How Well Do Feature-Additive Explainers Explain Feature-Additive Predictors?

Zachariah Carmichael* **Walter J. Scheirer**
Department of Computer Science & Engineering
University of Notre Dame
{zcarmich,walter.scheirer}@nd.edu

Abstract

Surging interest in deep learning from high-stakes domains has precipitated concern over the inscrutable nature of black box neural networks. Explainable AI (XAI) research has led to an abundance of explanation algorithms for these black boxes. Such post hoc explainers produce human-comprehensible explanations, however, their fidelity with respect to the model is not well understood – explanation evaluation remains one of the most challenging issues in XAI. In this paper, we ask a targeted but important question: can popular feature-additive explainers (e.g., `LIME`, `SHAP`, `SHAPR`, `MAPLE`, and `PDP`) explain feature-additive predictors? Herein, we evaluate such explainers on ground truth that is analytically derived from the additive structure of a model. We demonstrate the efficacy of our approach in understanding these explainers applied to symbolic expressions, neural networks, and generalized additive models on thousands of synthetic and several real-world tasks. Our results suggest that all explainers eventually fail to correctly attribute the importance of features, especially when a decision-making process involves feature interactions.

1 Introduction

The counterintuitive mispredictions and undesirable behaviors of black box AI systems [99, 116, 54, 117, 77] has piqued widespread interest in explainable AI (XAI) solutions, including from the medical, financial, and the legal domains [77, 83, 105, 118]. Late interest has saturated due to real-world consequences [90, 20, 16, 81] and regulatory pushes [36, 120, 37, 76, 29]. Accordingly, a plethora of XAI approaches have been proposed to shed light on previously inscrutable black boxes. However, explanations are notoriously difficult to evaluate [34]. Measuring the fidelity of explanations has remained so unverifiable [15] that we are starting to see meta-evaluations (quality evaluations of quality evaluation metrics of explanations of black box models) [53].

In this work, we study the evaluation of a specific but popular class of XAI methods: post hoc feature-additive explainers, like `LIME` [103], `SHAP` [79], and `PDP` [41]. We ask, "can feature-additive explainers explain feature-additive predictors?" We propose a novel explainer evaluation methodology that overcomes many issues present in prior work. This work presents the following contributions:

- We construct a test bed for the evaluation of feature-additive post hoc explanations against *ground truth* derived analytically from feature-additive models. By definition, perfect explanations should be *exactly equal* to this ground truth.
- To facilitate evaluation, we propose an algorithm, MatchEffects, that directly maps any model with any amount of additive structure to feature-additive post hoc explanations.

*Code available at github.com/craymichael/PostHocExplainerEvaluation

XAI in Action: Past, Present, and Future Applications @ NeurIPS 2023.

- We evaluate the popular post hoc explainers LIME [103], SHAP [79], SHAPR [1], PDP [41], and MAPLE [95] on thousands of synthetic tasks and models, as well as with neural networks and generalized additive models on several real-world datasets.
- We demonstrate that although SHAP outperforms the other explainers, all explainers begin to fail in the presence of higher-dimensional data, models with higher-order interactions, and models with more interaction effects.

2 Background

Algorithms for Local Post Hoc Explanation Whereas *ante hoc* explainers have an intrinsic notion of interpretability, post hoc methods serve as a surrogate explainer for a black box. There are several classes of post hoc explanation methods, including salience maps [11, 110], local surrogate models [79, 103], counterfactuals [128], and global interpretation techniques [132]. A comprehensive overview can be found in [48, 85, 122, 108, 8]. However, here we strictly focus on feature-additive local approximation [21], which is one of the most prevalent explanation strategies. Local post hoc explainers estimate the feature importance for a single decision whereas global explainers provide explanations of a model for an entire dataset. Explainers aim to recover the local model response about an instance while isolating the most important features to produce comprehensible explanations. Specifically, we consider the LIME [103], SHAP [79], SHAPR [1], PDP [41], and MAPLE [95] explainers. The explainers are detailed in Appendix A.

Evaluation of Explainers There are three main types of evaluations: application-grounded (real humans, real tasks), human-grounded (real humans, simplified tasks), and functionally-grounded (no humans, proxy tasks) [34]. Human- and application-grounded evaluations are expensive, subjective, and qualitative. However, they measure the human utility and effectiveness of explanations. Functionally-grounded metrics are concerned with proxies for the same objectives, but also can quantitatively score the fidelity of an explanation with respect to the model being explained [89]. We are interested in the *functionally-grounded* evaluation of explanation *fidelity* (correctness) in this paper. The highest-fidelity evaluations involve comparing explanations to the ground truth explanation. For the sake of space, we abbreviate related work and elaborate in Appendix A. In short, there are three types of ground truth checks that can be performed: against annotations, controlled data, and white boxes [7, 89] – we are interested in the latter as it offers the highest-fidelity evaluation of an explainer.

White box checks evaluate the correspondence between explanations and the known white box reasoning. There are several types of evaluations:

- *Feature selection* approaches isolate a subset of features that the model uses, e.g., by having the model use a subset of features globally, only considering the features leading to the predicted leaf of a tree, or only considering the features that a rule comprises [19, 62, 61, 133].
- *Feature ranking* approaches identify the relative importance of each feature so they can be ranked, such as via the coefficients of a logistic regression model [45, 103, 136].
- *Inexact feature contributions* methods use a proxy measure to estimate feature contributions from a model, such as the gradient with respect to each feature in differentiable models or the Gini impurity of trees [65, 47, 88].
- *Exact feature contributions* methods identify the exact amount each feature contributes to the predicted outcome of a model with respect to a formal (and useful) definition of contribution. Typically, and in this paper, a *contribution* is the amount that a feature (or subset of features) at a particular value adds to the predicted outcome such that the sum of all contributions totals the model prediction. For instance, the prediction could be the number of times a non-overlapping pattern appears in an image, thus the additive contribution of each pixel is known [84]. Alternatively, the contributions can be taken from a linear regression model [28, 72]. In [18], a connection it is shown that GAMs (with or without interaction effects) can be recovered from Shapley values (with or without interaction effects). They demonstrate that interaction effects with an order of two can be precisely estimated, but can only be detected at higher orders with Shapley values in experiments.

White box checks offer the highest fidelity estimate of explanation fidelity as the form of the model, and thus how it uses the data, is well-understood. However, the feature selection and ranking approaches are limited in that the contribution of each feature is unknown. Exact feature

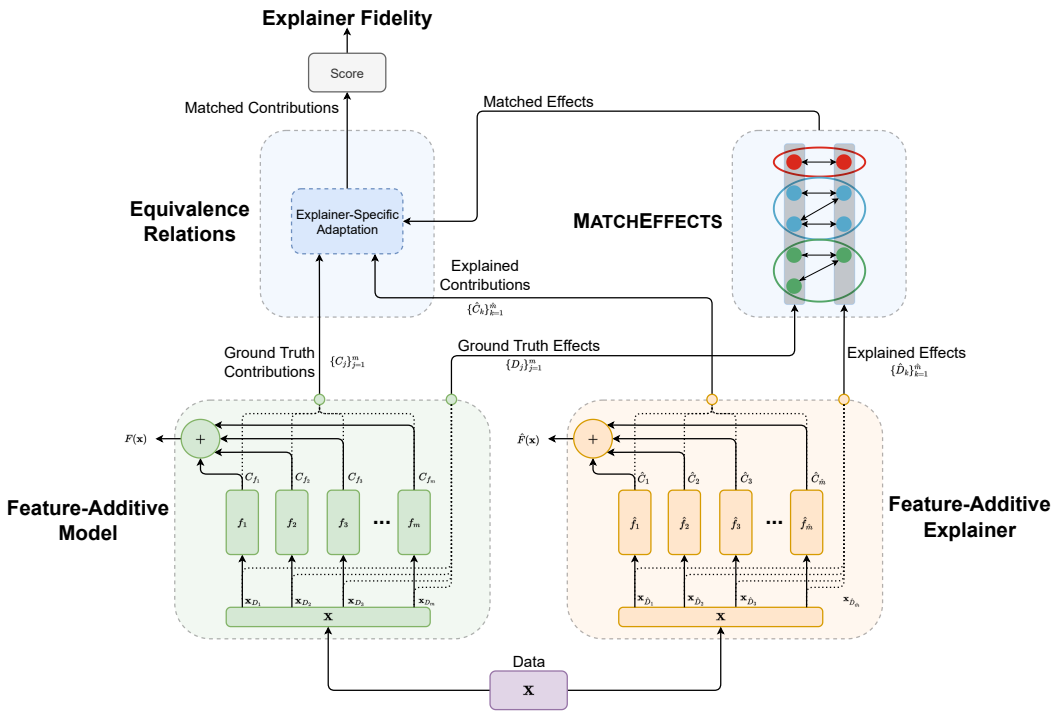

Figure 1: High-level overview of the proposed evaluation of post hoc explainer quality. Therein, a feature-additive post hoc explainer estimates the feature contributions of a feature-additive model for some data $\mathbf{X}$. Each f_i and $\hat{f}_j$ is a function as described in Section 3. Both the model and the explainer produce a set of effects and contributions – those of the model are the ground truth. To compare the model and explainer, the effects of the explainer are aligned with the ground truth using the MATCHEFFECTS algorithm (Section 3). Thereafter, the matched effects inform how to carry out the equivalence relations – this process allows for direct comparison of explained contributions to the ground truth (Section 3). Finally, the fidelity of the explainer is computed using the matched contributions – perfect explainer explanations should be *exactly equal* to the ground truth explanations.

contribution approaches are also able to evaluate both feature selection and ranking. In addition, they are more faithful to the true feature contributions within the explained model than inexact feature contributions as no proxy is needed. Our approach fits into this category of white box checks.

Our approach has several advantages over prior work. (1) [28, 72] define the exact feature contributions as the coefficients of a linear regression model to evaluate the fidelity of explainers, such as `LIME` and `SHAP`. However, this is inappropriate for these explainers and thus does not provide an exact set of feature contributions (see Appendix A for details). We correct for these issues in our work for all considered explainers. (2) No prior work on white box checks has considered the case of feature interactions [84, 28, 72], which are ubiquitous among black box models, especially neural networks. We consider models with a various number of feature interactions and order of feature interactions. (3) Unlike prior work [84, 28, 72], we consider in our experiments both synthetic and real-world data, both tabular and image data, as well as both non-learnable and learnable models, including convolutional neural networks.

Further comprehensive overviews of explanation evaluation methodologies and aspects are detailed in [7, 89, 134, 123, 92, 101] (as well as Appendix A).

3 Methodology

We propose to evaluate feature-additive explainers by comparing their explanations to the ground truth explanations from feature-additive white box models. We provide a fair way of comparing

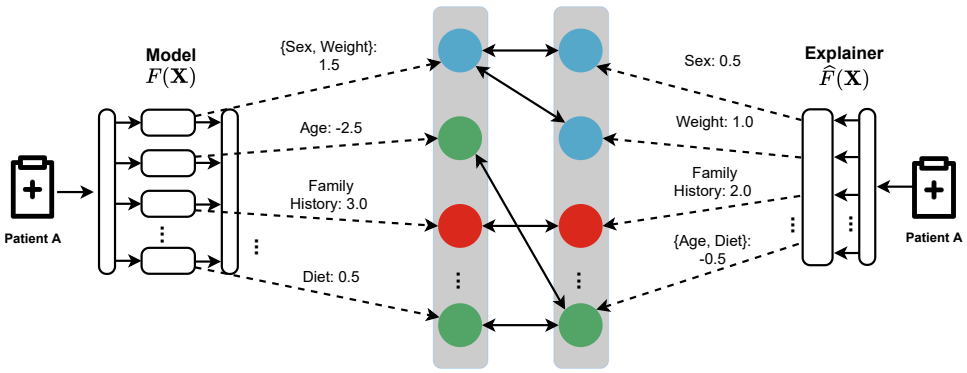

Figure 2: A simple example demonstrating MatchEffects for a hypothetical medical task. Like colors from each side of the graph can be directly compared. While no explainer evaluated in this work explicitly detects interactions ($|\hat{m}| \geq 2$), the visual demonstrates how such explainers are compatible with the framework.

these two explanations (sets of feature contributions) when the model has feature interactions, or, more generally, when the explainer and model explanations comprise different sets of effects. Figure 1 shows a high-level overview of our approach.

Feature-Additive Model & Explainer Formulation We consider a general form of feature-additive white box models, similar to, but distinct from, generalized additive models (GAMs)[2]. Concise definitions of feature contributions follow naturally from its additive structure while still allowing for feature interactions, high dimensionality, and highly nonlinear effects. Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ be a matrix with n samples and d features, $\mathbf{x} \in \mathbf{X}$ be a sample, $D = \{i\}_{i=1}^{d}$ be the set of all feature indices, and $F(\cdot)$ be an additive function comprising m effects. Each effect is given by a non-additive function $f_j(\cdot)$ that takes a subset of features $D_j \subseteq D$ as input and yields an additive contribution C_j to the model output as shown in Eq. (1). In this paper, we refer to an *effect* by the subset of features D_j that it comprises. If $|D_j| > 1$, it is an interaction effect, otherwise it is a main effect. The ground truth explanation is then the set of effects and their contributions $\{(D_j, C_j)\}_{j=1}^{m}$.

$$F(\mathbf{x}) = \sum_{j=1}^{m} f_j\left(\mathbf{x}_{D_j}\right) = \sum_{j=1}^{m} C_j \quad (1)$$

$$\hat{F}(\mathbf{x}) = \sum_{k=1}^{\hat{m}} \hat{f}_k\left(\mathbf{x}_{\hat{D}_k}\right) = \sum_{k=1}^{\hat{m}} \hat{C}_k \quad (2)$$

Explainer For explainers, we denote local estimates of the model as $\hat{F}(\cdot)$ which comprise a summation of $\hat{m}$ effects given by each $\hat{f}_k(\cdot)$ as in Eq. (2). Similarly, the explanation from an explainer has the form $\{(\hat{D}_k, \hat{C}_k)\}_{k=1}^{\hat{m}}$. The explainers evaluated in this work have $\hat{m} \leq d$, however, explainers with $\hat{m} > d$ are compatible with our formulation and implementation.

Synthetic Models We generate synthetic models with controlled degrees of sparsity, order of interaction, nonlinearity, and size. This allows us to study how different model characteristics affect explanation quality. Here, each $f_j(\cdot)$ is a composition of random non-additive unary and/or binary operators for a random subset of features D_j. Expressions are generated based on these parameters and we verify that the domains and ranges are in $\mathbb{R}$. For example, a generated expression with $m = d = 4$ and 2 dummy features could look like $F(\mathbf{x}) = \mathbf{x}_1 + e^{\mathbf{x}_4} + \log(\mathbf{x}_1 \mathbf{x}_4) + \frac{\mathbf{x}_4}{\mathbf{x}_1}$. See Appendix E for details of our algorithm used to generate such models.

Learned Models We consider two types of learned models: GAMs and feature-additive neural networks (NNs). The former is a rich yet simple model that models nonlinear effects while being conducive for understanding feature significance [51]. Each $f_j(\cdot)$ is a smooth nonparametric function that is fit using splines. A link function relates the summation of each $f_j(\cdot)$ to the target response, such as the identity link for regression and the logit link for classification.

[2]This formulation notably differs from GAMs in that each $f_j(\cdot)$ can be non-smooth.

The feature-additive NNs we consider have the same additive structure, but each $f_j(\cdot)$ is instead a fully-connected NN. Each NN can have any architecture, operates on D_j, and yields a scalar value for regression or a vector for classification. The output is the summation of each NN with a link function similar to the GAM. This structure is related to the neural additive model proposed in [6]. This NN formulation also holds for convolutional NNs (CNNs), which can have a non-unary m as long as the receptive field at any layer does not cover the full image. Notably, while the CNN operates on the image data $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n \times d_1 \times d_2 \times d_3}$, the explainers that we consider operate on the flattened data $\mathbf{X} \in \mathbb{R}^{n \times d_1 d_2 d_3}$. See Appendix B for more details.

The number of effects m and each effect D_j are selected randomly for learned models such that $m > 1$, the number of matches from MatchEffects (introduced in Section 3) is >1, and the task error is satisfactorily low.

Ground Truth Alignment: MatchEffects With our formalism, we now have a model and an explainer, each of which produces explanations as a set of effects and their corresponding contributions. Because there may not be a one-to-one correspondence between the two sets, we cannot directly compare the effects. Consider the case of a model with an interaction effect, *i.e.*, some $|D_j| \geq 2$; if the explanation has no $\hat{D}_k = D_j$, then a direct comparison of explanations is not possible. To this end, we propose the MatchEffects algorithm, which matches subsets of effects between the model and explainer. Put simply, the algorithm finds the smallest feature interaction effects that are common between the model and explainer explanations. For example, if an explainer explanation contains contributions for features 1 and 2, and the model ground truth explanation contains a contribution for the interaction effect involving both 1 and 2, then the sum of the explainer contributions for these features is compared to the ground truth contribution. A visual example of MatchEffects is given in Figure 2.

To achieve this matching, we consider all D_j and $\hat{D}_{f_k}$ to be the left- and right-hand vertices, respectively, of an undirected bipartite graph. Edges are added between effects with common features. We then find the connected components of this graph to identify groups of effects with inter-effect dependencies. If every component contains an exact match, for example, if $\text{match}_F = \{\{2\}, \{2,3\}\}$ and $\text{match}_{\hat{F}} = \{\{2\}, \{2,3\}\}$, then each contribution by $\{2\}$ and $\{2,3\}$ will be compared separately. Further details and algorithm illustrations are provided in Appendix A.

Equivalence Relations to Explainers With MatchEffects and MaloU defined, a direct comparison between true and explained explanations is nearly possible. However, some adaptation is still required due to the use of normalization and differing definitions of "contribution" between explainers. Here, we bridge together these definitions. `LIME` normalizes the data as z-scores, *i.e.*, $z = (x_i - \mu_i)/\sigma_i$, before learning a linear model. We then need to scale the coefficients $\Theta = \{\theta_i\}_{i=1}^{d}$ of each local linear model using the estimated means μ_i and standard deviations σ_i from the data as in Eqs. (3) and (4). In `SHAP`, the notion of feature importance is the approximation of the mean-centered independent feature contributions for an instance. The expected value $\mathbb{E}[F(\mathbf{x})]$ is estimated from the background data `SHAP` receives. In order to allow for valid comparison, we add back the expected value of the true contribution $\mathbb{E}[C_i]$ estimated from the same data. However, since a 1:1 matching is not a guarantee, we must consider all effects grouped by said matching as in Eq. (5). The same procedure applies to `SHAPR`. See Appendix C for the derivations of these relations. Furthermore, `LIME` and `MAPLE` provide feature-wise explanations as the coefficients Θ of a linear regression model. In turn, we must simply compute the product between each coefficient and feature vector $\mathbf{x}_i \theta_i$ to yield the contribution to the output according to the explainer.

$$\theta_0' = \theta_0 - \sum_i \frac{\mu_i \theta_i}{\sigma_i} \tag{3}$$

$$\theta_i' = \frac{\theta_i}{\sigma_i} \tag{4}$$

$$C_{\text{match}_{\hat{F}}} = \sum_{k \in \text{match}_{\hat{F}}} \hat{f}_k(\mathbf{x}_k) + \sum_{j \in \text{match}_F} \mathbb{E}[C_j] \tag{5}$$

4 Experimental Results

We evaluate the explainers on thousands of synthetic problems and popular real-world datasets. By varying the data and models, we identify when explainers fail, whether plausible explanations are faithful, and other interesting trends. See Appendix B for experimental setup and implementation details.

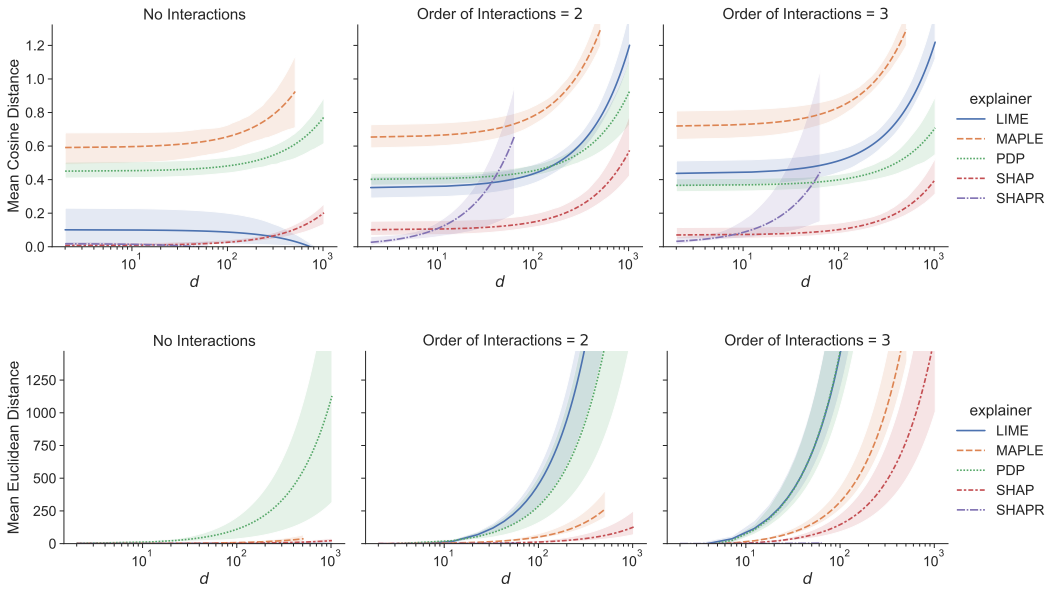

Figure 3: Average cosine distances (top) and Euclidean distances (bottom) between ground truth and explained effect contributions as a function of the number of features d and the order of interaction effects. As the dimensionality, the degree of interactions, and the number of interactions increase, the disagreement with ground truth grows for all explainers.

Evaluation We measure the error between the explanations of the ground truth and explainer using a few metrics. The set of contributions comprising each explanation can be thought of as a vector collectively, thus we can compute the distance between them after applying MatchEffects. First considered is Euclidean distance to understand the magnitude of the disagreement with the ground truth. To quantify the disagreement in orientation, we utilize cosine distance. Each measure of error here is the quantified *infidelity* of explanations.

Synthetic Problems We first demonstrate our approach on 2,000 synthetic models that are generated with a varied number of effects, order of interaction, number of features, degree of nonlinearity, and number of unused (dummy) variables. The explainers are evaluated on each model with access to the full dataset and black box access to the model. Some explainers failed to explain some models – see Appendix B for details.

Results demonstrate the efficacy of the proposed approach in understanding explanation quality, as well as factors that influence it when paired with the experimental design. As the dimensionality, the degree of interactions, and the number of interactions increase, the disagreement between ground truth and explanation grows. Figure 3 illustrates these results for all of the explained synthetic models. Because `LIME` failed to explain a substantial portion of synthetic models, it appears to improve with an increased d in the leftmost plots; in reality, it only succeeded in explaining simpler models with a larger d. `SHAP` performs the best relative to the other explainers, maintaining both a closer and more correctly-oriented explanation compared to the ground truth. Interestingly, the ranking of `LIME` and `MAPLE` swaps when comparing average cosine and Euclidean distances. Surprisingly, `SHAPR` struggles to handle interaction effects effectively – the baseline `SHAP` outperforms it substantially. Appendix D includes additional analyses and figures with synthetic models, including evaluation as a function of the number of number of interactions, number of nonlinearities, and dummy features.

Real-World Case Studies We evaluate GAMs and feature-additive NNs on several real-world datasets: Boston housing [50], COMPAS [9], FICO HELOC [40], and a down-sampled version of MNIST [75]. Table 1 contains the aggregate results across all real-world datasets for the considered models. Among the considered explainers, `SHAP` outperforms on all datasets and models, often by several orders of magnitude. Surprisingly, `SHAPR` performs worse than `SHAP`, but

Dataset	Model	Explainer Error					ρ_{perf}
		PDP	LIME	MAPLE	SHAP	SHAPR	
Boston	GAM	0.340	0.709	0.652	0.001	0.111	0.995
	NN	0.278	0.182	0.431	0.001	0.209	0.351
COMPAS	GAM	0.821	0.781	0.863	0.000	–	0.800
	NN	0.328	0.062	0.274	0.001	–	1.000
FICO	GAM	0.795	0.949	0.962	0.003	–	0.200
	NN	0.761	0.193	0.270	0.001	–	0.800
MNIST	CNN	0.660	0.253	0.318	0.049	0.175	0.410

Table 1: Real-world explainer results on several datasets for GAMs and NNs. Here, explainer error is the cosine distance averaged over all samples and classes, if applicable. ρ_{perf} is Spearman's rank correlation coefficient between the mean explanation cosine distance and explainer accuracy. SHAPR is not implemented for data with categorical variables in this work.

still ranks well compared to the other explainers. PDP, LIME, and MAPLE produce poor explanations in general, and all explainers struggled more with the GAMs than the considered NNs. To test whether explainer fidelity correlates with accuracy, we compute the Spearman's rank correlation coefficient ρ_{perf} between the mean explanation cosine similarity (explanation fidelity) and explainer accuracy. Recall that under the feature-additive perspective that the sum of the contributions from an explainer approximates the model output, which can be treated as the prediction of the explainer. The scores, shown in Table 1, demonstrate that a plausible explainer, *i.e*, one that predicts accurately, does not necessarily produce faithful explanations, and vice versa.

5 Discussion

The answer to our question – whether feature-additive explainers effectively explain feature-additive predictors – is a nuanced "no." It depends on the application, the data, and the model. However, typical NNs contain a greater number of interaction effects and order of interactions than those considered here – if an explainer underperforms on feature-additive white boxes, then it should not be expected to perform well with black box predictors.

The shortcomings of these explainers arise from their underlying assumptions, such as feature independence and the locality of linearity. These assumptions are further impacted by the explainer hyperparameters which require tuning dependent upon the data and model. In practice, these knobs can be adjusted until the explanations "look right," which is not realistic when the most faithful hyperparameters need to be derived from the black box itself. This is especially troubling as studies show that data scientists overtrust or do not understand interpretability techniques [68, 71]. With the results of our study, even those practitioners who do not abuse these explanation tools may still be mislead.

Our results corroborate findings in prior research. Post hoc explainers have been show to be unverifiable, unfaithful, inconsistent, incomplete, intractable, unsuitable for real-time applications, and/or untrustworthy [114, 106, 26, 10, 71, 17, 30, 22, 43]. Additionally, these methods can be fooled [112, 32, 33, 12]. However, they may increase user trust in AI systems [24], user performance under certain conditions [57], and trustlessly audit black boxes [23]. Nonetheless, post hoc explanation is often argued to be unsuitable for high-stakes applications [105]. Rather, intrinsically interpretable models should be favored [114, 106], which are more desirable to experts and can even be more accurate than their black box counterparts in high-stakes application domains [4, 60, 25, 52].

Future Work A natural extension of this work would be to evaluate additional explanation methods that consider interaction effects and guide the improvement of explainer quality. We believe that progress within this class of explainers will emerge by accounting for interdependence between features, better defining locality, and scaling computation for high-dimensional data. Last, we echo the arguments that XAI research needs to be rigorous with certifiable guarantees, clear and falsifiable hypotheses, and justified generalization checks [46, 94, 74].

References

[1] Kjersti Aas, Martin Jullum, and Anders Løland. Explaining individual predictions when features are dependent: More accurate approximations to shapley values. *Artificial Intelligence*, 298:1–24, September 2021. 2, 20

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 21

[3] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 700–712. Curran Associates, Inc., 2020. 18

[4] Michael Anis Mihdi Afnan, Yanhe Liu, Vincent Conitzer, Cynthia Rudin, Abhishek Mishra, Julian Savulescu, and Masoud Afnan. Interpretable, not black-box, artificial intelligence should be used for embryo selection, 2021. 7

[5] Chirag Agarwal, Eshika Saxena, Satyapriya Krishna, Martin Pawelczyk, Nari Johnson, Isha Puri, Marinka Zitnik, and Himabindu Lakkaraju. OpenXAI: Towards a transparent evaluation of post hoc model explanations. *arXiv preprint arXiv:2206.11104*, 2022. 18

[6] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey Hinton. Neural additive models: Interpretable machine learning with neural nets. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, pages 1–13, 2021. 5

[7] Nourah Alangari, Mohamed El Bachir Menai, Hassan Mathkour, and Ibrahim Almosallam. Exploring evaluation methods for interpretable machine learning: A survey. *Information*, 14(8), 2023. 2, 3, 18, 19

[8] Sajid Ali, Tamer Abuhmed, Shaker El-Sappagh, Khan Muhammad, Jose M. Alonso-Moral, Roberto Confalonieri, Riccardo Guidotti, Javier Del Ser, Natalia Díaz-Rodríguez, and Francisco Herrera. Explainable artificial intelligence (XAI): What we know and what is left to attain trustworthy artificial intelligence. *Information Fusion*, 99:101805, 2023. 2

[9] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: there's software used across the country to predict future criminals. And it's biased against blacks., 2016. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing. 6, 24

[10] Boris Babic, Sara Gerke, Theodoros Evgeniou, and I. Glenn Cohen. Beware explanations from AI in health care. *Science*, 373(6552):284–286, 2021. 7

[11] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140, July 2015. 2

[12] Hubert Baniecki. Adversarial explainable AI. https://hbaniecki.com/adversarial-explainable-ai/, 2023. Accessed: 2023-01-28. 7

[13] Cher Bass, Mariana da Silva, Carole Sudre, Petru-Daniel Tudosiu, Stephen Smith, and Emma Robinson. ICAM: Interpretable classification via disentangled representations and feature attribution mapping. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 7697–7709. Curran Associates, Inc., 2020. 18

[14] Christian F. Baumgartner, Lisa M. Koch, Kerem Can Tezcan, Jia Xi Ang, and Ender Konukoglu. Visual feature attribution using wasserstein GANs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 18

[15] Usha Bhalla, Suraj Srinivas, and Himabindu Lakkaraju. Verifiable feature attributions: A bridge between post hoc explainability and inherent interpretability. In *ICML 3rd Workshop on Interpretable Machine Learning in Healthcare (IMLH)*, 2023. 1, 18

[16] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren E. Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir P. Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher R'e, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A. Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models. *arXiv*, 2021. 1

[17] Sebastian Bordt, Michèle Finck, Eric Raidl, and Ulrike von Luxburg. Post-hoc explanations fail to achieve their purpose in adversarial contexts. *ACM Conference on Fairness, Accountability, and Transparency*, 5, 2022. 7

[18] Sebastian Bordt and Ulrike von Luxburg. From shapley values to generalized additive models and back. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 709–745. PMLR, 25–27 Apr 2023. 2, 19

[19] Rafaël Brandt, Daan Raatjens, and Georgi Gaydadjiev. Precise benchmarking of explainable AI attribution methods. *arXiv preprint arXiv:2308.03161*, 2023. 2, 18

[20] Joy Buolamwini and Timnit Gebru. Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. In *Conference on Fairness, Accountability and Transparency*, pages 77–91. PMLR, January 2018. 1

[21] Oana-Maria Camburu, Eleonora Giunchiglia, Jakob Foerster, Thomas Lukasiewicz, and Phil Blunsom. Can I Trust the Explainer? Verifying Post-hoc Explanatory Methods. *NeurIPS 2019 Workshop on Safety and Robustness in Decision Making*, 1:1–13, December 2019. 2, 18

[22] Zachariah Carmichael and Walter J. Scheirer. A framework for evaluating post hoc feature-additive explainers. *arXiv*, abs/2106.08376, 2021. 7

[23] Zachariah Carmichael and Walter J Scheirer. Unfooling perturbation-based post hoc explainers. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 2023. 7

[24] Selina Carter and Jonathan Hersh. Explainable AI helps bridge the AI skills gap: Evidence from a large bank. *Economics Faculty Articles and Research*, 276, 2022. 7

[25] Julius Chapiro. Explainable AI for prostate MRI: Don't trust, verify. *Radiology*, 307(4):e230574, 2023. 7

[26] Alicja Chaszczewicz. Is task-agnostic explainable AI a myth? *arXiv preprint arXiv:2307.06963*, 2023. 7

[27] Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 883–892. PMLR, 10–15 Jul 2018. 18

[28] Jonathan Crabbe, Yao Zhang, William Zame, and Mihaela van der Schaar. Learning outside the black-box: The pursuit of interpretable models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17838–17849. Curran Associates, Inc., 2020. 2, 3, 19

[29] CSIS. Sen. Chuck Schumer launches SAFE innovation in the AI age at CSIS. https://www.csis.org/analysis/sen-chuck-schumer-launches-safe-innovation-ai-age-csis, 2023. Accessed: 2023-09-13. 1

[30] Guy Van den Broeck, Anton Lykov, Maximilian Schleich, and Dan Suciu. On the tractability of SHAP explanations. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6505–6513. AAAI Press, 2021. 7

[31] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. ERASER: A benchmark to evaluate rationalized NLP models. *Transactions of the Association for Computational Linguistics*, pages 1–16, July 2020. 20

[32] Botty Dimanov, Umang Bhatt, Mateja Jamnik, and Adrian Weller. You shouldn't trust me: Learning models which conceal unfairness from multiple explanation methods. *Frontiers in Artificial Intelligence and Applications: ECAI*, 2020. 7

[33] Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. *Advances in neural information processing systems*, 32, 2019. 7

[34] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017. 1, 2, 18

[35] Li Du, Xiao Ding, Kai Xiong, Ting Liu, and Bing Qin. e-CARE: a new dataset for exploring explainable causal reasoning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 432–446. Association for Computational Linguistics, 2022. 18

[36] Council of the EU and European Parliament. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). *Official Journal of the European Union*, L 119:1–88, 2016. 1

[37] European Commission. Proposal for a regulation of the European Parliament and the Council: Laying down harmonised rules on Artificial Intelligence (Artificial Intelligence Act) and amending certain Union legislative acts. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206, 4 2021. 1

[38] Ming Fan, Jiali Wei, Wuxia Jin, Zhou Xu, Wenying Wei, and Ting Liu. One step further: Evaluating interpreters using metamorphic testing. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2022, page 327–339, New York, NY, USA, 2022. Association for Computing Machinery. 18

[39] Shi Feng and Jordan Boyd-Graber. What can AI do for me? Evaluating machine learning interpretations in cooperative play. In Wai-Tat Fu, Shimei Pan, Oliver Brdiczka, Polo Chau, and Gaelle Calvary, editors, *Annual Conference on Intelligent User Interfaces*, pages 229–239. ACM, March 2019. 20

[40] Fair Isaac Corporation (FICO). FICO explainable machine learning challenge: Home equity line of credit (HELOC) dataset, 2018. https://community.fico.com/s/explainable-machine-learning-challenge. 6, 24

[41] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001. 1, 2, 19

[42] Christopher Frye, Colin Rowat, and Ilya Feige. Asymmetric shapley values: incorporating causal knowledge into model-agnostic explainability. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1229–1239. Curran Associates, Inc., 2020. 18

[43] Damien Garreau and Ulrike von Luxburg. Explaining the explainer: A first theoretical analysis of LIME. In Silvia Chiappa and Roberto Calandra, editors, *International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1287–1296. PMLR, August 2020. 7

[44] L Gautier. rpy2: A simple and efficient access to R from Python. *URL http://rpy.sourceforge.net/rpy2.html*, 3:1, 2008. 21

[45] Yingqiang Ge, Shuchang Liu, Zelong Li, Shuyuan Xu, Shijie Geng, Yunqi Li, Juntao Tan, Fei Sun, and Yongfeng Zhang. Counterfactual evaluation for explainable AI. *arXiv preprint arXiv:2109.01962*, 2021. 2, 19

[46] Robert Geirhos, Roland S Zimmermann, Blair Bilodeau, Wieland Brendel, and Been Kim. Don't trust your eyes: on the (un) reliability of feature visualizations. *arXiv preprint arXiv:2306.04719*, 2023. 7

[47] Riccardo Guidotti. Evaluating local explanation methods on ground truth. *Artificial Intelligence*, 291:1–16, February 2021. 2, 19, 20

[48] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5):93:1–93:42, August 2018. 2

[49] Charles R. Harris, K. Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585:357–362, 2020. 21

[50] 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, March 1978. 6, 24, 28

[51] Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer, February 2009. 4

[52] Joshua Hatherley, Robert Sparrow, and Mark Howard. The virtues of interpretable medical artificial intelligence. *Cambridge Quarterly of Healthcare Ethics*, pages 1–10, 2022. 7

[53] Anna Hedström, Philine Bommer, Kristoffer K Wickstrøm, Wojciech Samek, Sebastian Lapuschkin, and Marina M-C Höhne. The meta-evaluation problem in explainable AI: Identifying reliable estimators with MetaQuantus. *arXiv preprint arXiv:2302.07265*, 2023. 1

[54] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Conference on Computer Vision and Pattern Recognition*, pages 15262–15271. IEEE/Computer Vision Foundation, 2021. 1

[55] Lukas Hoyer, Mauricio Munoz, Prateek Katiyar, Anna Khoreva, and Volker Fischer. Grid saliency for context explanations of semantic segmentation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 18

[56] Eugen Hruska, Liang Zhao, and Fang Liu. Ground truth explanation dataset for chemical property prediction on molecular graphs. *ChemRxiv*, 2022. 18

[57] Christina Humer, Andreas Hinterreiter, Benedikt Leichtmann, Martina Mara, and Marc Streit. Comparing effects of attribution-based, example-based, and feature-based explanation methods on ai-assisted decision-making. *OSF Preprints*, 2022. 7

[58] John D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3):90–95, 2007. 21

[59] Aya Abdelsalam Ismail, Mohamed Gunady, Hector Corrada Bravo, and Soheil Feizi. Benchmarking deep learning interpretability in time series predictions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6441–6452. Curran Associates, Inc., 2020. 18

[60] Vaishali Jain, Ted Enamorado, and Cynthia Rudin. The Importance of Being Ernest, Ekundayo, or Eswari: An Interpretable Machine Learning Approach to Name-Based Ethnicity Classification. *Harvard Data Science Review*, 4(3), jul 28 2022. https://hdsr.mitpress.mit.edu/pub/wgss79vu. 7

[61] Yunzhe Jia, James Bailey, Kotagiri Ramamohanarao, Christopher Leckie, and Michael E Houle. Improving the quality of explanations with local embedding perturbations. In *Proceedings of the 25th ACM SIGKDD International conference on knowledge discovery & Data Mining*, pages 875–884, 2019. 2, 18

[62] Yunzhe Jia, James Bailey, Kotagiri Ramamohanarao, Christopher Leckie, and Xingjun Ma. Exploiting patterns to explain individual predictions. *Knowledge and Information Systems*, 62(3):927–950, Mar 2020. 2, 18

[63] Li Jiangchun, Carlos Daniel C Santos, Michael Kuhlen, and Angertdev Singh. Pdpbox: v0.2.1, March 2021. 21

[64] Weina Jin, Xiaoxiao Li, and Ghassan Hamarneh. The XAI alignment problem: Rethinking how should we evaluate human-centered AI explainability techniques. *arXiv preprint arXiv:2303.17707*, 2023. 18

[65] Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *International Conference on Learning Representations*, 2020. 2, 19

[66] Joblib Development Team. Joblib: running python functions as pipeline jobs, 2020. 21

[67] Fredrik Johansson et al. *mpmath: a Python library for arbitrary-precision floating-point arithmetic (version 0.18)*, December 2013. `http://mpmath.org/`. 21

[68] Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting interpretability: Understanding data scientists' use of interpretability tools for machine learning. In Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik, editors, *Conference on Human Factors in Computing Systems*, pages 1–14. ACM, April 2020. 7

[69] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2668–2677. PMLR, 10–15 Jul 2018. 18

[70] Janis Klaise, Arnaud Van Looveren, Giovanni Vacanti, and Alexandru Coca. Alibi: Algorithms for monitoring and explaining machine learning models, 2019. 21, 28

[71] Satyapriya Krishna, Tessa Han, Alex Gu, Javin Pombra, Shahin Jabbari, Steven Wu, and Himabindu Lakkaraju. The disagreement problem in explainable machine learning: A practitioner's perspective. *arXiv:2202.01602*, pages 1–46, 2022. 7

[72] Himabindu Lakkaraju, Nino Arsov, and Osbert Bastani. Robust and stable black box explanations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5628–5638. PMLR, 13–18 Jul 2020. 2, 3, 19

[73] Sebastian Lapuschkin, Alexander Binder, Gregoire Montavon, Klaus-Robert Muller, and Wojciech Samek. Analyzing classifiers: Fisher vectors and deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 18

[74] Matthew L. Leavitt and Ari Morcos. Towards falsifiable interpretability research. In *NeurIPS Workshop on ML-Retrospectives, Surveys & Meta-Analyses*, pages 1–15. arXiv, 2020. 7, 18

[75] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 6, 25

[76] Library of Congress. H.R.6580 - 117th Congress (2021-2022): Algorithmic accountability act of 2022. `https://www.congress.gov/bill/117th-congress/house-bill/6580/text`, 2 2022. 1

[77] Zachary C. Lipton. The mythos of model interpretability. *ACM Queue*, 16(3):30, July 2018. 1

[78] Ninghao Liu, Donghwa Shin, and Xia Hu. Contextual outlier interpretation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 2461–2467. AAAI Press, 2018. 18

[79] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Conference on Neural Information Processing Systems*, pages 4765–4774, December 2017. 1, 2, 20, 28

[80] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19620–19631. Curran Associates, Inc., 2020. 18

[81] Sean McGregor. Preventing repeated real world AI failures by cataloging incidents: The AI incident database. In *AAAI Conference on Innovative Applications of Artificial Intelligence, IAAI*, pages 15458–15463. AAAI Press, 2021. 1

[82] Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, Amit Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka, Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. SymPy: symbolic computing in Python. *PeerJ Computer Science*, 3:e103, January 2017. 21

[83] D. Douglas Miller and Eric W. Brown. Artificial intelligence in medical practice: The question to the answer? *The American Journal of Medicine*, 131(2):129–133, 2018. 1

[84] Miquel Miró-Nicolau, Antoni Jaume-i Capó, and Gabriel Moyà-Alcover. A novel approach to generate datasets with XAI ground truth to evaluate image models. *arXiv preprint arXiv:2302.05624*, 2023. 2, 3, 19

[85] Christoph Molnar. *Interpretable Machine Learning*. Leanpub, 2019. https://christophm.github.io/interpretable-ml-book/. 2

[86] Sascha Mücke and Lukas Pfahler. Check mate: A sanity check for trustworthy AI. In Pascal Reuss, Viktor Eisenstadt, Jakob Michael Schönborn, and Jero Schäfer, editors, *Proceedings of the LWDA 2022 Workshops: FGWM, FGKD, and FGDB, Hildesheim (Germany), Oktober 5-7th, 2022*, volume 3341 of *CEUR Workshop Proceedings*, pages 91–103. CEUR-WS.org, 2022. 18

[87] Woo-Jeoung Nam, Shir Gur, Jaesik Choi, Lior Wolf, and Seong-Whan Lee. Relative attributing propagation: Interpreting the comparative contributions of individual units in deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 2501–2508, 2020. 18

[88] Karthikeyan Natesan Ramamurthy, Bhanukiran Vinzamuri, Yunfeng Zhang, and Amit Dhurandhar. Model agnostic multilevel explanations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5968–5979. Curran Associates, Inc., 2020. 2, 19

[89] Meike Nauta, Jan Trienes, Shreyasi Pathak, Elisa Nguyen, Michelle Peters, Yasmin Schmitt, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable AI. *ACM Comput. Surv.*, 55(13s), jul 2023. 2, 3, 18, 19

[90] Cathy O'Neil. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown, September 2016. 1

[91] Jose Oramas, Kaili Wang, and Tinne Tuytelaars. Visual explanation by interpretation: Improving visual feedback capabilities of deep neural networks. In *International Conference on Learning Representations*, 2019. 18

[92] Martin Pawelczyk, Sascha Bielawski, Johannes van den Heuvel, Tobias Richter, and Gjergji Kasneci. Carla: a Python library to benchmark algorithmic recourse and counterfactual explanation algorithms. *arXiv:2108.00783*, pages 1–22, 2021. 3, 19

[93] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011. 21

[94] Uwe Peters and Mary Carman. Unjustified sample sizes and generalizations in explainable AI research: Principles for more inclusive user studies. *arXiv preprint arXiv:2305.09477*, 2023. 7

[95] Gregory Plumb, Denali Molitor, and Ameet S. Talwalkar. Model agnostic supervised local explanations. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Conference on Neural Information Processing Systems*, pages 2520–2529, December 2018. 2, 19

[96] Gregory Plumb, Jonathan Terhorst, Sriram Sankararaman, and Ameet Talwalkar. Explaining groups of points in low-dimensional representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7762–7771. PMLR, 13–18 Jul 2020. 18

[97] Nina Poerner, Hinrich Schütze, and Benjamin Roth. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 340–350, Melbourne, Australia, July 2018. Association for Computational Linguistics. 18

[98] Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online, July 2020. Association for Computational Linguistics. 18

[99] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009. 1

[100] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2021. 21

[101] Yanou Ramon, David Martens, Foster J. Provost, and Theodoros Evgeniou. A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data: Sedc, LIME-C and SHAP-C. *Advances in Data Analysis and Classification*, 14(4):801–819, 2020. 3, 19

[102] Maria H Rasmussen, Diana S Christensen, and Jan H Jensen. Do machines dream of atoms? Crippen's logP as a quantitative molecular benchmark for explainable AI heatmaps. *SciPost Chemistry*, 2(1):002, 2023. 18

[103] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1135–1144. ACM, August 2016. 1, 2, 19, 28

[104] Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 2662–2670. AAAI Press, 2017. 18

[105] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, May 2019. 1, 7

[106] Cynthia Rudin. Why black box machine learning should be avoided for high-stakes decisions, in brief. *Nature Reviews Methods Primers*, 2(1):81, 2022. 7

[107] Adriel Saporta, Xiaotong Gui, Ashwin Agrawal, Anuj Pareek, Steven Q. H. Truong, Chanh D. T. Nguyen, Van-Doan Ngo, Jayne Seekins, Francis G. Blankenberg, Andrew Y. Ng, Matthew P. Lungren, and Pranav Rajpurkar. Benchmarking saliency methods for chest X-ray interpretation. *Nature Machine Intelligence*, 4(10):867–878, Oct 2022. 18

[108] Gesina Schwalbe and Bettina Finzel. A comprehensive taxonomy for explainable artificial intelligence: a systematic survey of surveys on methods and concepts. *Data Mining and Knowledge Discovery*, Jan 2023. 2

[109] Nikolai Sellereite, Martin Jullum, and Annabelle Redelmeier. *shapr: Prediction Explanation with Dependence-Aware Shapley Values*, 2021. R package version 0.2.0. 21

[110] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, October 2017. 2

[111] Daniel Servén, Charlie Brummitt, Hassan Abedi, and hlink. dswah/pygam: v0.8.0, October 2018. 21

[112] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In Annette N. Markham, Julia Powles, Toby Walsh, and Anne L. Washington, editors, *Conference on AI, Ethics, and Society*, pages 180–186. AAAI/ACM, February 2020. 7

[113] Akshayvarun Subramanya, Vipin Pillai, and Hamed Pirsiavash. Fooling network interpretation in image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 18

[114] Vinitra Swamy, Jibril Frej, and Tanja Käser. The future of human-centric eXplainable artificial intelligence (XAI) is not post-hoc explanations. *arXiv preprint arXiv:2307.00364*, 2023. 7

[115] Alona Sydorova, Nina Poerner, and Benjamin Roth. Interpretable question answering on knowledge bases and text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4943–4951, Florence, Italy, July 2019. Association for Computational Linguistics. 18

[116] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations*, pages 1–10, April 2014. 1

[117] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations*, pages 1–10, April 2014. 1

[118] Erico Tjoa and Cuntai Guan. A survey on explainable artificial intelligence (XAI): toward medical XAI. *IEEE Transactions on Neural Networks and Learning Systems*, 32(11):4793–4813, 2020. 1

[119] Michael Tsang, Sirisha Rambhatla, and Yan Liu. How does this interaction affect me? interpretable attribution for feature interactions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6147–6159. Curran Associates, Inc., 2020. 18

[120] U.S.-EU TTC. U.S.-EU joint statement of the Trade and Technology Council. https://www.commerce.gov/news/press-releases/2022/05/us-eu-joint-statement-trade-and-technology-council, 5 2022. Accessed: 2022-05-10. 1

[121] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995. 21

[122] Sahil Verma, John P. Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *arXiv:2010.10596*, pages 1–13, 2020. 2

[123] Tom Vermeire, Dieter Brughmans, Sofie Goethals, Raphael Mazzine Barbossa de Oliveira, and David Martens. Explainable image classification with evidence counterfactual. *Pattern Analysis and Applications*, 26:1–21, Jan 2022. 3, 19

[124] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. 21

[125] Minh Vu and My T. Thai. PGM-Explainer: Probabilistic graphical model explanations for graph neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12225–12235. Curran Associates, Inc., 2020. 18

[126] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021. 21

[127] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56–61, 2010. 21

[128] Adam White and Artur S. d'Avila Garcez. Measurable counterfactual local explanations for any classifier. In Giuseppe De Giacomo, Alejandro Catalá, Bistra Dilkina, Michela Milano, Senén Barro, Alberto Bugarín, and Jérôme Lang, editors, *European Conference on Artificial Intelligence*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 2529–2535. IOS Press, 2020. 2

[129] Sarah Wiegreffe and Ana Marasovic. Teach me to explain: A review of datasets for explainable natural language processing. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks*, 2021. 18

[130] Orcun Yalcin, Xiuyi Fan, and Siyuan Liu. Evaluating the correctness of explainable AI algorithms for classification. *arXiv preprint arXiv:2105.09740*, 2021. 18

[131] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. GNNExplainer: Generating explanations for graph neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 18

[132] Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting CNNs via Decision Trees. In *Computer Vision and Pattern Recognition*, pages 6261–6270. IEEE, 2019. 2

[133] Yujia Zhang, Kuangyan Song, Yiming Sun, Sarah Tan, and Madeleine Udell. "why should you trust my explanation?" understanding uncertainty in LIME explanations. *arXiv preprint arXiv:1904.12991*, 2019. 2, 18

[134] Jianlong Zhou, Amir H. Gandomi, Fang Chen, and Andreas Holzinger. Evaluating the Quality of Machine Learning Explanations: A Survey on Methods and Metrics. *Electronics*, 10(5):593, January 2021. 3, 19

[135] Yilun Zhou, Serena Booth, Marco Tulio Ribeiro, and Julie Shah. Do Feature Attribution Methods Correctly Attribute Features? *NeurIPS 1st Workshop on eXplainable AI approaches for debugging and diagnosis (XAI4Debugging)*, pages 1–22, 2021. 18

[136] Zihan Zhou, Mingxuan Sun, and Jianhua Chen. A model-agnostic approach for explaining the predictions on clustered data. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1528–1533, 2019. 2, 19

Supplemental Material

Table of Contents

A Expanded Details

Comprehensive Related Work

There are three main types of evaluations: application-grounded (real humans, real tasks), human-grounded (real humans, simplified tasks), and functionally-grounded (no humans, proxy tasks) [34]. Human- and application-grounded evaluations are expensive, subjective, and qualitative. However, they measure the human utility and effectiveness of explanations. Functionally-grounded metrics are concerned with proxies for the same objectives, but also can quantitatively score the fidelity of an explanation with respect to the model being explained [89]. The "Co-12" explanation properties include aspects about the user (e.g., context and controllability), presentation (e.g., compactness and confidence), and content (e.g., correctness and consistency) [89]. Whereas content-based explanation properties evaluate desirable proxy characteristics, such as the complexity of feature interactions (covariate complexity) or the similarity of explanations between like examples (continuity), evaluations of correctness, or *fidelity*, concern nothing but the explanation faithfulness with respect to the model being explained. If an explanation is unfaithful to the model, then the question of human utility, trust, or otherwise is irrelevant [89, 64, 74]. Thus, we are interested in the *functionally-grounded* evaluation of explanation *fidelity* in this paper.

Prior Evaluations of Explanation Fidelity While there are evaluation methods for post hoc explainers that act as surrogate predictors (e.g., by knowledge distillation) and models that jointly predict and explain [7, 89], we keep our review focused on fidelity evaluations applicable to the explainers detailed in Section 2. *Perturbation*-based approaches perturb the model or data and verify that the explanation changes proportionally to either the perturbation or the model [7, 89]. *Removal*-based approaches delete or mask input feature(s) and measure the correlation between model output change and explanation importance score [7, 89]. This can be done with single features, or incrementally with feature ordering determined by the explanation importance scores. The removal can also be accomplished via pixel-flipping, baseline substitutions, zero-padding, or cropping. While these approaches guarantee that explanations have certain desirable proxy properties, they do not guarantee that explainers are faithful to the exact model behavior [89, 64, 74]. That is unless the model is modified to have those constraints imposed and verified [15]. However, the question of explanation descriptive completeness of model behavior would still remain [89]. In turn, we are interested in fidelity evaluations using *ground truth* evaluations. However, this term is overloaded in the literature, so we delineate the three ways it is used here:

- **Annotation checks** measure the correlation between feature importance scores and annotated data that is deemed important to the task. Existing evaluations use human annotations, whether it is sample-wise annotations or a crafted annotation-generation process, that evaluate explainers via a proxy task, e.g., object localization or rationale generation [73, 87, 14, 13, 119, 107, 97, 113, 115, 38, 35, 86, 56, 129, 21]. However, this quantifies explanation plausibility rather than fidelity with respect to the model.
- **Controlled data checks** involve creating a dataset (typically synthetic) such that a well-performing model should follow some a priori reasoning. For example, this reasoning could be a region of an image belonging to an object of interest, a set of nodes in a graph belonging to a discriminative motif, or a subset of features that are deemed highly discriminative. Two types of evaluations have been studied:
  - *Feature selection* studies evaluate whether an explanation captures a subset of the important features according to the a priori reasoning [3, 55, 69, 91, 42, 78, 80, 131, 125, 96, 98, 104, 5, 102, 130, 135].
  - *Feature ranking* studies evaluate whether an explanation ranks the importance of (a subset of) features according to the a priori reasoning [59, 27].

  However, there is no guarantee that the model actually follows the a priori reasoning as it is still a black-box, even if it performs well on the data.
- **White box checks** evaluate the correspondence between explanations and the known white box reasoning. There are several types of evaluations that have
  - *Feature selection* approaches isolate a subset of features that the model uses, e.g., by having the model use a subset of features globally, only considering the features leading to the predicted leaf of a tree, or only considering the features that a rule comprises [19, 62, 61, 133].

- *Feature ranking* approaches identify the relative importance of each feature so they can be ranked, such as via the coefficients of a logistic regression model [45, 103, 136].
- *Inexact feature contributions* methods use a proxy measure to estimate feature contributions from a model, such as the gradient with respect to each feature in differentiable models or the Gini impurity of trees [65, 47, 88].
- *Exact feature contributions* methods identify the exact amount each feature contributes to the predicted outcome of a model with respect to a formal (and useful) definition of contribution. Typically, and in this paper, a *contribution* is the amount that a feature (or subset of features) at a particular value adds to the predicted outcome such that the sum of all contributions totals the model prediction. For instance, the prediction could be the number of times a non-overlapping pattern appears in an image, thus the additive contribution of each pixel is known [84]. Alternatively, the contributions can be taken from a linear regression model [28, 72]. In [18], a connection it is shown that GAMs (with or without interaction effects) can be recovered from Shapley values (with or without interaction effects). They demonstrate that interaction effects with an order of two can be precisely estimated, but can only be detected at higher orders with Shapley values in experiments.

White box checks offer the highest fidelity estimate of explanation fidelity as the form of the model, and thus how it uses the data, is well-understood. However, the feature selection and ranking approaches are limited in that the contribution of each feature is unknown. Exact feature contribution approaches are also able to evaluate both feature selection and ranking. In addition, they are more faithful to the true feature contributions within the explained model than inexact feature contributions as no proxy is needed. Our approach fits into this category of white box checks.

Our approach has several advantages over prior work. (1) First, [28, 72] define the exact feature contributions as the coefficients of a linear regression model to evaluate the fidelity of explainers, such as `LIME` and `SHAP`. However, this is inappropriate for these explainers and thus does not provide an exact set of feature contributions. `LIME` yields explanations as linear regression coefficients on normalized data, which must be taken into consideration for computing error. `SHAP` yields explanations as feature-additive contributions rather than coefficients – for a linear model, the corrected feature contribution is simply given by multiplication between each coefficient and each feature value. In addition, the contributions need to be adjusted for the baseline values that `SHAP` uses. We correct for these issues in our work for all considered explainers. (2) No prior work has considered the case of feature interactions [84, 28, 72], which are ubiquitous among black box models, especially neural networks. We consider models with a various number of feature interactions and order of feature interactions. (3) Unlike prior work [84, 28, 72], we consider in our experiments both synthetic and real-world data, both tabular and image data, as well as both non-learnable and learnable models, including convolutional neural networks.

Further comprehensive overviews of explanation evaluation methodologies and aspects are detailed in [7, 89, 134, 123, 92, 101].

Considered Local Post Hoc Explainers

- **Partial Dependence Plots** `PDPs` [41] estimate the average marginal effect of a subset of features on the output of a model using the Monte Carlo method. When the subset comprises one or two features, the model output is plotted as a function of the feature values. `PDPs` give a global understanding of a model, but can also yield a local explanation for the specific feature values of a sample.
- **Local Interpretable Model-agnostic Explanations** `LIME` [103] explains by learning a linear model from a randomly sampled neighborhood around z-score normalized instances. Feature selection is controlled by hyperparameters that limit the total number of features used in approximation, such as the top-k largest-magnitude coefficients from a ridge regression model.
- **Model Agnostic Supervised Local Explanations** `MAPLE` [95] employs a tree ensemble, e.g., a random forest, to estimate the importance (the net impurity) of each feature. Feature selection is performed upfront on the background data by iteratively adding important features to a linear model until error is minimized on held out validation data. For local explanations, `MAPLE` learns a ridge regression model on the background data distribution with samples weighed by the tree leafs relevant to the explained instance.

- **Shapley Additive Explanations** SHAP [79] takes a similar but distinct approach from LIME by approximating the Shapley values of the conditional expectation of a model. Feature selection is controlled using a regularization term. Note that when we write SHAP, we are specifically referring to Kernel SHAP, which is distinguished from its other variants for trees, structured data, etc. An extension of SHAP to handle dependent features has also been proposed [1], which we refer to as SHAPR after the associated R package. In effort to improve the accuracy of SHAP explanations, SHAPR estimates the conditional distribution assuming features are statistically dependent.

MatchEffects

MatchEffects is formalized in Algorithm 1 and illustrated for a few examples in Figure 4. This process guarantees the most fair and direct comparison of explanations, and does not rely on gradients, sensitivity, or other proxy means [47, 31, 39]. The worst-case time complexity of MatchEffects is $\mathcal{O}\left(m\hat{m}d\right)$ and the space complexity is $\mathcal{O}\left(m\hat{m}\right)$ (see Appendix C for proofs). It should be noted that in all practical use cases, the wall-time and memory bottlenecks of the framework arise from the explainers, especially those that scale combinatorially with d or require a reference data set that scales with n.

Algorithm 1: MatchEffects

Input: $D = \{D_j \mid 1 \leq j \leq m_F\}$, the set of feature subsets operated on by model F
Input: $\hat{D} = \{\hat{D}_k \mid 1 \leq k \leq m_{\hat{F}}\}$, the set of feature subsets operated on by explainer $\hat{F}$
Result: Corresponding sets of effects that can be compared

```
// add edges between effects with mutual features
E ← new array;
for D_j ∈ D do
    for D̂_k ∈ D̂ do
        if |D_j ∩ D̂_k| > 0 then
            E.append({D_j, D̂_k});
V ← D ∪ D̂;                                   // effects are vertices
G ← (V, E);
// find connected components (CCs) for the undirected graph G
CCs ← ConnectedComponents(G);
matches ← new array;
// V_c and E_c comprise the vertices and edges of component c, respectively
for {V_c, E_c} ∈ CCs do                      // unpack the components
    match_F ← new array;
    match_F̂ ← new array;
    for D_c ∈ V_c do
        if D_c ∈ D_F then
            match_F.append(D_c);
        else
            match_F̂.append(D_c);
    if match_F = match_F̂ then
        // elements of identical sets are each a perfect match
        for D_c ∈ match_F do
            matches.append({{D_c}, {D_c}});
    matches.append({match_F, match_F̂});
return matches
```

One could exploit MatchEffects by producing explanations that attribute the entire output of the model to a single effect comprising all d features; the comparison of contributions could trivially yield perfect but uninformative scores. Likewise, a model with such interaction effects, like most deep NNs, would render this evaluation inconsequential. To mitigate this issue, we introduce a metric that evaluates the goodness of the matching. Let E_c be the set of edges of a single component found by MatchEffects. For an edge $\{D_j, \hat{D}_k\} \in E_c$, the intersection-over-union (IoU), also known as the Jaccard index, is calculated between D_j and $\hat{D}_k$. The total goodness for a component is the average of the IoU scores of each edge in E_c, and the total goodness for

Dataset	MaloU				
	PDP	LIME	MAPLE	SHAP	SHAPR
Boston	0.979	0.979	0.214	0.979	0.979
COMPAS	0.971	0.971	0.286	0.971	–
FICO	0.750	0.659	0.648	0.659	–
MNIST	0.500	0.500	0.500	0.500	0.500

Table 2: The MaloU for each explainer on several real-world datasets. Note that MaloU is identical for both GAMs and NNs due to the experimental design: both are constrained to use the same (but still randomly selected) effects for each dataset. SHAPR is not implemented for data with categorical variables in this work.

a matching is the mean value of these averages: the mean-average-IoU (MaloU). This metric is given by Equation (6)

$$\text{MaloU}(CCs) = \frac{1}{|CCs|} \sum_{\{V_c, E_c\} \in CCs} \text{aIoU}(E_c) \tag{6}$$

and the average IoU (aIoU) is defined by Equation (7)

$$\text{aIoU}(E_c) = \frac{1}{|E_c|} \sum_{\{D_j, \hat{D}_k\} \in E_c} \frac{|D_j \cap \hat{D}_k|}{|D_j \cup \hat{D}_k|}. \tag{7}$$

where CCs is defined in Algorithm 1. MaloU can be thought of the degree to which the effects uncovered by an explainer agree with the true effects of the model. Figure 4c shows the effectiveness of MaloU on an example with three components, and Figure 4d shows how MaloU can inform when an explanation is uninformative and mechanistically incorrect, mitigating the aforementioned consequences.

Table 2 shows the MaloU for each explainer on each dataset. Note that MaloU is identical for both GAMs and NNs due to the experimental design: both are constrained to use the same (but still randomly selected) effects for each dataset. Of the explainers, MAPLE has the worst (lowest) average MaloU due to its feature selection process that picks relatively few features compared to the other explainers. LIME, SHAP, and SHAPR achieve the same scores as they all provide feature-wise explanations and feature selection is not forced. This favors explanation completeness over human-comprehensibility, which is more favorable for testing fidelity. PDP follows the same line of reasoning except for FICO; PDP provides non-zero estimates of several more features than LIME and SHAP. On the MNIST task, all explainers achieve the same MaloU – we explain this phenomenon in Appendix D following the details of the feature-additive CNNs.

B Reproducibility

Implementation and Setup

We use SymPy [82] to generate synthetic models and represent expressions symbolically as expression trees. This allows us to automatically discover the additivity of arbitrary expressions. See Appendix C for the unary and binary operators, parameters, and operation weights considered in random model generation. All stochasticity is seeded for reproducibility, and all code is documented and open-sourced[3]. The framework is implemented in Python [121] with the help of SymPy and the additional libraries NumPy [49], SciPy [124], pandas [127], Scikit-learn [93], Joblib [66], mpmath [67], pyGAM [111], PDPbox [63], alibi [70], TensorFlow [2], Matplotlib [58], and seaborn [126]. Furthermore, we build a Python interface to the R [100] package shapr [109] using rpy2 [44]. Appendix B also details hyperparameters used to train the GAMs and feature-additive NNs, and the hardware used to run experiments. Last, we consider the explanation of

[3]The source code for this work is available at github.com/craymichael/PostHocExplainerEvaluation

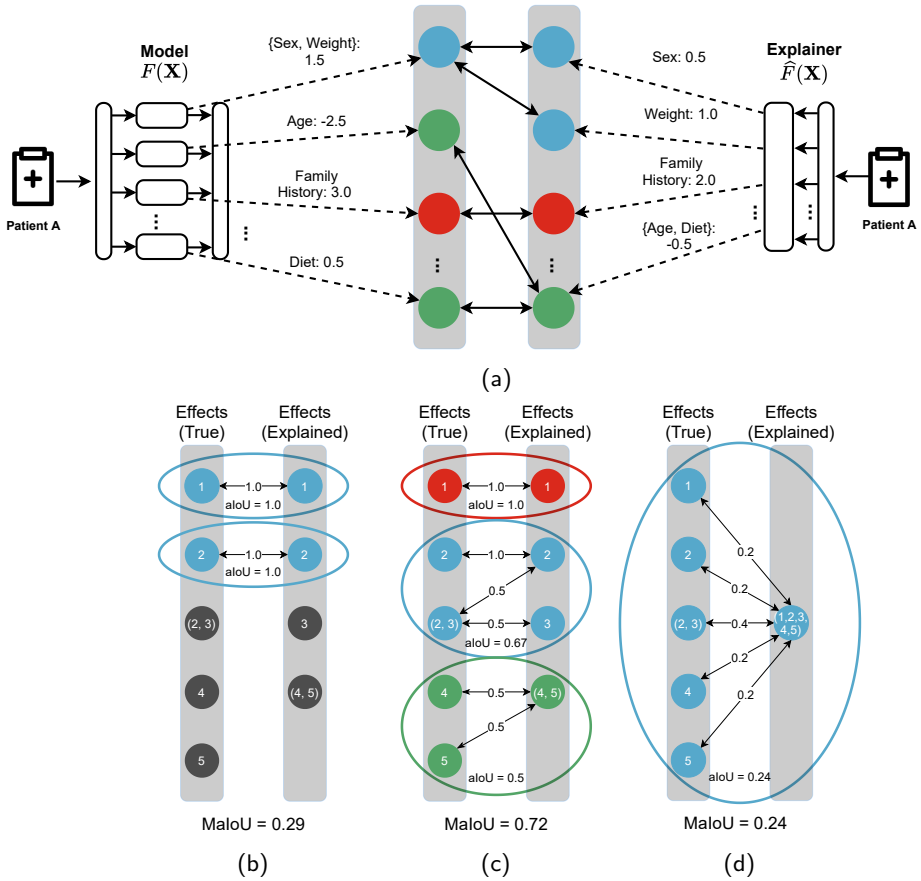

Figure 4: Examples of MatchEffects and MaIoU (Equation 6) in facilitating fair comparison between feature-additive explanations and *ground truth*. (a) A simple example demonstrating MatchEffects for a hypothetical medical task. Like colors from each side of the graph can be directly compared. While no explainer evaluated in this work explicitly detects interactions ($|\hat{m}| \geq 2$), the visual demonstrates how such explainers are compatible with the framework. (b) A strict one-to-one matching between effects severs partially correct effects from comparison and yields an over-penalizing MaIoU. (c) With MatchEffects, effects are fairly grouped together for comparison and a more reasonable MaIoU is given – ideally, the sum of the true contributions is equivalent to the sum of the explained contributions in each group (component). (d) Importantly, MaIoU defines the goodness of a match, in this case indicating that a superficially perfect explanation with MatchEffects (the sum of feature contributions is equivalent within in each group) is uninformative and incorrect.

an effect to be 0 if every estimated contribution is within a tolerance[4]. This is a fairer evaluation and tends to favor the explainers in experiments when dummy variables are present.

Model Generation Parameters

See Algorithm 2 for definitions of parameters. The * in Table 3 means '1' is implied when $pct_{interact}$ is zero.

Parameter	Values
d	{2, 4, 7, 16, 32, 64, 127, 256, 512, 1024}
n_{dummy}	{0, 0.2375d, 0.475d, 0.7125d, 0.95d}
$pct_{nonlinear}$	{0, .375, .75, 1.125, 1.5}
$pct_{interact}$	{0, 0.167, 0.333, 0.5}
$order_{interact}$	{1*, 2, 3}

Table 3: Model generation parameters

Weights are the probability of being drawn as an operator (normalized against all considered operators in the considered classes). The add operation is only considered when the operator does not break up the interaction effect. The values of n_{dummy} are a function of d.

Name	Type	Nonlinear	Weight
$\cosh(\cdot)$	unary	yes	0.015
$\cosh(\cdot)$	unary	yes	0.015
$\sin(\cdot)$	unary	yes	0.015
$\sinh(\cdot)$	unary	yes	0.015
$\text{asinh}(\cdot)$	unary	yes	0.015
$\tan(\cdot)$	unary	yes	0.015
$\tanh(\cdot)$	unary	yes	0.015
$\text{atan}(\cdot)$	unary	yes	0.015
$\cot(\cdot)$	unary	yes	0.015
$\text{acot}(\cdot)$	unary	yes	0.015
$\csc(\cdot)$	unary	yes	0.015
$\text{sech}(\cdot)$	unary	yes	0.015
$\text{sinc}(\cdot)$	unary	yes	0.015
$\lvert\cdot\rvert$	unary	yes	0.133
$\sqrt{(\cdot)}$	unary	yes	0.133
$(\cdot)^2$	unary	yes	0.133
$(\cdot)^3$	unary	yes	0.133
$\exp(\cdot)$	unary	yes	0.133
$\log(\cdot)$	unary	yes	0.133
$(\cdot) \times (\cdot)$	binary	no	0.8
$(\cdot)/(\cdot)$	binary	no	0.2
$(\cdot) + (\cdot)$	binary	no	-
$\min(\cdot, \cdot)$	binary	yes	0.5
$\max(\cdot, \cdot)$	binary	yes	0.5

Table 4: Operators considered

Explainer Hyperparameters

In general, the defaults were used and no tuning was performed. We only allowed explainers to explain as many effects as possible as the goal wasn't to produce comprehensible explanations, but rather faithful ones as the only criteria. See the table below for specified parameters of interest. Note that we do not use L1 regularization with `SHAP` as far too many features would be filtered and we do not tune explainer hyperparameters.

[4]See the documentation of `numpy.allclose` for details

Explainer		
LIME	num_samples	5000
	num_features	d
	discretize_continuous	False
	feature_selection	'auto'
MAPLE	train_size	2/3
	fe_type	'rf'
	n_estimators	200
	max_features	0.5
	min_samples_leaf	10
	regularization	1e-3
SHAP	n_background_samples	100
	summarization	kmeans
	l1_reg	False

Table 5: Explainer hyperparameters

PDP Local Explanations

To generate local explanations using PDP, we compute the PDP for each feature individually. The PDPBox library uses percentiles to sample the domain of each feature. We compute PD for 100 sample points for each feature. Thereafter, we use linear interpolation between each point, and extrapolation for values outside the range, to give a feature contribution for "unseen" values. The local explanation is thus the interpolated PD of each feature.

Dataset Descriptions

We demonstrate our framework on 2,000 synthetic models that are generated with a varied number of effects, order of interaction, number of features, degree of nonlinearity, and number of unused (dummy) variables. For each, we discard models with invalid ranges and domains that do not intersect with the interval $[-1, 1]$. The data of each feature $\mathbf{x}_{*,i}$ is sampled independently from a uniform distribution $\mathcal{U}(-1, 1)$. We draw n samples quadratically proportional to the number of features d as $n = 500\sqrt{d}$. The explainers are evaluated on each model with access to the full dataset and black box access to the model. Of the 2,000 models, 16 were discarded due to the input domain producing non-real numbers. Furthermore, some explainers were not able to explain every model due to invalid perturbations and resource exhaustion[5]. The former occurred with PDP, LIME, and SHAP, typically due to models with narrower feature domains, while the latter occurred with MAPLE and SHAPR due to the inefficient use of background data and intrinsic computational complexity. In total, 82%, 39%, 80%, 91%, and 40% of models were successfully explained by PDP, LIME, MAPLE, SHAP, and SHAPR, respectively. We consider the failure to produce an explanation for valid input to be a limitation of an explainer or its implementation.

The Boston housing dataset [50] contains median home values in Boston, MA, that can be predicted by several covariates, including sensitive attributes, e.g., those related to race. Models that discriminate based off of such features necessitate that their operation be exposed by explanations.

We also evaluate explainers on the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) recidivism risk dataset [9]. The dataset was collected by ProPublica in 2016 and contains covariates, such as criminal history and demographics, the proprietary COMPAS risk score, and recidivism data for defendants from Broward County, Florida.

The FICO Home Equity Line of Credit (HELOC) dataset [40], introduced in a 2018 XAI challenge, is also used in this work. It comprises anonymous HELOC applications made by consumers requesting a credit line in the range of $5,000 and $150,000. Given the credit history and

[5]See Appendix B for hardware and time budgets.

characteristics of an applicant, the task is to predict whether they will be able to repay their HELOC account within two years.

Last, we evaluate on a down-sampled version of the MNIST dataset [75]. With the aim of reducing explainer runtime and improving comprehensibility of effect-wise results, we crop and then resize each handwritten digit in the dataset to 12×10 and only include a subset of the 10 digits. We evaluate explainers on a down-sampled version of the MNIST dataset as mentioned in the text. Again, all steps taken here are to reduce explainer run-times and improve the comprehensibility of the results. We select a subset of classes to reduce the amount of data and the number of classes to explain. Specifically, we include only the four digits 0, 1, 5, and 8 due to separability between the data of each class. The crop was selected by observing the percentage of non-zero pixels that would be removed for all crop values, i.e., of the top and bottom rows, and the left and right columns. We remove about 1% of all non-zero pixels by using a global crop of 3 pixels from the top, 2 from the bottom, 5 from the left, and 3 from the right. This crop changes each image size from 28×28 to 23×20. We then resize each handwritten digit in the dataset to 12×10 using the `scikit-image` function `resize` with anti-aliasing.

Experiment Reproducibility

While all experiments use random seeds, some results may not be completely reproducible due to the behavior of `SymPy` (see the discussion in issue #20522[6]). For instance, the model generation uses the `sympy.calculus.util.continuous_domain` function to determine if generated models have valid input domains. This function randomly iterates through assumptions, and due to bugs, may not converge to the same result. Thus, we provide every `SymPy` model in the `pickle` format, and the generated data, and true contributions, and the explainer contributions in a `NumPy` format. The latter files should not suffer from reproducibility issues, but are provided to guarantee reproducibility. These files are located in a shared Google Drive folder: https://drive.google.com/drive/folders/1cBDwi4JIXmAihOv9yfjqrLNsohMCX5W?usp=sharing.

The source code is also linked in the main paper with the same seeds used in our experiments as the default arguments.

For the neural network, we train each pathway (for each effect) with 3 fully-connected layers [64, 64, 32] with the first two using a ReLU activation and the latter with no activation (identity function). The Adam optimizer is used with a learning rate of 1e-3 and early stopping with a patience of 100 based on the training loss, restoring the best weights at the end of training. The maximum number of epochs is 1,000. For the GAM, a spline term is added with 25 splines for each main effect, and a tensor term with 10 splines per marginal term is used for interaction effects. The link function is logistic for classification and identity for regression.

The convolutional neural network (CNN) uses the same optimization hyperparameters but a slightly different architecture. The first layer is 2D convolution with a kernel size and stride size of (2, 1), `SAME` padding (i.e., with a stride of one, the filtered output shape is the same as the input shape), and 4 filters. This implies sparsity within the model, thus additive structure. A dense layer then gives the output for each interaction effect from the output of the convolutional layer; due to the kernel and strides sizes, the filter outputs will all comprise the nonlinear ReLU function of 2 features.

For the real-world experiments, data is normalized (z-score normalization) before training. Training uses the full dataset as generalization is not of interest — rather, we only care if explainers can faithfully explain the model's predictions. Feature contributions and data are inverse normalization in all figures for better readability in terms of the underlying features.

Hardware

Experiments were run on a cluster running the Univa Grid Engine (UGE) software. Each job was allocated 16 cores of an Intel(R) Xeon(R) CPU E5-2680 v3 @ 2.50GHz and 10 GiB of RAM (soft maximum) per explanation of a model. Note that by pooling resources, a set of explanation jobs can contend for and pool up to 128 GiB of RAM. The operating system in use was Red

[6] https://github.com/sympy/sympy/issues/20522

Hat Enterprise Linux Server release 7.9 (Maipo). For total time running synthetic experiments, LIME took ∼6 hours for all explanations, SHAP took ∼18 hours for all explanations, and MAPLE exceeded a 2 week budget (although, each run was faster than SHAP up until $d > 64$). SHAPR exceeded the memory limits for several processes, as well as the time budget of 2 weeks. PDP finished all jobs in 6 days.

Licenses

The FICO HELOC dataset license is available at https://community.fico.com/s/explainable-machine-learning-challenge?tabset-3158a=a4c37. MNIST is under the Creative Commons Attribution-Share Alike 3.0 license. For software, see the corresponding licenses of the cited libraries in the text. Our software is under the MIT License.

C Proofs and Derivations

Proof of MatchEffects Complexities

The worst-case time complexity of MatchEffects is $\mathcal{O}\left(m\hat{m}d\right)$ and the space complexity is $\mathcal{O}\left(\max\left(d(m+\hat{m}), m\hat{m}\right)\right)$ (note that we write $\mathcal{O}\left(m\hat{m}\right)$ in the text for simplicity as the number of effects is almost always $\gg d$ in practical usage of the algorithm). Here we prove these claims, starting with the time complexity.

Lines 2-5 perform the following number of set intersections in the worst-case:

$$\begin{aligned} & \mathcal{O}(|D||\hat{D}|) \\ = \; & \mathcal{O}(m\hat{m}) \end{aligned}$$

Similarly, this is the worst-case number of edges $|E| = m\hat{m}$, which occurs if the bipartite graph is fully-connected (all effects relate to all other effects). Each set intersection (linear with hash sets in the implementation) has the following worst-case time complexity:

$$\begin{aligned} & \mathcal{O}\left(\max\left(\left\{|D_j| \mid D_j \in (D \cup \hat{D})\right\}\right)\right) \\ = \; & \mathcal{O}\left(|D_j^{\max}|\right) \\ = \; & \mathcal{O}\left(d\right) \end{aligned}$$

In the absolute worst-case every subset has d features in the effect. Thus, the time complexity of these lines is $\mathcal{O}\left(m\hat{m}d\right)$.

In line 6, we compute the union of feature subsets (they become the graph vertices).

$$\begin{aligned} & \mathcal{O}\left(|V|\right) \\ = \; & \mathcal{O}\left(|D| + |\hat{D}|\right) \\ = \; & \mathcal{O}\left(m + \hat{m}\right) \end{aligned}$$

Set union takes linear time.

Line 8 performs the well-known connected components algorithm using graph traversal (BFS/DFS). Thus this takes

$$\begin{aligned} & \mathcal{O}(|V| + |E|) \\ = \; & \mathcal{O}\left(m + \hat{m} + m\hat{m}\right) \\ = \; & \mathcal{O}\left(m\hat{m}\right) \end{aligned}$$

time.

Lines 10-22 will traverse through each vertex exactly once (the vertices comprising each V_c are guaranteed to be unique). For each vertex, checking membership in a set takes ($\mathcal{O}(1)$) time for each check. Thus, we have $\mathcal{O}\left(|V|\right) = \mathcal{O}\left(m + \hat{m}\right)$ for these lines. The match equality comparisons over all iterations in the loop will also take the same time per the guarantee each vertex is visited once.

Thus, the worst-case time complexity is $\mathcal{O}\left(m\hat{m}d\right)$ □

Here we consider the space complexity. The size of the graph is simply the size of the vertices and edges $\mathcal{O}\left(|V| + |E|\right) = \mathcal{O}\left(m\hat{m}\right)$. Note that the graph is represented in a sparse format (nonzero values only), though this doesn't reduce space in the dense worst-case scenario.

The other space to consider is from matches. This contains sets, each with two sets of effects (ground truth and explained). For efficiency, each effect is represented as an index, reducing the space required from $\mathcal{O}(d)$ to $\mathcal{O}(1)$. Thus matches ($|V|$ effects total, no matter how large a single match is) takes $\mathcal{O}(m + \hat{m})$ space.

Last, the input data is the same size as matches, except effects are not represented as indices. Therefore, the input data (D and $\hat{D}$) takes $\mathcal{O}(d(m + \hat{m}))$ space.

The total space required is:

$\mathcal{O}\left(\max\left(d(m + \hat{m}), m\hat{m}\right)\right)$ □

Derivation of Equivalence Relations

`LIME`

For `LIME`, we derive the unnormalized coefficients for use in producing contributions. `LIME` uses z-score normalization ($(x_i - \mu_i)/\sigma_i$) on the input data before learning local linear regression models.

$$\hat{F}(\mathbf{x}) = \theta_0 + \sum_i^d \frac{x_i - \mu_i}{\sigma_i}\theta_i$$

$$\hat{F}(\mathbf{x}) = \theta_0 + \sum_i^d \left(\frac{x_i}{\sigma_i} - \frac{\mu_i}{\theta_i}\right)$$

$$\hat{F}(\mathbf{x}) = \left(\theta_0 - \frac{\mu_i}{\theta_i}\right) + \sum_i^d \frac{x_i}{\sigma_i}$$

Thus we simply just need to scale the coefficients as follows (same as the main text)

$$\theta_0' = \theta_0 - \sum_i \frac{\mu_i\theta_i}{\sigma_i}$$

$$\theta_i' = \frac{\theta_i}{\sigma_i}$$

where θ_0' is the adjusted bias term and each θ_i' is an adjusted coefficient.

`SHAP`

`SHAP` estimates the contributions relative to the mean-centered model response. In other words:

$$\hat{F}(\mathbf{x}) \approx F(\mathbf{x}) - \mathbb{E}\left[F(\mathbf{x})\right]$$

Thus, the `SHAP` estimation can be written as follows

$$\hat{F}(\mathbf{x}) = \sum_i^d \hat{f}_i\left(\mathbf{x}_i\right) - \mathbb{E}[f_i(\mathbf{x}_i)]$$

due to the fact that

$$\begin{aligned}\mathbb{E}\left[F(\mathbf{x})\right] &= \mathbb{E}\left[\sum_{j}^{m} f_j\left(\mathbf{x}_{D_j}\right)\right] \\ &= \sum_{j}^{m} \mathbb{E}\left[f_j\left(\mathbf{x}_{D_j}\right)\right].\end{aligned}$$

So for each contribution, we can write that of the explainer as

$$\hat{C}_i = \hat{f}_i(\mathbf{x}_i) + \mathbb{E}[C_i]$$

in order to correct for the removed expected value. However, this assumes that there is some $i = j = k$ for every $f_j(\cdot)$ and $\hat{f}_k(\cdot)$ of the white box and explainer. As this is not always the case, we have to consider the effects of a match holistically. This then gives us the final relation for some match:

$$C_{\text{match}_{\hat{F}}} = \sum_{k \in \text{match}_{\hat{F}}} \hat{f}_k(\mathbf{x}_k) + \sum_{j \in \text{match}_F} \mathbb{E}[C_j]$$

This same process applies to SHAPR.

D Additional Results and Figures

We also visualize a subset of results in Figure 5 as feature shapes. We consider normalized (interquartile) root-mean-square error (NRMSE) for comparing individual effects, as defined by Eq. (8)

$$\text{NRMSE}(\mathbf{a}, \mathbf{b}) = \frac{1}{Q_3^{\mathbf{a}} - Q_1^{\mathbf{a}}} \sqrt{\frac{\sum_i^n (a_i - b_i)^2}{n}} \tag{8}$$

where $Q_3^{\mathbf{a}}$ and $Q_1^{\mathbf{a}}$ are the third and first quartiles of $\mathbf{a}$, respectively. This measure of error is the quantified *infidelity* of explanations. This shows more clearly that several explainers do not faithfully explain some of the feature contributions. For example, SHAPR, MAPLE, and LIME fail to satisfactorily unearth how the proportion of African Americans living in an area (feature B), according to the NN, drive the housing price; SHAPR produces a high-variance estimate ($\text{NRMSE} = 1.68$), MAPLE fails to even detect the effect ($\text{NRMSE} = 3.59$), and LIME only is able to approximate the mean contribution value ($\text{NRMSE} = 3.04$). This type of failure is incredibly misleading to any user and potentially damaging if the model is deployed. Fortunately, in this instance, SHAP reveals this relationship within reasonable error ($\text{NRMSE} = 0.129$). The COMPAS visualization shows another example of explanations of the "Age" feature of a GAM. Again, several explainers produce misleading and noisy explanations. Notably, some explained feature shapes correlate with the ground truth (e.g., "RAD") but are offset (the expected value of the feature contributions deviate). In turn, this becomes a problem when the ranks of feature contributions are considered, which is how many interpretability tools present explanations [70, 79, 103].

Surprisingly, SHAPR struggles to handle interaction effects effectively – the baseline SHAP outperforms it substantially. While the quantitative comparison is clear, it is difficult to intuit poor explanations. In turn, we visualize an instance of an explained interaction effect by the best-performing explainer, SHAP, in Figure 6. As the feature value on the y-axis decreases, SHAP and the ground truth contributions deviate exponentially (average cosine distance of 0.492 and Euclidean distance of 2.54).

Looking at aggregate metrics is not sufficient to understand how poor an explanation may be. We consider the utility of an explainer in high-stakes applications to be limited by its worst explanation. Consequently, we visualize the worst explanations from each explainer and show

[8] While the Boston housing dataset is widely studied as a baseline regression problem, the data column ("B") is notably controversial; the original paper [50] includes and preprocesses the data as $B = 1000(B' - 0.63)^2$ where B' is the proportion of African Americans by town.

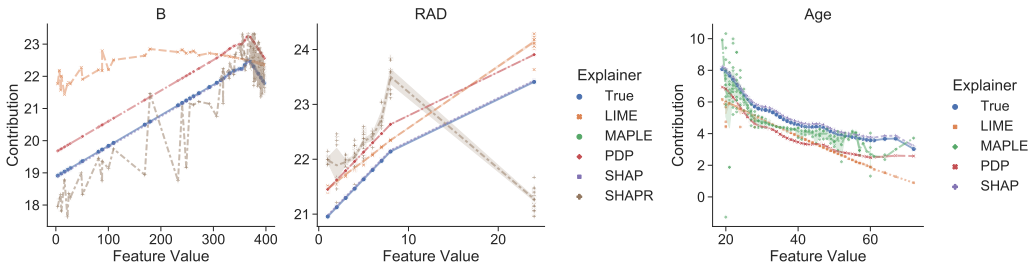

Figure 5: The true and explained feature shapes of (a) RAD (index of accessibility to radial highways) and B (proportion of African American population by town[8]) from a NN trained on the Boston housing dataset. `SHAPR`, `MAPLE`, and `LIME` fail to satisfactorily unearth how the proportion of African Americans living in an area (feature B), drive the housing price. Fortunately, in this instance, `SHAP` reveals this relationship within reasonable error. (b) The true and explained feature shapes of Age from a GAM trained on the COMPAS dataset. Several explainers produce misleading and noisy explanations, and some explained feature shapes correlate with the ground truth but are offset across all feature values. See Appendix C for feature shapes of all remaining main effects on each task.

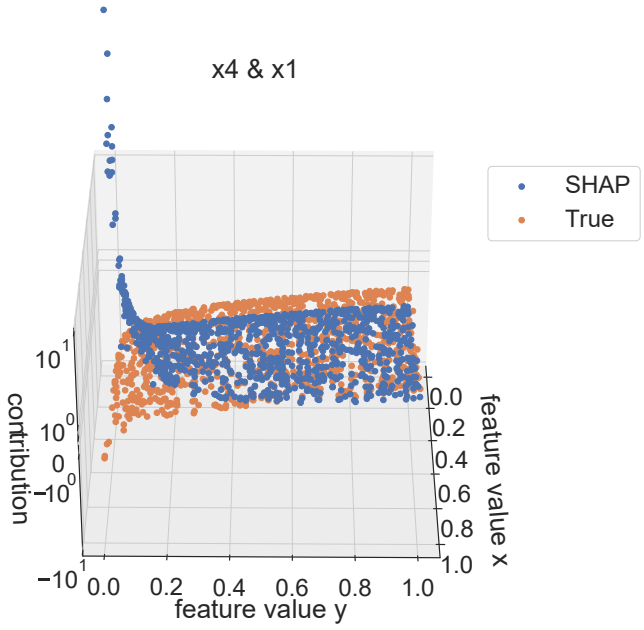

Figure 6: `SHAP` explanations for the generated synthetic expression: $x_1 + e^{x_4} + \log(x_1 x_4) + \frac{x_4}{x_1}$. The expression is an interaction effect of the two variables x_1 and x_4, so the application of MatchEffects results in the comparison of the sum of the explained contributions of each variable. As feature value y approaches 0, the `SHAP`-estimated contributions deviate exponentially from the ground truth. See Appendix C for additional angles and examples of explanations.

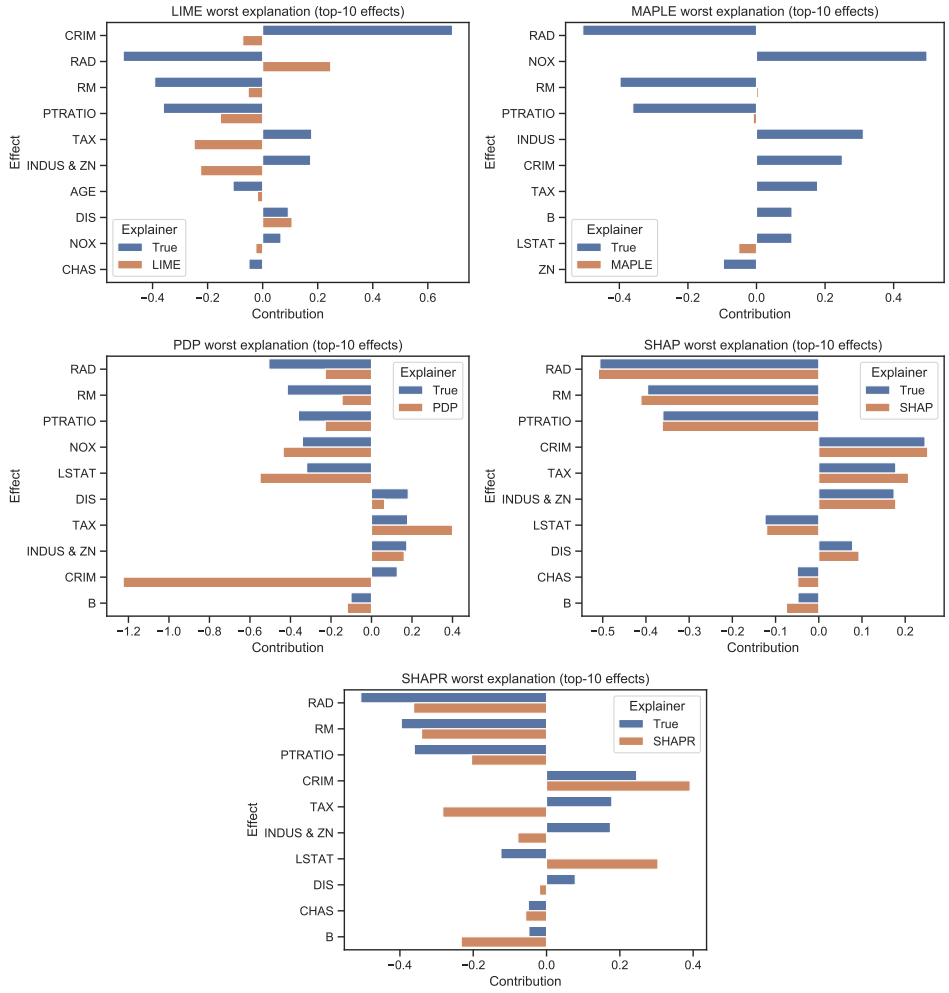

Figure 7: The top-10 explained effects of the worst explanation of a GAM trained on the Boston dataset from each explainer. Top effects are ranked by magnitude and the quality of explanation is ranked by the mean cosine distance among all explained samples.

those on the Boston dataset in Figure 7. The 10 most relevant effects to each decision are shown and the quality of the explanation is assessed using cosine distance. Again, the low point of SHAP is still of relatively high quality, and the other explainers reveal incorrect attributions of effects. MAPLE selects few important features correctly and does so conservatively. PDP, LIME, and SHAPR all problematically assign opposite-signed contributions for several effects. Similarly, we visualize the worst explanations for the MNIST task in Figure 8. Every explainer manages to flip the sign of at least a few contributions with the extreme case being PDP. While SHAP performs notably better than the other explainers, as shown in Table 1, its worst explanation is qualitatively misleading with some exaggerated contributions and some contributions with the opposite sign. The MNIST task of explaining the CNN is more difficult due to the higher dimensionality of the data and the number of interaction effects in the CNN. These interaction effects are of course due to the convolutional layers that operate over local neighborhoods of pixels. Appendix A goes into greater detail on this point.

E Synthetic Model Generation

Synthetic models are generated as described by Algorithm 2, GenerateModel. This algorithm takes in three absolute parameters: the number of features, the number of dummy (unused) features, and the order of interactions. It also takes in two relative parameters: the percentage of

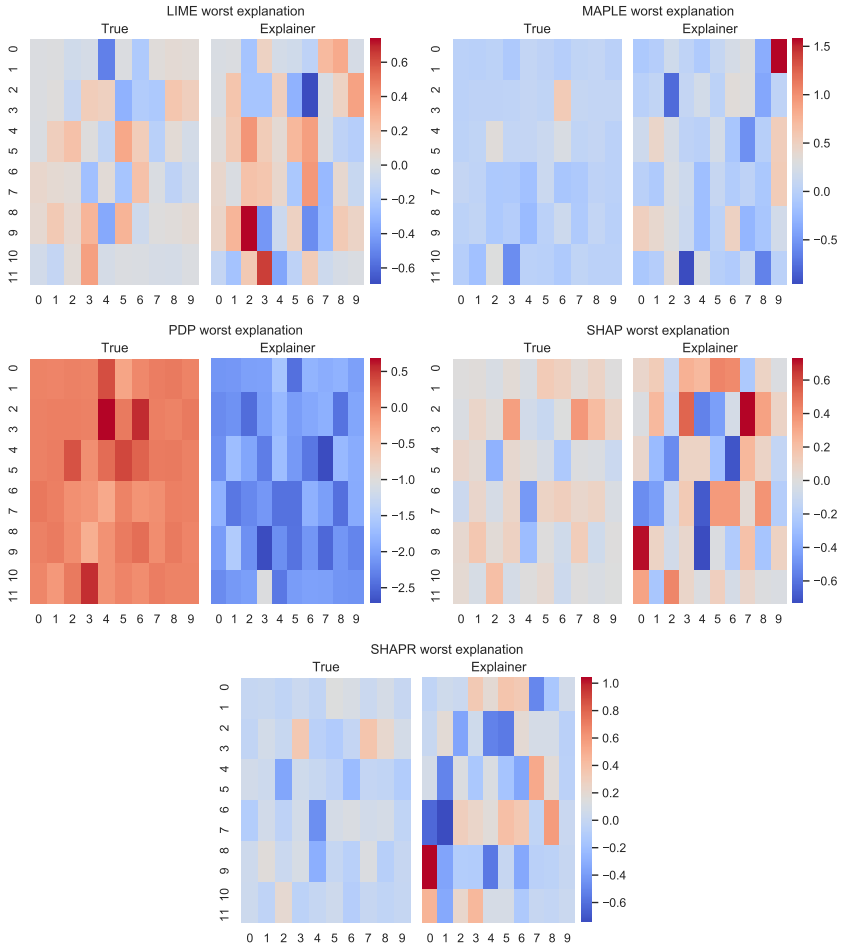

Figure 8: The heatmap of explained interaction effects of the worst explanations by `LIME`, `MAPLE`, `PDP`, `SHAP`, and `SHAPR` (left-to-right, top-to-bottom, respectively) for a CNN trained on the MNIST dataset as described in the text. See Appendix C for details on the CNNs in experiments and the derived interaction effects. The worst explanation is determined by the cosine distance from the ground truth explanations.

nonlinear operators and the percentage of interaction terms. Generation is split into four phases: nonlinear main effects, linear main effects, nonlinear interaction effects, and linear interaction effects. These phases are marked by corresponding comments in the algorithm.

Before any phase, we select the unique features to use in the model, which is simply the d features with the dummy features removed from consideration. After model generation, data is still drawn for these unused variables, but the model ignores it. For nonlinear main effects, we use at most the percentage of nonlinear operators times the number of features as the number of effects to generate. If this product is larger than the number of features, then we determine that the residual nonlinear operators will be applied to effects multiple times. For example, for $d = 2$ and a nonlinear percentage of 2 (200%), we may end up with something like $\cos(|x_1|) + \exp(\sqrt{x_2})$. This describes the steps where we place operators into some amount of bins (which is done as uniformly as possible with the values of each bin being an integer). Following this selection, we simply iterate over the unique features, applying the unary nonlinear operators to each, and add the result (still in symbolic form) to the expression. For linear main effects, we simply add the number of remaining features that have not had nonlinearities applied, if any, to the expression.

We have two parameters to consider for interaction effects: the interaction order (the number of features involved in each interaction) and the percentage of interaction terms (treated in the same manner as the percentage of nonlinear operators). We first select the unique interactions based

on the number of interactions specified and the number of main effects. This is a simple way of constraining the sparsity of generated models — with too many interaction terms, separation of effect contributions may not be possible by MatchEffects. The unique interactions selected are also naturally limited by the number of possible unique combinations given the number of features and the order of interactions. From these interactions, we select the number to be nonlinear in the exact same way as the main effects. However, we make choices from both unary and binary operators — binary operators are used to bridge together terms to form a whole effect and can include linear binary operators if the number of nonlinear operators is not sufficient to do so (*i.e.*, less than the number of features in an interaction minus one). Finally, we select the remaining linear interactions, choose linear interaction operators, and additionally add these to the expression.

See the previous supplemental content listing the unary and binary operators. The implementation of this algorithm has all randomness, *e.g.*, choices, seeded. For simplicity, the data structures (binary expression trees), random choices with operator weights, and valid domain checking are omitted from this algorithm.

Algorithm 2: GENERATEMODEL: Generates a synthetic model satisfying various arguments

Input: d: the number of features
Input: n_{dummy}: the number of unused features
Input: $pct_{nonlinear}$: the percentage of nonlinearities used (relative to d)
Input: $pct_{interact}$: the percentage of interaction terms (relative to d)
Input: $order_{interact}$: the order of interaction terms (≥ 2)
Result: A randomly generated expression (model)

$features \leftarrow$ choose $(d - n_{dummy})$ unique features;
// Initialize Expression
$expr \leftarrow 0$;
// Nonlinear Main Effects
$n'_{main\_nonlinear} \leftarrow pct_{nonlinear} \times |features|$;
// Number of terms
$n_{main\_nonlinear} \leftarrow \min(n'_{main\_nonlinear}, |features|)$;
// Each bin will on average contain $n'_{main\_nonlinear}/n_{main\_nonlinear}$ operators
$ops_{main\_nonlinear} \leftarrow$ place $n'_{main\_nonlinear}$ unary nonlinear operators into $n_{main\_nonlinear}$ bins;
// Cycle keeps track of the current element in a sequence, starting at the beginning if the previous element was at the end
$main_{features} \leftarrow$ cycle$(features)$;
for $i \in \{i \mid 1 \leq i \leq n_{main\_nonlinear}$ **do**
  // Get the next feature in the cycle
  $term \leftarrow$ next $main_{features}$;
  // Apply nonlinearities
  for $op \in ops_{main\_nonlinear}[i]$ **do**
    $term \leftarrow op(term)$;
  $expr \leftarrow expr + term$;
// Linear Main Effects
$n_{main\_linear} \leftarrow |features| - n_{main\_nonlinear}$;
for $i \in \{i \mid 1 \leq i \leq n_{main\_linear}\}$ **do**
  $feature \leftarrow$ next $main_{features}$;
  $expr \leftarrow expr + feature$;
// Nonlinear Interaction Effects
$n_{interact} \leftarrow \min\{pct_{interact} \times |features|, |features|\}$;
$n'_{interact\_nonlinear} \leftarrow pct_{nonlinear} \times n_{interact}$;
$interactions \leftarrow$ choose $n_{interact}$ unique feature pairs of size $order_{interact}$;
$n_{interact\_nonlinear} \leftarrow \min(n'_{interact\_nonlinear}, n_{interact})$;
// Each is a unary/binary nonlinear operator or binary linear operator. # of binary operators per effect $= order_{interact} - 1$
$ops_{interact\_nonlinear} \leftarrow$ place $n'_{interact\_nonlinear}$ (non)linear operators into $n_{interact\_nonlinear}$ bins;
$interact_{features} \leftarrow$ cycle$(interactions)$;
for $i \in \{i \mid 1 \leq i \leq n_{interact\_nonlinear}$ **do**
  $interaction \leftarrow$ cycle(next $interact_{features}$);
  $term \leftarrow$ next $interaction$;
  for $op \in ops_{interact\_nonlinear}[i]$ **do**
    if op is unary **then**
      $term \leftarrow op(term)$;
    else
      $feature \leftarrow$ next $interaction$;
      $term \leftarrow op(term, feature)$;
  $expr \leftarrow expr + term$;
// Continues on the following page...

```
// Linear Interaction Effects
n_interact_linear ← n_interact − n_interact_nonlinear;
for i ∈ {i | 1 ≤ i ≤ n_interact_linear} do
    interaction ← cycle(next interact_features);
    ops_interact_linear ← choose |interaction| − 1 linear non-additive binary operations;
    term ← next interaction;
    for op ∈ ops_interact_linear do
        feature ← next interaction;
        term ← op(term, feature);
    expr ← expr + feature;
return expr
```