RbX: Region-based explanations of prediction models

Anonymous authors
Paper under double-blind review

Abstract

We introduce region-based explanations (RbX), a novel, model-agnostic method to generate local explanations of scalar outputs from a black-box prediction model using only query access. RbX is based on a greedy algorithm for building a convex polytope that approximates a region of feature space where model predictions are close to the prediction at some target point. This region is fully specified by the user on the scale of the predictions, rather than on the scale of the features. The geometry of this polytope — specifically the change in each coordinate necessary to escape the polytope — quantifies the local sensitivity of the predictions to each of the features. These "escape distances" can then be standardized to rank the features by local importance. RbX is guaranteed to satisfy a "sparsity" axiom, which requires that features which do not enter into the prediction model are assigned zero importance. At the same time, real data examples and synthetic experiments show how RbX can more readily detect all locally relevant features than existing methods.

1 Introduction

Suppose we have a prediction model $\hat{f}(x)$ to estimate a scalar outcome y given a set of features $x \in \mathbb{R}^d$. We do not assume that $\hat{f}$ takes any particular functional form. Rather, we consider $\hat{f}$ as a black-box function to which we only have query acccess. That is, we may compute the value of $\hat{f}(x)$ for any desired input x, but do not have any additional information about the structure of the model $\hat{f}$.

After making a prediction at a target point x_0, we seek to quantify the local importance of each feature on the prediction. Of those features involved, some will be more influential than others in the predictions from $\hat{f}$ near x_0. We would like a systematic way of identifying these.

As a motivating example, consider a loan borrower applying for credit. Statistical learning techniques are increasingly carried out to assess credit-worthiness of prospective applicants (Chen et al., 2018a). For example, credit-granting agents may collect many features about the applicant including age, gender, occupation, etc., and use them to predict their probability of default. If the bank denies their request, it is natural for the user to ask: *Are there any critical features that the applicant could change to improve their outcome?* We return to this setting in Section 5.1, with a real data example based on the FICO Explainable Machine Learning Challenge.

We distinguish our problem, which we call *local prediction importance*, from the questions of feature selection and feature importance. Feature selection methods, such as the LASSO for linear models (Tibshirani, 1996) and modern extensions like LassoNet for black-box models (Lemhadri et al., 2021), aim to select a small subset of features to generate a predictive model with greater accuracy and/or interpretability. In our setting, the prediction model $\hat{f}$ is given, and we seek only to faithfully explain the predictions of that model, without regard to the unknowable data-generating process that created the features and response. Feature importance methods include popular permutation-based approaches introduced by (Breiman, 2001) for random forests, which were extended to generic black-box models by (Fisher et al., 2019) and to a local method by (Casalicchio et al., 2018). These approaches fix the prediction model $\hat{f}$, but provide importance measures based on changes in the predictive performance of that model, and thus also depend on the data-generating process. By contrast, the term *prediction importance* emphasizes the singular role of the structure of prediction model $\hat{f}$, independent of how the model was trained or validated.

The distinction between local prediction importance and local feature importance is not always made in the literature. However, it is relevant for a user who only cares about understanding the output of a given black-box model, and does not want prediction explanations conflated with the underlying signal the model is trying to approximate.

2 WHY REGION-BASED EXPLANATIONS?

Our proposed approach to local prediction importance is via region-based explanations (RbX). RbX is "model-agnostic," meaning it does not require any knowledge about the structure of $\hat{f}$. We defer a detailed description of the algorithm to Section 3.1, but given a target point x_0, the main idea is to construct a polytope that approximates the region in feature space with prediction values "close" to the prediction at x_0. The user can define this "closeness region" in a way that is relevant for the context in which the model is being deployed. For example, if $\hat{f}$ corresponds to estimated class probabilities in a binary classification setting, the region could consist of all prediction values on the same side of the decision boundary. If $\hat{f}$ predicts a numeric medical outcome, and x_0 corresponds to a healthy patient, then the region might be the accepted range of healthy outcomes.

Given this polytope approximation of the closeness region, we then argue that distances from x_0 to the boundaries of this polytope in directions parallel to the coordinate axes inform the local sensitivities of $\hat{f}$ to each feature in a desirable way. The approach is perhaps best motivated when the features are continuous, but works for ordered categorical features (including binary features) as well.

2.1 PREVIOUS WORK

Existing approaches to local prediction importance can be broadly divided into two categories: surrogate methods and gradient-based methods. Surrogate methods locally approximate $\hat{f}$ by fitting a simpler prediction model that treats the predictions of $\hat{f}$ in a region near x_0 as the response. The weights assigned to each feature in this model are then used for local importance. For instance, LIME (Ribeiro et al., 2016) draws feature instances from a density centered at the target point x_0 and uses a linear surrogate. Lundberg & Lee (2017) propose Kernel SHAP (hereafter just SHAP), which they showed is an algorithmic approximation to fitting an additive surrogate model with weights corresponding to Shapley values.

Gradient-based methods consider infinitesimal regions on the decision surface and use the resulting first-order approximation to derive local feature importance. For example, Baehrens et al. (2010) provide local prediction importances based on the absolute value of the components of the gradient vector $\nabla \hat{f}(x_0)$; their approach for estimating this gradient is by fitting a global surrogate model using Parzen windows. Integrated gradients (Sundararajan et al., 2017) considers the line integral of the components of the gradient of $\hat{f}$ over a straight line path in feature space from a baseline point x to x_0. Other gradient methods are not model-agnostic. For instance, DeepLIFT (Shrikumar et al., 2017) relies on backpropagation to estimate gradients in neural networks.

2.2 SPARSITY AND DETECTION POWER

RbX is designed to satisfy two properties, loosely analogous to type I error control and power in classical hypothesis testing.

Property 1. *(Sparsity) A feature not involved in the prediction model $\hat{f}$ is assigned no importance.*

Sparsity says we don't want a local prediction importance method to make any "false discoveries" by asserting that a completely irrelevant feature is important. Of course, sparsity is not *sufficient* for a *good* local prediction importance method. We also want "detection power":

Property 2. *(Detection power) Any locally relevant feature for $\hat{f}$ has nonzero importance.*

Non-axiomatic local prediction importance methods such as LIME and L2X (Chen et al., 2018b) — a method that computes local feature scores by maximizing a variational relaxation of the mutual information between y and the features x encoded by $\hat{f}$ — evaluate their methods based on sparsity and detection power. For instance, the experiments in Ribeiro et al. (2016) show that LIME does

a better job than some baseline methods in finding the features used in sparse logistic regression models and decision trees. (Chen et al., 2018b) show that L2X does better than LIME, SHAP, and various gradient methods in recovering the relevant features in a sparse signal approximated by a dense neural network.

For the purposes of local prediction importance (rather than local feature importance), what matters is whether a method can recover the relevant features in a sparse *prediction model* (rather than a sparse *signal*). Thus, our synthetic experiments in Section 5.2 replace the dense neural net from Chen et al. (2018b) with sparse models.

SHAP and IG satisfy sparsity axiomatically. LIME and L2X do not. Unlike sparsity, detection power is not precisely defined, due to subjectivity in the definition of "locally relevant". But a reasonable sufficient condition for local relevance of feature j is for the j-th component of $\nabla \hat{f}(x_0)$, the gradient of the prediction at x_0, to be nonzero. Then in the case that $\hat{f}$ is a sparse, additive regression model, Property 1 corresponds to assigning zero importance to all features with zero coefficients, while Property 2 means assigning nonzero importance to all other features. A simple gradient-based method using finite differences would then always perfectly satisfy both properties, as the set of features with nonzero gradients would always be precisely the relevant features. By contrast, LIME only does this 90%-92% of the time in the experiments from Ribeiro et al. (2016).

What remains is to improve detection power in nonlinear models without sacrificing sparsity. In such models, a feature might be locally relevant near the target point x_0, even if $\nabla \hat{f}(x_0) = 0$. Sundararajan et al. (2017) motivate IG in this way, noting that gradient methods fail when $\hat{f}$ has zero gradient with respect to a particular feature, but still varies in that direction within a non-infinitesimal neighborhood that is considered locally relevant.

IG addresses this zero-gradient issue by accumulating gradients along the entire line segment between some baseline feature combination x and the target point x_0. However, this still only detects features that happen to vary infinitesimally somewhere along this line segment. There are a lot of additional areas of the feature space near x_0 where $\hat{f}$ could depend on a given feature. Our approach, RbX, examines the sensitivity of $\hat{f}$ in a large number of directions, while inheriting sparsity from the finite differences gradient method. It does so in a non-infinitesimal neighborhood of x_0, adapting the search space to cover the entire region the user deems locally relevant via the closeness region.

2.3 Baseline features

Many methods including IG and SHAP rely on the specification of a baseline feature combination x, such that the sum of all feature explanation scores at any target x_0 is equal to $f(x_0) - f(x)$. While Sundararajan et al. (2017) note that natural baselines exist in settings like image classification and sentiment analysis, for a general prediction or classification setting there may not be a canonical choice, and the feature attributions will be sensitive to the choice of baseline. SHAP's reliance on a baseline feature combination is eliminated by the cohort Shapley method of Mase et al. (2019), though cohort Shapley still retains an additivity constraint that all feature attributions must add up to $\hat{f}(x_0) - \bar{f}$, where $\bar{f}$ is the mean prediction on a set of n observations. If $\bar{f}$ is not a meaningful value then the individual feature scores do not have a direct interpretation.

Instead of requiring the user to pick a single "representative" combination of *features*, RbX asks for a region of prediction values on the *outcome* scale that are "close" to the prediction for the target point x_0. We believe this to be a simpler and more interpretable decision point in many settings, particularly when there are complex interactions between the features. Region-based explanations can also avoid using information from areas of feature space that are implausible (see Appendix A.2).

3 The RbX algorithm

We now describe the details of RbX. For ease of exposition we assume the closeness region is an interval $\mathcal{I} = [\hat{f}(x_0) - \epsilon_L, \hat{f}(x_0) + \epsilon_H]$, which depends on the user's choice of nonnegative parameters $\epsilon = (\epsilon_L, \epsilon_H)$. For example, if the prediction at the target point x_0 is $\hat{f}(x_0) = 13$, and the user thinks predictions between 10 and 20 should be close, then $\epsilon_L = 3$ and $\epsilon_H = 7$. We define

$\mathcal{E} = \{x \in \mathbb{R}^d \mid x \in \mathcal{I}\}$ to be the feature values for which the predictions are "ϵ-close" to $\hat{f}(x_0)$. Points outside $\mathcal{E}$ are said to be "ϵ-far." The RbX algorithm approximates $\mathcal{E}$ by a polytope $\mathcal{P}$.

Definition 1. *A polytope $\mathcal{P} \subset \mathbb{R}^d$ is any finite intersection of affine halfspaces, i.e.*

$$\mathcal{P} \equiv \cap_{1 \leq k \leq K} H_k,$$

where $H_k = \{x \in \mathbb{R}^d : x^T u_k \leq c_k\}$ is defined by its normal vector $u_k \in \mathbb{R}^d$ and intercept $c_k \in \mathbb{R}$.

The use of a polytope approximation, as opposed to a smooth shape like an ellipsoid, enables sparsity.

3.1 A POLYTOPE APPROXIMATION ALGORITHM

RbX is a greedy procedure that constructs the polytope approximation $\mathcal{P}$ of the ϵ-close region $\mathcal{E}$ one halfspace at a time (Algorithm 1). The algorithm requires a set of *context samples* $\mathcal{X} = \{x_i\}_{1 \leq i \leq n}$ that form the basis of the sampling procedure. These should be representative feature combinations, for instance from a possibly unlabeled training or validation set. To make the polytope $\mathcal{P}$ scale equivariant, all the features are scaled to have standard deviation 1 across the context samples. Only ϵ-far context points are used by the remainder of the procedure.

Algorithm 1 The RbX Algorithm

Input: target $x_0 \in \mathbb{R}^d$, closeness thresholds $\epsilon \succeq 0$, prediction model $\hat{f}$ with query access, context samples $\{x_i\}_{1 \leq i \leq n}$, maximum number of splits K.
Compute s, the vector of standard deviations of the d features across the context samples $\{x_i\}_{1 \leq i \leq n}$.
Standardize $x_i \leftarrow (s)^{-1} x_i$ for $i = 0, 1, \ldots, n$.
for $i \in [1 : n]$ if x_i is ϵ-far **do**
    Shrink x_i onto the ϵ-decision boundary using line search: $\tilde{x_i} \leftarrow$ Line-Search$(x_i, \hat{f}, \epsilon, x_0)$
end for
Initialize $\mathcal{R} \leftarrow \{\tilde{x_i}\}$
Initialize the set of support vectors $\mathcal{S} \leftarrow \emptyset$
Initialize $k \leftarrow 1$
while $\mathcal{R} \neq \emptyset$ and $k \leq K$ **do**
    Find $\tilde{x}^{(k)} \leftarrow \underset{\tilde{x} \in R}{\arg\min}\, \tilde{x} - x_{02}$
    Estimate gradient of $\hat{f}$ at $\tilde{x}^{(k)}$: $g_k \leftarrow$ Estimate-Grad$(\tilde{x}^{(k)}; \hat{f}, \delta, r, m)$
    Compute halfspace: $H_k \leftarrow \{x \in \mathbb{R}^d \mid x^T g_k \leq (\tilde{x}^{(k)})^T g_k\}$
    Update region: $\mathcal{R} \leftarrow \mathcal{R} \cap \text{int } H_k$
    Update support vectors: $\mathcal{S} \leftarrow \mathcal{S} \cup \{\tilde{x}^{(k)}\}$
    Increment k: $k \leftarrow k + 1$
end while
return $\{H_k\}_{1 \leq k \leq n}$, the collection of halfspaces defining the polytope $\mathcal{P}$
Notation: Line-Search is defined in Algorithm 3. Estimate-Grad is given in Algorithm 2.

The first step in the RbX algorithm is to shrink each (standardized and ϵ-far) context point along the line segment in $\mathbb{R}^d$ connecting it to x_0 to a point on the ϵ-boundary. This can be done quickly to exponential accuracy via a standard bisection-based line search (see Algorithm 3 in the Appendix for an example), noting that x_0 is always ϵ-close.

Next, RbX finds $x^{(1)}$, the context point whose shrunken counterpart $\tilde{x}^{(1)}$ is closest to x_0 in Euclidean distance. The first halfspace H_1 of the polytope $\mathcal{P}$ is chosen to pass through $\tilde{x}^{(1)}$ and have normal vector equal to an estimate of the gradient of $\hat{f}$, computed using finite differences (Algorithm 2) at $\tilde{x}^{(1)}$. This is motivated by the fact that in the case that $\hat{f}$ is differentiable, H_1 is a first-order approximation of a level set of $\hat{f}$. Finally, all shrunken context points outside the interior of H_1 are discarded, and the process is iterated with the remaining shrunken context points until either K halfspaces have been learned, or there are no more shrunken context points remaining.

Note that $\tilde{x}^{(k)}$, the shrunken context point chosen on the k-th iteration of the algorithm, lies outside the interior of the k-th halfspace H_k by construction. Thus the number of remaining shrunken context

points decreases by at least one after each iteration, and so the algorithm terminates in at most n iterations. In practice, far fewer iterations are often needed. To further reduce computation, one can impose early stopping by specifying a maximum number of halfspaces $K < n$ to be learned.

The "greedy" nature of the algorithm stems from choosing the closest shrunken context points first. This helps enforce a better approximation of the parts of the decision boundary closer to x_0, i.e. the "locally relevant" ones. It also tends to decrease the number of iterations needed before termination, as there will typically be more shrunken context points on the opposite side of each learned halfspace.

3.2 Finite differences gradient estimation and sparsity

The gradients of $\hat{f}$ at the shrunken context points are estimated using finite differences (Algorithm 2). We allow the user to average gradient estimates at m "jittered" points that are the original points corrupted by a small amount of Gaussian noise. This jittering is designed to smooth the gradient estimate when $\hat{f}$ has discontinuities on the ϵ-boundary. If $\hat{f}$ has no dependence on the i-th feature, it is clear that the i-th component of $\nabla \hat{f}(x)$ will be 0 according to Algorithm 2, regardless of the parameters used. Note this would not be the case if Algorithm 2 were replaced by a smooth gradient estimator, such as the Parzen windows used by Baehrens et al. (2010). Having zero gradient estimates for irrelevant features ensures that our local importance scores satisfy sparsity, as described in Section 4. All results in this paper are presented with parameters $r = 0.01$, $\delta = 0.1$, and $m = 10$.

Algorithm 2 Estimate-Grad Algorithm - Finite Differences

Input: point $x \in \mathbb{R}^d$, prediction model $\hat{f}$, step size δ, jitter radius r, number of jitter samples m
for $j \in [1 : m]$ **do**
  Generate $z \sim r \cdot \mathcal{N}(0, I_d)$
  $v \leftarrow x + z$
  $g_i^{(j)} \leftarrow \frac{\hat{f}(v+\delta e_i) - \hat{f}(v-\delta e_i)}{2\delta}$, $i = 1, \ldots, d$
end for
return $\nabla \hat{f}(x) = \frac{1}{m} \sum_{j=1}^{m} (g_1^{(j)}, ..., g_d^{(j)})$

3.3 Toy example

We briefly illustrate the RbX algorithm in Fig. 1, in a toy example with $d = 2$, $\hat{f}(x) = x_1 \cdot x_2$, $x_0 = (0, 0)$, and $\epsilon = (0.5, 0.5)$. 500 context points generated from a standard bivariate Gaussian distribution were used.

Each iteration of the algorithm approximates a plane tangent to the ϵ-boundary at the closest shrunken context point. All context points on the side of the plane not containing x_0 are then discarded. The resulting $\mathcal{P}$ at termination (after 4 splits) is a diamond-shaped region that truncates the true ϵ-close region, which is non-convex and extends infinitely along the coordinate axes in both directions.

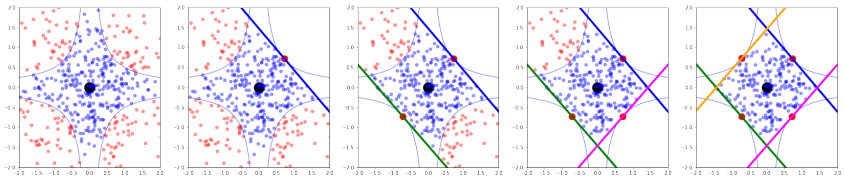

Figure 1: An illustration of the RbX algorithm for the pairwise interaction model $\hat{f}(x) = x_1 \cdot x_2$ with $\epsilon = (0.5, 0.5)$ and target point $x_0 = (0, 0)$, highlighted in the center of each plot. The smaller dots are the context points, colored by whether they are ϵ-far. Each panel shows the additional halfspace constructed in one iteration of the RbX algorithm. The lines are the halfspaces learned at each step of the algorithm; the larger dots along these lines are the shrunken context points used.

4 FROM POLYTOPES TO LOCAL FEATURE IMPORTANCE

Given the polytope $\mathcal{P}$ output by the RbX algorithm, we derive local prediction importance scores $S_j(\mathcal{P})$ using "feature escape distances" for each feature $j \in \{1, \ldots, d\}$:

$$S_j^+(\mathcal{P}) = \inf\{\alpha > 0 \mid x_0 + \alpha e_j \notin \mathcal{P}\}; \quad S_j^-(\mathcal{P}) = \inf\{\alpha > 0 \mid x_0 - \alpha e_j \notin \mathcal{P}\};$$
$$S_j(\mathcal{P}) = \min(S_j^+(\mathcal{P}), S_j^-(\mathcal{P})) \cdot \text{sign}(S_j^+(\mathcal{P}) - S_j^-(\mathcal{P}))$$

Here e_j is the j-th standard basis vector in $\mathbb{R}^d$. Then $S_j(\mathcal{P})$ is the minimum signed distance needed to escape the RbX polytope $\mathcal{P}$ by varying only the j-th feature from the target point x_0. If the corresponding "escape path" goes through regions of feature space deemed unlikely or untrustworthy, the S_j can be set to ∞ (Appendix A.2).

Note that the escape distances are reported on the original scales of each feature (before standardization), which enables them to be interpreted individually without reference to the escape distances of the other features. Alternatively, by reporting the escape distances on the standardized scale, we can sort them (from smallest to largest in absolute value) to obtain a ranking of the local importance of the features (from most important to least important).

If $\hat{f}$ doesn't depend on a feature x_j, then all of the halfspaces defining $\mathcal{P}$ will have normal vectors with 0 component in the x_j direction. This implies that the corresponding $S_j(\mathcal{P})$ will be ∞, and thus such features will have no importance, showing our procedure satisfies sparsity.

It is natural to compare the feature escape distances $S_j(\mathcal{P})$ with the "simple feature escape distances" $S_j(\mathcal{E})$, which use the original ϵ-close region $\mathcal{E}$ in place of the polytope $\mathcal{P}$. They can be computed via a line search similar to Algorithm 3, without running RbX. Clearly, a feature ranking based on the $S_j(\mathcal{E})$ would also satisfy sparsity. Furthermore, it would have better detection power than a gradient-based method. This is because the $S_j(\mathcal{E})$ look beyond an infinitesimal neighborhood of x_0, instead focusing on a typically larger region defined in terms of the prediction values of $\hat{f}$ to be meaningful. However, it still cannot capture changes in $\hat{f}$ near x_0 that cannot be detected when only one feature is changed at a time from its value at x_0.

Since the polytope $\mathcal{P}$ looks in many directions around x_0, using the RbX distances $S_j(\mathcal{P})$ provides better detection power. A simple example of this can be seen in Fig. 1. There we have $\tilde{S}_1 = \tilde{S}_2 = \infty$ but $S_1 \approx S_2 \approx 1.4$. This is illustrated further by the data and experiments in the next section.

5 DATA EXAMPLE AND SYNTHETIC EXPERIMENTS

We now compare RbX to existing methods for local prediction importance on a credit scoring example alluded to in the introduction, along with simulated experiments.

5.1 CREDIT SCORING

The home equity line of credit (HELOC) dataset from the FICO xML Challenge (`community.fico.com/s/xml`) contains the `RiskPerformance` of 2,502 credit applicants. This is a binary indicator of whether they were ever more than 90 days past due in the first two years after account opening. The goal is to interpretably classify each individual as having either "Bad" or "Good" `RiskPerformance` based on 23 predictors that are all either quantitative or ordered categorical. We consider the local prediction importance of a shallow and sparse decision tree. The scalar prediction output is taken to be the predicted probability of a "Good" `RiskPerformance`.

We split the dataset randomly into 1,751 training observations and 751 test observations. The tree classifier is fit to the training observations and has depth 3. A visualization of the classifier is in Fig. 2. Without hyperparameter tuning, it achieves an out-of-sample misclassification rate of around 30%, compared to 26% for the state of the art (Chen et al., 2018a).

We focus on a randomly chosen target point, corresponding to the individual labeled 5,238 in the dataset. Selected feature values for that individual (who has prediction value 0.191) are given in Table 1, along with feature scores from various local importance methods. For LIME and SHAP, we compute the scores from the open-source implementations and default settings, except that for LIME

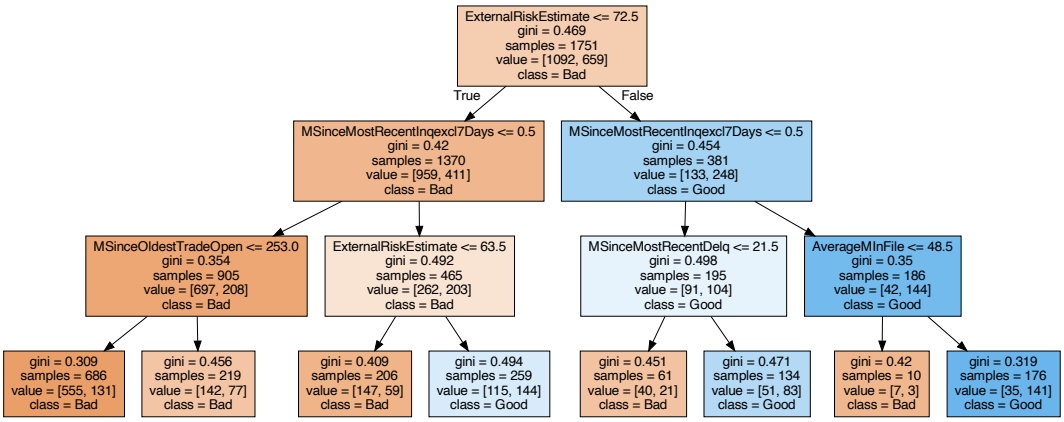

Figure 2: A visualization of a classification tree fit to the HELOC data from the FICO xML Challenge.

Table 1: Feature values and importance scores for a sample target point based on $\hat{f}$ as in Fig. 2

Feature	Feature value	S_j	S_j norm.	LIME	SHAP 1	SHAP 2
`ExternalRiskEstimate`	61.0	2.5	0.32	0.12	-0.13	-0.13
`MSinceOldestTradeOpen`	149.0	∞	∞	0.018	0	-0.08
`MSinceMostRecentInqexcl7Days`	0.0	0.58	0.12	0.084	-0.23	-0.15
`MSinceMostRecentDelq`	3.0	∞	∞	0.0067	0	0
`AverageMInFile`	49.0	∞	∞	0.016	0	0
`NumTrades60Ever2DerogPubRec`	1	∞	∞	0.0011	0	0

we do not discretize the features, which greatly improves its performance. For RbX we assume a decision boundary of $\{x \mid \hat{f}(x) = 0.5\}$, and take $\mathcal{E}$ to be the set of points on the same side of this decision boundary as x_0. All training observations are used as context points.

The RbX algorithm terminates in 6 iterations. We obtain escape distances (on the original scale of the features) of 2.5 for `ExternalRiskEstimate` and 0.58 for `MSinceMostRecentInqexcl7Days`. All other feature escape distances are infinite. An intepretation is that increasing `ExternalRiskEstimate` by 2.5 and `MSinceMostRecentInqexcl7Days` by 0.58 would change the classification of the target individual under the tree classifier to `Good`.

By contrast, simple feature escape distances are infinite for *every* feature, as changing any single feature in the target point — and keeping all others fixed — cannot change the target individual's classification from Bad to Good under the tree model. Once again, this shows the increased detection power of the feature escape distances, compared to the simple feature escape distances.

Table 1 also shows the sensitivity of SHAP explanations to the choice of baseline. For two different baseline choices that are in the same leaf of the tree classifier, SHAP gives different explanations. One of these baselines yields nonzero SHAP value for `MSinceOldestTradeOpen`, yet using the other baseline gives zero SHAP for that feature (Table 1). Finally, the LIME explanations for this example seem to agree with RbX and SHAP, except LIME assigns nonzero attribution scores to all six features that appear in the tree classifier. Yet out of 1,000 LIME repetitions with random samples for fitting the surrogate linear model (one of which was randomly chosen to fill Table 1), 46 of them yielded a higher attribution score to a feature that doesn't appear at all in the classifier than to one of the six features that do, demonstrating the failure of sparsity.

In Appendix A.3 we also examine local prediction attribution scores for a gradient boosted tree ensemble. Since it is difficult to visualize this classifier, and it is not sparse, there is not a clear ground truth we can use to evaluate the different methods' explanations. However, we do note that as for the decision tree, all the simple feature escape distances are infinite, so that method provides no information. We provide a more systematic comparison of the different approaches in the synthetic experiments of the next section.

5.2 SYNTHETIC EXPERIMENTS

To better illustrate the detection power of different local prediction importance methods, we carry out a modification of the experiments of Chen et al. (2018b), with four data generating scenarios having sparse signals. As explained in Section 2.2, we replace the dense neural network used there by two fully sparse prediction models. In particular we evaluate performance on both the Bayes prediction model (smooth) and a K-nearest neighbors (KNN) regressor (nonsmooth).

The precise data scenarios are described as follows:

1. Generate $(X_1, \ldots, X_9)$ from a spherical standard Gaussian distribution. Generate X_{10} independently from an equally weighted mixture of two Gaussian distributions with standard deviation 1, centered at $+3$ and -3.
2. Let $X = (X_1, \ldots, X_{10})$. Then an outcome Y is generated as follows:
   - <u>XOR:</u> $\mathbb{E}(Y \mid X = x) = (1 + x_1 x_2)^{-1} := p_X(x_1, x_2)$
   - <u>Orange skin:</u> $\mathbb{E}(Y \mid X = x) = \left(1 + \exp\left(\sum_{i=1}^4 x_i^2 - 4\right)\right)^{-1} := p_O(x_1, x_2, x_3, x_4)$
   - <u>Nonlinear additive:</u> $\mathbb{E}(Y \mid X = x) = (1 + \exp(-100 \sin(2x_1) + 2|x_2| + x_3 + \exp(-x_4)))^{-1} := p_N(x_1, x_2, x_3, x_4)$
   - <u>Feature switching:</u> $\mathbb{E}(Y \mid X = x) = p_O(x_1, \ldots, x_4) r(x_{10}) + p_N(x_5, \ldots, x_8)(1 - r(x_{10}))$

The feature switching scenario represents the setting where Y is drawn from the orange skin model using features $X_1, \ldots, X_4$ whenever X_{10} is from the component with center $+3$. Otherwise, Y is drawn from the nonlinear additive model with features $X_5, \ldots, X_8$. Then $r(x_{10})$ is the posterior probability that X_{10} was drawn from the component with center $+3$, given $X_{10} = x_{10}$.

As in Chen et al. (2018b), for the feature switching case we deem features 5-9 are locally relevant if x_{10} is nonnegative; otherwise features 1-4 and 9 are locally relevant. In the other scenarios, the globally relevant features are the locally relevant ones for all target points.

For each method we select the M locally most important features, where M is the true number of locally relevant features (2 for XOR, 5 for switch, 4 for the other two). Ties in importance scores are broken randomly but features with no importance are never selected. We evaluate the ability of RbX feature importance scores to recover the locally relevant features under each scenario, compared with popular methods with publicly available (or trivial) implementations: simple feature importance (SFI), a gradient method using Algorithm 2 ("Gradient"), LIME, and SHAP. Of these approaches, only LIME does not satisfy sparsity.

We evaluate performances for each prediction model, scenario, and method on the same 1,000 randomly generated target points. The KNN models are fit with $K = 5$ using 1,000 independent training points and only look at the locally relevant features (to ensure they are sparse). For RbX we use 1,000 independent context points. The ϵ-close region for each target point consists of all points on the same side of the decision boundary $\{x \mid \hat{f}(x) = 0.5\}$. For SHAP the baseline point is the origin.

The results are shown in Fig. 3. For all scenarios, RbX shows the highest recall among the methods considered, with perfect performance on both classifiers for all scenarios besides feature switching. The gradient-based and simple feature importance methods particularly struggle with the nonlinear additive model since there are regions where the classifiers are locally flat — for instance, wherever $\sin(2X_1)$ is sufficiently greater than zero so that the predictions in a neighborhood are all numerically equivalent to 1. RbX overcomes this by examining a non-infinitesimal neighborhood around x_0.

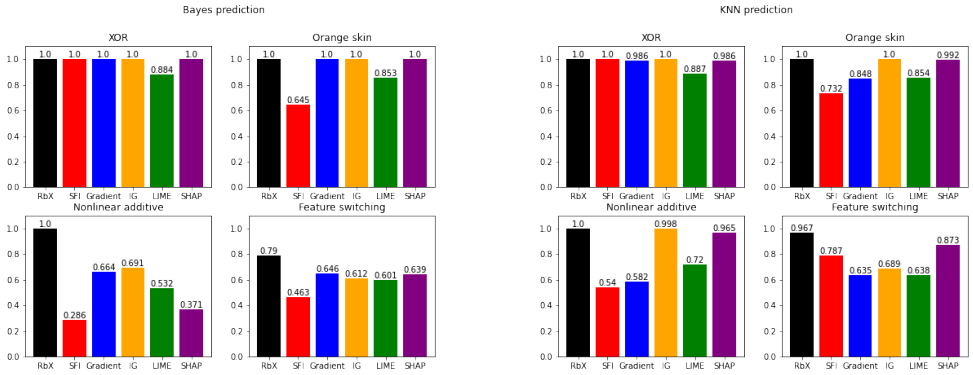

Figure 3: (Left) Recovery rates, based on 1,000 random target points, of the locally relevant features for various local prediction importance methods, using the smooth, sparse Bayes prediction model for each of the 4 simulated scenarios described in the main text. (Right) Same as the left panel, but for the nonsmooth KNN prediction model.

6 Summary

We have proposed region-based explanations (RbX) as a novel approach for generating instancewise explanations of black-box prediction models. The method is agnostic to the inner workings of the prediction model, only requiring query access to it. The main idea behind RbX is quite simple — it successively refines polytope approximations to a region of feature space with similar predictions to the target point. The user can directly specify what prediction values are “similar" based on the context in which the prediction model is being used. By contrast, existing methods specify a “locally relevant" region in terms of the values of the *features*. When the number of features is moderate and there are interactions between them, specifying such a region becomes difficult.

Unlike some other widely used methods, RbX is guaranteed to preserve sparsity, meaning that features which are completely irrelevant for a prediction model are always assigned zero importance. At the same time, our data examples and simulation results suggest RbX has a strong ability to detect locally relevant features, particularly in sparse prediction models.

Further work in the direction of region-based explanations might leverage more theoretical work on polytopic approximation to develop mathematical guarantees about the output region (Bronstein, 2008; Arya et al., 2012). There may also be other ways in which the polytope constructed by Algorithm 1 or a variant thereof could be useful for local prediction importance, beyond the “escape distances" of Section 4. Finally, we believe it could be informative to formalize the notion of detection power and develop principled ways of maximizing it subject to sparsity.

References

Sunil Arya, Guilherme D Da Fonseca, and David M Mount. Polytope approximation and the mahler volume. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pp. 29–42. SIAM, 2012.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *The Journal of Machine Learning Research*, 11:1803–1831, 2010.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Efim M Bronstein. Approximation of convex sets by polytopes. *Journal of Mathematical Sciences*, 153(6):727–762, 2008.

Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. Visualizing the feature importance for black box models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 655–670. Springer, 2018.

Chaofan Chen, Kangcheng Lin, Cynthia Rudin, Yaron Shaposhnik, Sijia Wang, and Tong Wang. An interpretable model with globally consistent explanations for credit risk. *arXiv preprint arXiv:1811.12615*, 2018a.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pp. 883–892. PMLR, 2018b.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794, 2016.

Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *J. Mach. Learn. Res.*, 20(177):1–81, 2019.

Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, 19(1):1–67, 1991.

David Harrison Jr and Daniel L Rubinfeld. Hedonic housing prices and the demand for clean air. *Journal of environmental economics and management*, 5(1):81–102, 1978.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Ismael Lemhadri, Feng Ruan, and Rob Tibshirani. Lassonet: Neural networks with feature sparsity. In *International Conference on Artificial Intelligence and Statistics*, pp. 10–18. PMLR, 2021.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

Masayoshi Mase, Art B Owen, and Benjamin Seiler. Explaining black box decisions by shapley cohort refinement. *arXiv preprint arXiv:1911.00467*, 2019.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, pp. 3145–3153. PMLR, 2017.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pp. 3319–3328. PMLR, 2017.

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

A Appendix

A.1 Line search algorithm

Here we provide a sample line search algorithm for shrinking the context points given to the RbX algorithm to the ϵ-boundary.

Algorithm 3 Line-Search Algorithm

Input: ϵ-far point $x \in \mathbb{R}$ to be shrunk, thresholds $\epsilon \succeq 0$, prediction model $\hat{f}$, target $x_0 \in \mathbb{R}^d$, and maximum number of iterations M.
Set $t_H \leftarrow 1$, $t_L \leftarrow 0$.
for iter $\in [1 : M]$ **do**
    Compute $t_M \leftarrow \frac{1}{2}(t_H + t_L)$
    if $x_0 + t_M(x - x_0) \in \mathcal{E}$ **then**
        $t_L \leftarrow \frac{1}{2}(t_H + t_L)$
    else
        $t_H \leftarrow \frac{1}{2}(t_H + t_L)$
    end if
end for
return $\frac{1}{2}(t_H + t_L)$

A.2 Trustworthy regions

As discussed by Mase et al. (2019), a pitfall of baseline methods such as LIME, Kernel SHAP, and IG is that they often rely on predictions at implausible combinations of feature values, such as a graduation date before a birth date, due to interactions between features. Local prediction explanations utilizing such information have questionable fidelity.

To prevent the feature escape distances $S_j(\mathcal{P})$ from using information about the classifier near implausible feature combinations, we can establish a "trustworthy region" $\mathcal{T}$ containing x_0 that corresponds to the set of plausible feature values. Then if $S_j^+(\mathcal{T}) < S_j^+(\mathcal{P})$ — meaning that in order to escape $\mathcal{P}$ by increasing the j-th feature from x_0, we must leave the trustworthy region $\mathcal{T}$ — we set $S_j^+(\mathcal{P}) = \infty$. We do the same thing for the $S_j^-(\mathcal{P})$. We could also make the simple feature escape distances $S_j(\mathcal{E})$ more trustworthy in the same way.

In some settings, domain knowledge informs a reasonable choice for $\mathcal{T}$. Otherwise, there are many plausible ways to define $\mathcal{T}$, assuming access to a large collection of plausible feature combinations, such as the context points for the RbX algorithm. One such method is given in Appendix 5 of Mase et al. (2019). Another, based on Section 14.2.4 of Hastie et al. (2009), would be to estimate $r(x) = \frac{g(x)}{g_0(x)}$ where $g(\cdot)$ is viewed as an unknown joint density of the data generating process for the context points, and $g_0(\cdot)$ is a known baseline density that is positive at each context point, e.g. uniform over a rectangular region containing the context points. This function $r(x)$ can be estimated by any binary classification procedure that outputs class probabilities. By generating a large number of i.i.d. points from $g_0(\cdot)$, we can learn the probability that a point at x came from g rather than g_0, using both the original context points and the feature combinations generated from $g_0(\cdot)$. From this, an estimate $\hat{r}(x)$ of $r(x)$ follows via Bayes' rule. Then we define $\mathcal{T} = \{x \mid \hat{r}(x) \geq \beta\}$ for some trustworthiness threshold $\beta > 0$.

We illustrate a trustworthy region using a simple quadratic logistic regression classifier fit to the popular Boston housing dataset (Harrison Jr & Rubinfeld, 1978). The goal, as in Mase et al. (2019), is to predict whether median neighborhood value is less than $20,000 based on two features — `CRIM` (crime rate) and `RM` (median number of rooms per home). We use the same target point as Mase et al. (2019), marked with a red X in Fig. 4.

The trustworthiness classifier was fit with multivariate adaptive regression splines (Friedman, 1991) using a logistic link, allowing for order-two interactions. The contours of this classifier are shown in Fig. 4. If $\mathcal{T}$ is defined with any trustworthiness threshold β larger than about 0.15, the escape distance in the positive `CRIM` direction is set to ∞ because then it is impossible to escape the RbX polytope without leaving $\mathcal{T}$. As Mase et al. (2019) suggest, this is desirable since there are no context points with similar `RM` values in the ϵ-far region but high values of `CRIM`.

A.3 Results for boosted tree ensemble on FICO dataset

We replicate the results given in Table 1, but replacing the tree classifier in Fig. 2 with a gradient boosted ensemble fit using XGBoost (Chen & Guestrin, 2016) on the same training dataset. The

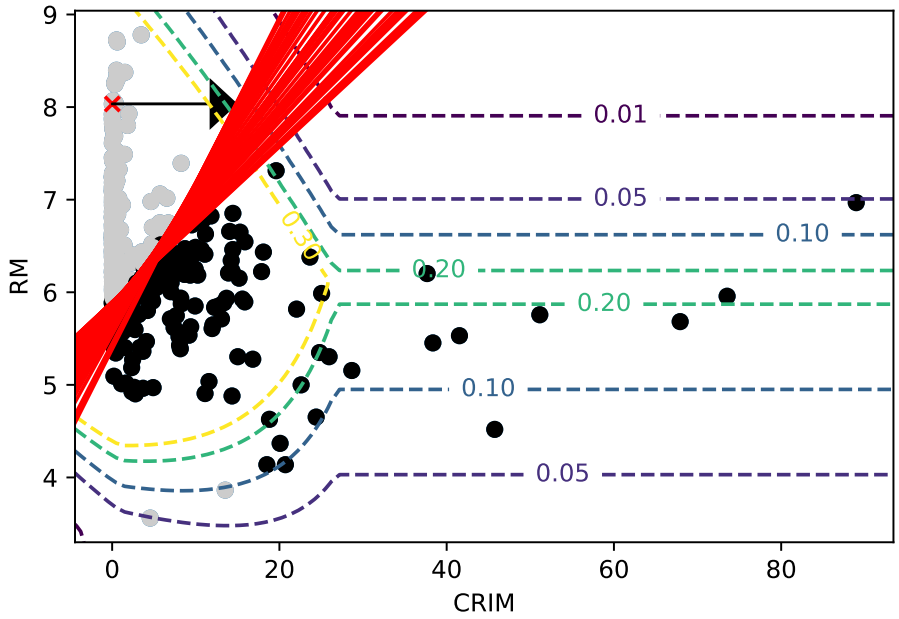

Figure 4: An illustration of the trustworthiness region classifier for the Boston housing data example described in the text. Its contours are indicated by the dashed lines. The solid lines indicate the halfspaces defining the polytope $\mathcal{P}$ from the RbX algorithm for the target point x_0, denoted with a red X. The arrow indicates the "polytope escape path" from x_0 in the `CRIM` direction. For any β such that the arrow crosses the β-contour (or smaller) of the trustworthiness region classifier, we set $S^+_{\texttt{CRIM}}(\mathcal{P})$ to ∞. The dots are candidate context points from the entire dataset, colored by whether they are ϵ-far.

training loss is the negative logistic log likelihood with early stopping after 10 iterations (and otherwise up to 1,000 boosting rounds).

Table 2: Same as Table 1, but for the boosted tree ensemble classifier

Feature	Feature value	S_j	S_j norm.	LIME	SHAP 1	SHAP 2
`ExternalRiskEstimate`	61.0	-4.95	-0.64	-0.10	-0.17	-0.24
`MSinceOldestTradeOpen`	149.0	-15.7	-0.17	-0.045	-0.12	0
`MSinceMostRecentInqexcl7Days`	0.0	1.72	0.40	-0.10	-0.32	-0.18
`MSinceMostRecentDelq`	3.0	8.2	0.39	-0.06	-0.026	0
`AverageMInFile`	49.0	-11.5	-0.43	-0.05	0	0
`NumTrades60Ever2DerogPubRec`	1	1.63	1.11	0.015	0.024	0.031