	PRIM = Anchors	PRIM = MAIRE	Anchors = MAIRE
coverage_training	0.761	0.618	**0**	0.579	**0**	0.473
coverage_sampled	**0**	0.044	**0**	**0**	**0**	**0**
coverage_$\mathcal{L}$_training	0.431	**0.001**	**0**	**0**	**0**	0.127
coverage_$\mathcal{L}$_sampled	**0**	0.035	0.004	0.059	**0**	**0**
precision_training	1	**0**	**0**	**0**	**0**	**0**
precision_sampled	0.025	**0**	0.623	0.042	0.104	0.004

Table S. 4: Statistical analysis of RQ 2. Pairwise comparison of using training data vs. sampled data for $\bar{\mathbf{X}}$. Each value corresponds to the p-value obtained for the Wilcoxon rank-sum test with H_0 that the performance of methods using training data is better than the performance of methods using sampled data. Cells printed in bold font correspond to p-values that are lower than $\alpha = 0.05/30$ (Bonferroni-adjustment) and indicate a preference towards using sampled data. Comparisons were only conducted for the methods run without post-processing.

measure	overall	MaxBox	PRIM	Anchors	MAIRE
coverage_training	1.00	1	0.999	0.445	0.744
coverage_sampled	0.00	**0**	0.987	0.402	0.003
coverage_$\mathcal{L}$_training	1.00	1	1	0.781	0.896
coverage_$\mathcal{L}$_sampled	0.00	**0**	0.236	0.476	0.172
precision_training	1.00	0.995	0.998	0.782	0.993
precision_sampled	0.00	0.011	**0**	**0.001**	0.381

Table S. 5: Statistical analysis of RQ 3. Pairwise comparison of using no post-processing vs. using post-processing. Each value corresponds to the p-value obtained for the Wilcoxon rank-sum test with H_0 that the performance of methods using no post-processing is better than the performance of methods using post-processing. Cells printed in bold font correspond to p-values that are lower than $\alpha = 0.05/60$ (Bonferroni-adjustment) and indicate a preference towards post-processing.

method	coverage_training	coverage_sampled	coverage_$\mathcal{L}$_training	coverage_$\mathcal{L}$_sampled	precision_training	precision_sampled
traindata	0.95	0	0.369	0	0	0
MaxBox	0.97	**0**	0.982	**0**	0.995	0.003
PRIM	1.00	1	1	0.452	0.999	**0**
Anchors	0.92	0.001	0.065	0.054	**0**	**0**
MAIRE	0.10	**0**	0.003	**0**	**0**	0.001
sampled	0.12	0	0	0	0	0
MaxBox	0.00	**0**	**0**	**0**	0.085	0.021
PRIM	0.45	0.19	0.262	0.468	0.003	0.001
Anchors	0.92	**0**	0.035	0.061	**0**	**0**
MAIRE	0.18	**0**	0.003	**0**	**0**	0.009

References

1. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research **16**, 321–357 (2002). https://doi.org/10.1613/jair.953
2. Eckstein, J., Hammer, P.L., Liu, Y., Nediak, M., Simeone, B.: The maximum box problem and its application to data analysis. Computational Optimization and Applications **23**(3), 285–298 (2002). https://doi.org/10.1023/a:1020546910706
3. Friedman, J.H., Fisher, N.I.: Bump hunting in high-dimensional data. Statistics and Computing **9**(2), 123–143 (1999). https://doi.org/10.1023/A:1008894516817
4. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv 1412.6980 v9, arXiv.org E-Print Archive (2017). https://doi.org/10.48550/arXiv.1412.6980
5. Kuratomi, A., Miliou, I., Lee, Z., Lindgren, T., Papapetrou, P.: JUICE: JUstIfied Counterfactual Explanations. In: Pascal, P., Ienco, D. (eds.) Discovery Science. pp. 493–508. Lecture Notes in Computer Science, Springer Nature Switzerland, Cham (2022). https://doi.org/10.1007/978-3-031-18840-4_35
6. Pfisterer, F., Poon, J., Lang, M.: mlr3keras: mlr3 keras extension. Github repository. URL https://github.com/mlr-org/mlr3keras (2022), Commit: bad8434b7898b51b2143fc680594057c00dc7080
7. Sharma, R., Reddy, N., Kamakshi, V., Krishnan, N.C., Jain, S.: MAIRE - a model-agnostic interpretable rule extraction procedure for explaining classifiers. In: Holzinger, A., Kieseberg, P., Tjoa, A.M., Weippl, E. (eds.) Machine Learning and Knowledge Extraction, vol. 12844, pp. 329–349. Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-84060-0_21, Series Title: Lecture Notes in Computer Science