On the Art and Science of Machine Learning Explanations

A Discussion with Practical Recommendations and a Use Case

Patrick Hall*

September 17, 2018

Abstract

This text discusses several explanatory methods that go beyond the error measurements and plots traditionally used to assess machine learning models. Some of the methods are tools of the trade while others are rigorously derived and backed by long-standing theory. The methods, decision tree surrogate models, individual conditional expectation (ICE) plots, local interpretable model-agnostic explanations (LIME), partial dependence plots, and Shapley explanations, vary in terms of scope, fidelity, and suitable application domain. Along with descriptions of these methods, this text presents real-world usage recommendations supported by a use case and in-depth software examples.

Key Words: Machine learning, interpretability, explanations, transparency, FATML, XAI.

1. Introduction

Interpretability of complex machine learning models is a multifaceted, complex, and still evolving subject. Others have defined key terms and put forward general motivations for better intepretability of machine learning models (and lamented a general lack of scientific rigor) [9], [11], [13], [18]. This applied text side-steps some looming, unsetttled intellectual matters to present viable practical methods for explaining the mechanisms and outputs of predictive models, typically supervised decision tree ensembles, for users who need to explain their work today.

Following Doshi-Velez and Kim, this discussion uses "the ability to explain or to present in understandable terms to a human," as the definition of *interpretable*. "When you can no longer keep asking why," will serve as the working definition for a *good explanation* of model mechanisms or predictions [11].

As in Figure 1, the presented explanatory methods help practitioners make random forests, GBMs, and other types of popular supervised machine learning models more interpretable by enabling post-hoc explanations that are suitable for:

- Facilitating regulatory compliance.
- Understanding or debugging model mechanisms and predictions.
- Preventing or debugging accidental or intentional discrimination by models.
- Preventing or debugging the malicious hacking or adversarial attack of models.

Detailed discussions of the explanatory methods begin below by defining notation. Then Sections 3 – 6 discuss explanatory methods and present recommendations for each

*H2O.ai, Mountain View, CA

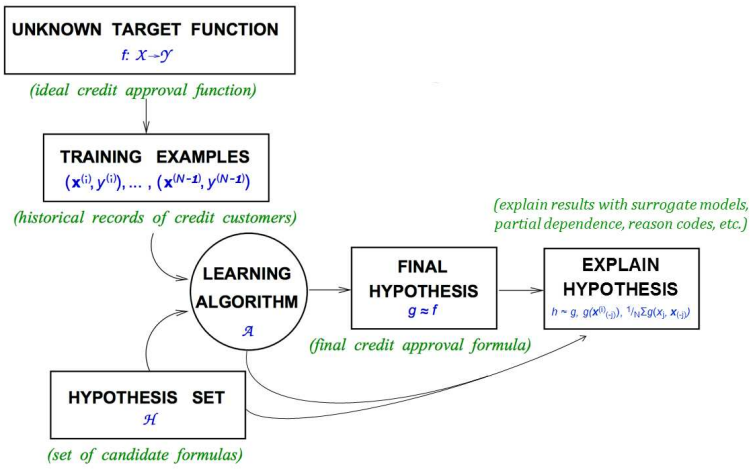

Figure 1: An augmented learning problem diagram in which several techniques create explanations for a credit scoring model. Adapted from *Learning From Data* [1].

method. Section 7 presents some general interpretability recommendations for practitioners. Section 8 applies some of the techniques and recommendations to the well-known UCI credit card dataset [17]. Section 9 discusses several additional interpretability subjects that are likely important for practitioners, and finally, Section 10 highlights software resources that accompany this text.

2. Notation

To facilitate technical descriptions of explanatory techniques, notation for input and output spaces, datasets, and models is defined.

2.1 Spaces

- Input features come from the set $\mathcal{X}$ contained in a P-dimensional input space, $\mathcal{X} \subset \mathbb{R}^P$.
- Known labels corresponding to instances of $\mathcal{X}$ come from the set $\mathcal{Y}$.
- Learned output responses come from the set $\hat{\mathcal{Y}}$.

2.2 Datasets

- The input dataset $\mathbf{X}$ is composed of observed instances of the set $\mathcal{X}$ with a corresponding dataset of labels $\mathbf{Y}$, observed instances of the set $\mathcal{Y}$.
- Each i-th observation of $\mathbf{X}$ is denoted as $\mathbf{x}^{(i)} = [x_0^{(i)}, x_1^{(i)}, \ldots, x_{P-1}^{(i)}]$, with corresponding i-th labels in $\mathbf{Y}, \mathbf{y}^{(i)}$, and corresponding predictions in $\hat{\mathbf{Y}}, \hat{\mathbf{y}}^{(i)}$.
- $\mathbf{X}$ and $\mathbf{Y}$ consists of N tuples of observations: $[(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathbf{x}^{(N-1)}, \mathbf{y}^{(N-1)})]$.
- Each j-th input column vector of $\mathbf{X}$ is denoted as $X_j = [x_j^{(0)}, x_j^{(1)}, \ldots, x_j^{(N-1)}]^T$.

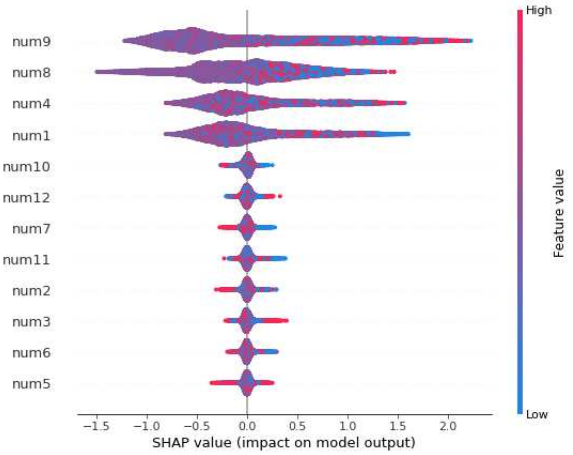

Figure 2: Shapley summary plot for known signal-generating function $f = \text{num}_1 * \text{num}_4 + |\text{num}_8| * \text{num}_9^2 + e$, and for learned GBM response function g_{GBM}.

2.3 Models

- A type of machine learning model g, selected from a hypothesis set $\mathcal{H}$, is trained to represent an unknown signal-generating function f observed as $\mathbf{X}$ with labels $\mathbf{Y}$ using a training algorithm $\mathcal{A}$: $\mathbf{X}, \mathbf{Y} \xrightarrow{\mathcal{A}} g$.
- g generates learned output responses on the input dataset $g(\mathbf{X}) = \hat{\mathbf{Y}}$, and on the general input space $g(\mathcal{X}) = \hat{\mathcal{Y}}$.
- The model to be explained is denoted as g.

3. Surrogate Decision Trees

The phrase *surrogate model* is used here to refer to a simple model, h, of a complex model, g. This type of model is referred to by various other names, such as *proxy* or *shadow* models and the process of training surrogate models is sometimes referred to as *model extraction* [8], [30], [5].

3.1 Description

Given a learned function g, a set of learned output responses $g(\mathbf{X}) = \hat{\mathbf{Y}}$, and a tree splitting and pruning approach $\mathcal{A}$, a global – or over all $\mathbf{X}$ – surrogate decision tree h_{tree} can be extracted such that $h_{\text{tree}}(\mathbf{X}) \approx g(\mathbf{X})$:

$$\mathbf{X}, g(\mathbf{X}) \xrightarrow{\mathcal{A}} h_{\text{tree}} \tag{1}$$

Decision trees can be represented as directed graphs where the relative positions of input features can provide insight into their importance and interactions [6]. This makes decision trees useful surrogate models. Input features that appear high and often in the directed graph representation of h_{tree} are assumed to have high importance in g. Input features directly above or below one-another in h_{tree} are assumed to have potential interactions in g. These relative relationships between input features in h_{tree} can be used to verify and analyze the feature importance, interactions, and predictions of g.

Figures 2 and 3 use simulated data to empirically demonstrate the desired relationships between input feature importance and interactions in the input space $\mathbf{X}$, the label space $f(\mathbf{X}) = \mathbf{Y}$, a GBM model to be explained g_{GBM}, and a decision tree surrogate h_{tree}. Data with a known signal-generating function depending on four input features with interactions and with eight noise features is simulated such that:

$$f = \text{num}_1 * \text{num}_4 + |\text{num}_8| * \text{num}_9^2 + e \tag{2}$$

g_{GBM} is trained: $\mathbf{X}, \mathbf{f}(\mathbf{X}) \xrightarrow{\mathcal{A}} g_{\text{GBM}}$ such that $g_{\text{GBM}} \approx f$. Then h_{tree} is extracted by $\mathbf{X}, g_{\text{GBM}}(\mathbf{X}) \xrightarrow{\mathcal{A}} h_{\text{tree}}$, such that $h_{\text{tree}}(\mathbf{X}) \approx g_{\text{GBM}}(\mathbf{X}) \approx f(\mathbf{X})$.

Figure 2 displays the local Shapley contribution values for an input feature's impact on each $g_{\text{GBM}}(\mathbf{X})$ prediction. Analyzing local Shapley values can be a more holistic and consistent feature importance metric than traditional single-value quantities [20]. Features are ordered from top to bottom by their mean absolute Shapley value across observations in Figure 2, and as expected, num_9 and num_8 tend to make the largest contributions to $g_{\text{GBM}}(\mathbf{X})$ followed by num_4 and num_1. Also as expected, noise features make minimal contributions to $g_{\text{GBM}}(\mathbf{X})$. Shapley values are discussed in detail in Section 6.

Figure 3 is a directed graph representation of h_{tree} that prominently displays the importance of input features num_9 and num_8 along with num_4 and num_1. Figure 3 also visually highlights the potential interactions between these inputs. URLs to the data and software used to generate Figures 2 and 3 are available in Section 10.

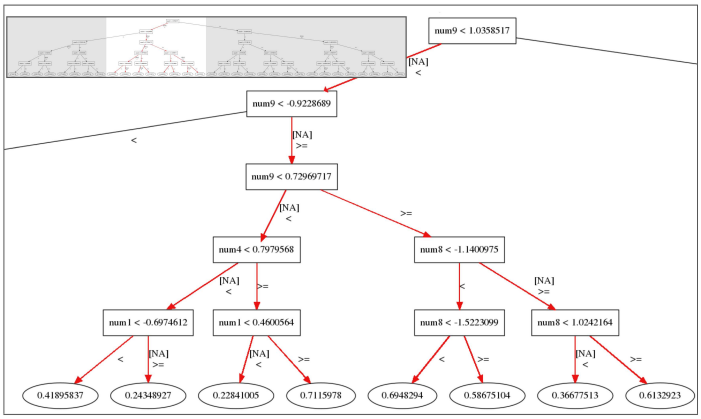

Figure 3: h_{tree} for previously defined known signal-generating function f and learned GBM response function g_{GBM}. An image of the entire h_{tree} directed graph is available in the supplementary materials described in Section 10.

3.2 Recommendations

- A shallow-depth h_{tree} displays a global, low-fidelity (e.g. approximate), high-interpretability flow chart of important features and interactions in g. Because there are few theoretical gaurantees that h_{tree} truly represents g, always use error measures to assess the trustworthiness of h_{tree}.

- Prescribed methods for training h_{tree} do exist [8] [5]. In practice, straightforward cross-validation approaches are often sufficient. Moreover, comparing cross-validated training error to traditional training error can give an indication of the stability of the single decision tree, h_{tree}.

- Hu et al. use local linear surrogate models, h_{GLM}, in h_{tree} leaf nodes to increase overall surrogate model fidelity while also retaining a high degree of interpretability [16].

4. Partial Dependence and Individual Conditional Expectation (ICE) plots

Partial dependence (PD) plots are a widely-used method for describing the average predictions of a complex model g across some partition of data $\mathbf{X}$ for some interesting input feature X_j [10]. Individual conditional expectation (ICE) plots are a newer method that describes the local behavior of g for a single instance $\mathbf{x} \in \mathcal{X}$. Partial dependence and ICE can be combined in the same plot to identify interactions modeled by g and to create a holistic portrait of the predictions of a complex model for some X_j [12].

4.1 Description

Following Friedman et al. a single feature $X_j \in \mathbf{X}$ and its complement set $\mathbf{X}_{(-j)} \in \mathbf{X}$ (where $X_j \cup \mathbf{X}_{(-j)} = \mathbf{X}$) is considered. $\text{PD}(X_j, g)$ for a given feature X_j is estimated as the average output of the learned function $g(\mathbf{X})$ when all the components of X_j are set to a constant $x \in \mathcal{X}$ and $\mathbf{X}_{(-j)}$ is left unchanged. $\text{ICE}(x_j, \mathbf{x}, g)$ for a given instance $\mathbf{x}$ and feature x_j is estimated as the output of $g(\mathbf{x})$ when x_j is set to a constant $x \in \mathcal{X}$ and all other features $\mathbf{x} \in \mathbf{X}_{(-j)}$ are left untouched. Partial dependence and ICE curves are usually plotted over some set of constants $x \in \mathcal{X}$.

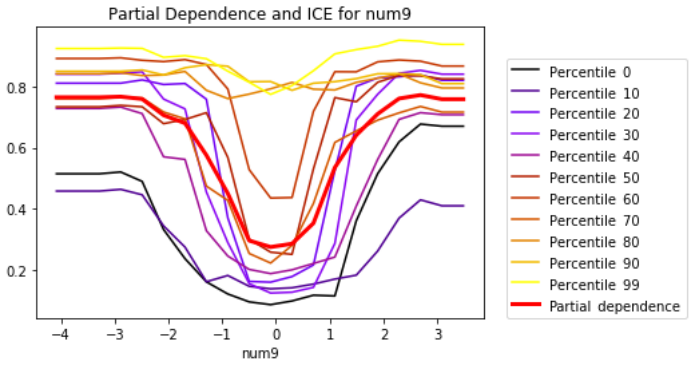

Figure 4: Partial dependence and ICE curves for previously defined known signal-generating function f, learned GBM response function g_{GBM}, and important input feature num_9.

As in Section 3, simulated data is used to highlight desirable characteristics of partial dependence and ICE plots. In Figure 4 partial dependence and ICE at the minimum, maximum, and each decile of $g_{\text{GBM}}(\mathbf{X})$ are plotted. The known quadratic behavior of num_9 is plainly visible, except for high value predictions, the 80^{th} percentiles of $g_{\text{GBM}}(\mathbf{X})$ and above and for $\sim -1 < \text{num}_9 <\sim 1$. When partial dependence and ICE curves diverge, this often points to an interaction that is being averaged out of the partial dependence. Given the form of Equation 2, there is a known interaction between num_9 and num_8. Combining the information from partial dependence and ICE plots with h_{tree} can help elucidate more detailed information about modeled interactions in g. For the simulated example, h_{tree} shows an interaction between num_9 and num_8 and additional modeled interactions between num_9, num_4, and num_1 for $\sim -0.92 \leq \text{num}_9 <\sim 1.04$. URLs to the data and software used to generate Figure 4 are available in Section 10.

4.2 Recommendations

- Combining h_{tree} with partial dependence and ICE curves is a convenient method for detecting, confirming, and understanding important interactions in g.
- As monotonicity is often a desired trait for interpretable models, partial dependence and ICE plots can be used to verify the monotonicity of g on average and across percentiles of $g(\mathbf{X})$ w.r.t. some input feature X_j.

5. Local Interpretable Model-agnostic Explanations (LIME)

Global and local scope are key concepts in explaining machine learning models and predictions. Section 3 presents decision trees as a global – or over all $\mathbf{X}$ – surrogate model. As learned response functions, g, can be complex, simple global surrogate models can sometimes be too approximate to be trustworthy. LIME attempts to create more representative explanations by fitting a local surrogate model, h, in the local region of some observation of interest $\mathbf{x} \in \mathcal{X}$. Both h and local regions can be defined to suit the needs of users.

5.1 Description

Ribeiro et al. specifies LIME for some observation $\mathbf{x} \in \mathcal{X}$ as:

$$\underset{h \in \mathcal{H}}{\arg\min}\, \mathcal{L}(g, h, \pi_{\mathbf{X}}) + \Omega(h) \tag{3}$$

where h is an interpretable surrogate model of g, often a linear model h_{GLM}, $\pi_{\mathbf{X}}$ is a weighting function over the domain of g, and $\Omega(h)$ limits the complexity of h [23]. Following Ribeiro et al. h_{GLM} is often trained by:

$$\mathbf{X}', g(\mathbf{X}') \xrightarrow{\mathcal{A}_{\text{LASSO}}} h_{\text{GLM}} \tag{4}$$

where $\mathbf{X}'$ is sampled from $\mathcal{X}$, $\pi_{\mathbf{X}}$ weighs $\mathbf{X}'$ samples by their Euclidean similarity to $\mathbf{x}$ to enforce locality, local feature contributions are estimated as the product of h_{GLM} coefficients and their associated observed values $\beta_j x_j$, and $\Omega(h)$ is defined as a LASSO, or L1, penalty on h_{GLM} coefficients inducing sparsity in h_{GLM}.

Figure 5 displays estimated local feature contribution values for the same g_{GBM} and simulated $\mathbf{X}$ with known signal-generating function f used in previous sections. To increase the nonlinear capacity of the three h_{GLM} models, information from the Shapley summary plot in Figure 2 is used to select inputs to discretize before training each h_{GLM}: $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9. Table 1 contains prediction and fit information for g_{GBM} and h_{GLM}. This is critical information for analyzing LIMEs.

Table 1: g_{GBM} and h_{GLM} predictions and h_{GLM} intercepts and fit measurements for the h_{GLM} models trained to explain $g_{\text{GBM}}(\mathbf{x}^{(i)})$ at the 10th, median, and 90th percentiles of previously defined $g_{\text{GBM}}(\mathbf{X})$ and known signal-generating function f.

$g_{\text{GBM}}(\mathbf{X})$ Percentile	$g_{\text{GBM}}(\mathbf{x})$ Prediction	$h_{GLM}(\mathbf{x})$ Prediction	h_{GLM} Intercept	h_{GLM} R^2
10th	0.16	0.13	0.53	0.72
Median	0.30	0.47	0.70	0.57
90th	0.82	0.86	0.76	0.40

Table 1 shows that LIME is not necessarily locally accurate, meaning that the predictions of $h_{GLM}(\mathbf{x})$ are not always equal to the prediction of $g_{\text{GBM}}(\mathbf{x})$. Moreover, the three

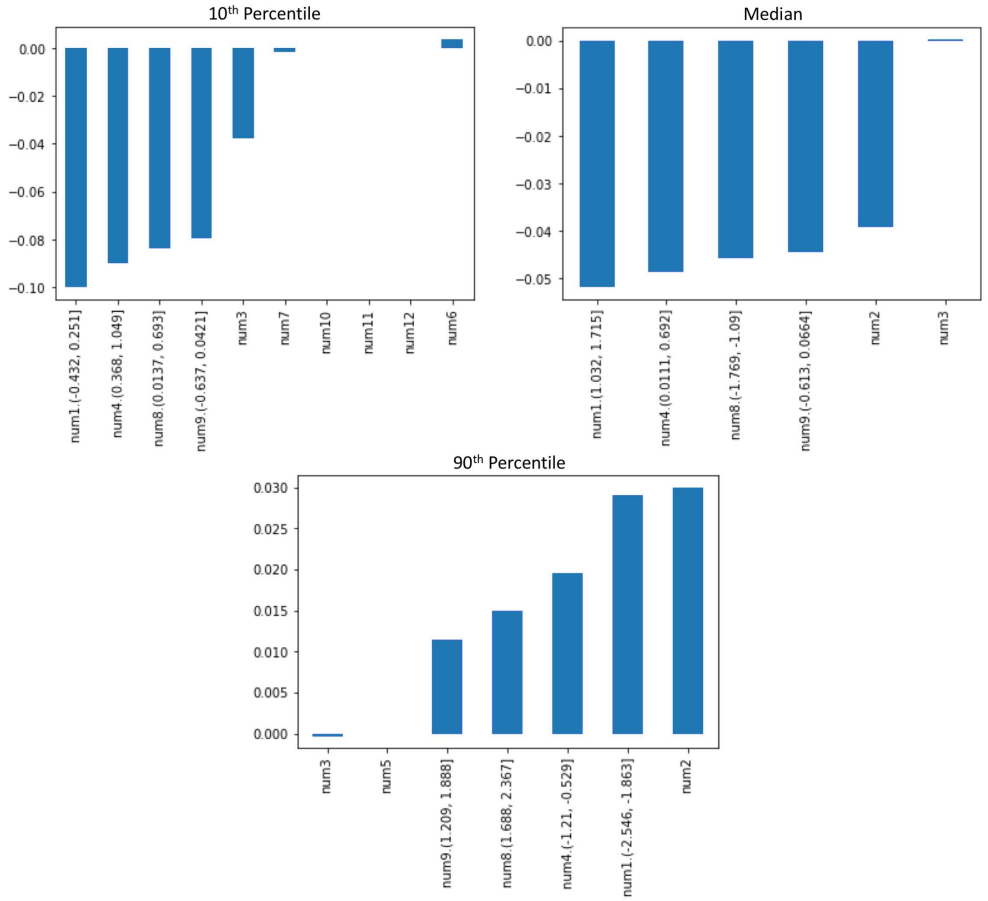

Figure 5: Sparse, low-fidelity local feature contributions found using LIME at three percentiles of $g_{\text{GBM}}(\mathbf{X})$ for known signal-generating function $f = \text{num}_1 * \text{num}_4 + |\text{num}_8| * \text{num}_9^2 + e$.

h_{GLM} models do not necessarily explain all of the variance of g_{GBM} predictions in the local regions around the three $\mathbf{x}^{(i)}$ of interest. h_{GLM} intercepts are also displayed because local feature contribution values, $\beta_j x_j^{(i)}$, are offsets from the local h_{GLM} intercepts.

An immediately noticeable characteristic of the estimated local contributions in Figure 5 is their sparsity. LASSO input feature selection drives some h_{GLM} β_j coefficients to zero so that some $\beta_j x_j^{(i)}$ local feature contributions are also zero. For the 10$^{\text{th}}$ percentile $g_{\text{GBM}}(\mathbf{X})$ prediction, the local h_{GLM} R^2 is adequate and the LIME values appear parsimonious with reasonable expectations. The contributions from discretized $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 outweigh all other noise feature contributions and the $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 contributions are all negative, as expected for the relatively low value of $g_{\text{GBM}}(\mathbf{x})$.

For the median prediction of $g_{\text{GBM}}(\mathbf{X})$, it could be expected that some estimated contributions for $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 should be positive and others should be negative. However, all local feature contributions are negative due to the relatively high value of the h_{GLM} intercept at the median percentile of $g_{\text{GBM}}(\mathbf{X})$. Because the h_{GLM} intercept is quite large compared to the $g_{\text{GBM}}(\mathbf{x}^{(i)})$ prediction, it is not alarming that all the $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 contributions are negative offsets w.r.t. the local h_{GLM} intercept value. For the median $g_{\text{GBM}}(\mathbf{X})$ prediction, h_{GLM} also estimates that the noise feature num_2 has a fairly large contribution and the local h_{GLM} R^2 is probably less than adequate

to generate fully trustworthy explanations.

For the 90$^{\text{th}}$ percentile of $g_{\text{GBM}}(\mathbf{X})$ predictions, the local contributions for $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 are positive as expected for the relatively high value of $g_{\text{GBM}}(\mathbf{x^{(i)}})$, but the local h_{GLM} R^2 is somewhat poor and the noise feature $num2$ has the highest local feature contribtion. This large attribution to the noise feature num2 could stem from problems in the LIME procedure or in the fit of g_{GBM} to f. Further investigation, or model debugging, is conducted in Section 6.

Generally the LIMEs in Section 5 would be considered to be sparse or high-interpretability but also low-fidelity explanations. This is not always the case with LIME and the fit of some h_{GLM} to a local region around some $g(\mathbf{x})$ will vary in accuracy. URLs to the data and software used to generate Table 1 and Figure 5 are available in Section 10.

5.2 Recommendations

- Always use fit measures to assess the trustworthiness of LIMEs.
- Local feature contribution values are often offsets from a local h_{GLM} intercept. Note that this intercept can sometimes account for the most important local phenomena. Each LIME feature contribution can be interpreted as the difference in $h(\mathbf{x})$ and some local offset, often β_0, associated with some feature x_j.
- Some LIME methods can be difficult to deploy for explaining predictions in real-time. Consider highly deployable variants for real-time applications [15], [16].
- Always investigate local h_{GLM} intercept values. Generated LIME samples can contain large proportions of out-of-domain data that can lead to unrealistic intercept values.
- To increase the fidelity of LIMEs, try LIME on discretized input features and on manually constructed interactions. Use h_{tree} to construct potential interaction terms.
- Use cross-validation to estimate standard deviations or even confidence intervals for local feature contribution values.
- When relying only on local linear models, note that LIME can fail to create acceptable explanations, particularly in the presence of extreme nonlinearity or high-degree interactions. Other types of local models with model-specific explanatory mechanisms, such as decision trees or neural networks, can be used in these cases.

6. Tree Shap

Shapley explanations, including tree shap and even certain implementations of LIME, are a class of additive, consistent local feature contribution measures with long-standing theoretical support [20]. Shapley explanations are the only possible locally accurate and consistent feature contribution values, meaning that Shapley explanation values for input features always sum to $g(\mathbf{x})$ and that Shapley explanation values can never decrease for some x_j when g is changed such that x_j truly makes a stronger contribution to $g(\mathbf{x})$ [20].

6.1 Description

For some observation $\mathbf{x} \in \mathcal{X}$, Shapley explanations take the form:

$$g(\mathbf{x}) = \phi_0 + \sum_{j=0}^{j=\mathcal{P}-1} \phi_j \mathbf{z}_j \tag{5}$$

In Equation 5, $\mathbf{z} \in \{0,1\}^{\mathcal{P}}$ is a binary representation of $\mathbf{x}$ where 0 indicates missingness. Each ϕ_j is the local feature contribution value associated with x_j and ϕ_0 is the average of $g(\mathbf{X})$.

Shapley values can be estimated in different ways. Tree shap is a specific implementation of Shapley explanations. It does not rely on surrogate models. Both tree shap and a related technique known as *treeinterpreter* rely instead on traversing internal tree structures to estimate the impact of each x_j for some $g(\mathbf{x})$ of interest [19], [25].

$$\phi_j = \sum_{S \subseteq \mathcal{P} \setminus \{j\}} \frac{|S|!(\mathcal{P} - |S| - 1)!}{\mathcal{P}!} [g_x(S \cup \{j\}) - g_x(S)] \tag{6}$$

Unlike treeinterpreter and as displayed in Equation 6, tree shap and other Shapley approaches estimate ϕ_j as the difference between the model prediction on a subset of features S without X_j, $g_x(S)$, and the model prediction with X_j and S, $g_x(S \cup \{j\})$, summed and weighed appropriately across all subsets S of $\mathcal{P}$ that do not contain X_j, $S \subseteq \mathcal{P} \setminus \{j\}$. (Here g_x incorporates the mapping between $\mathbf{x}$ and the binary vector $\mathbf{z}$.) Since trained decision tree response functions model complex dependencies between input features, removing different subsets of input features helps elucidate the true impact of removing x_j from $g(\mathbf{x})$.

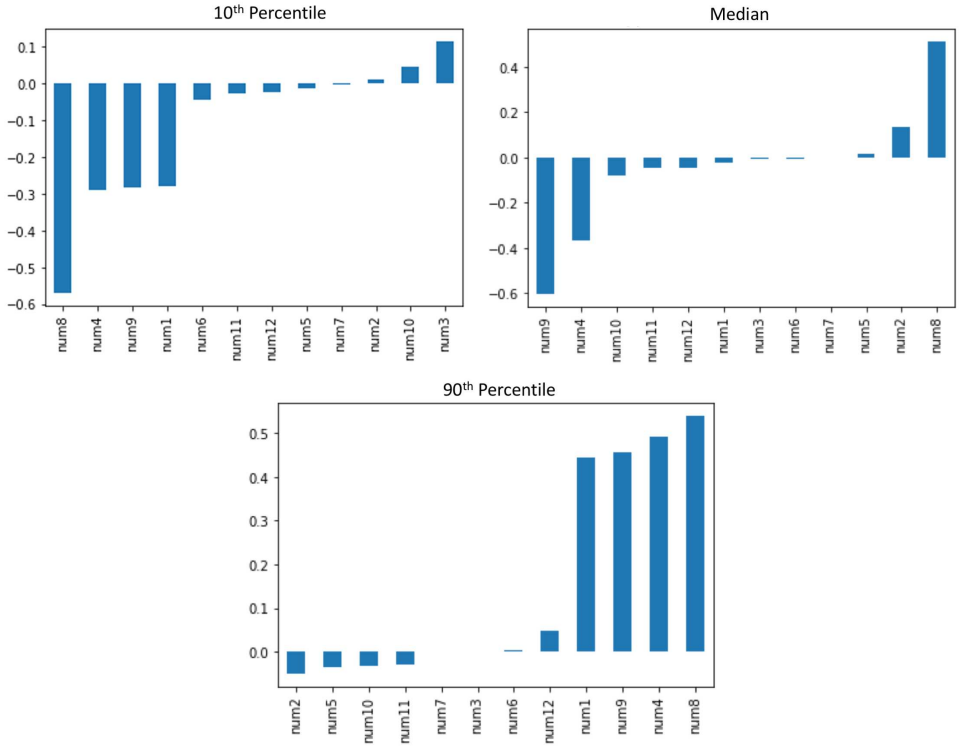

Figure 6: Complete, consistent local feature contributions found using tree shap at three percentiles of $g_{\text{GBM}}(\mathbf{X})$ and for known signal generating function $f = \text{num}_1 * \text{num}_4 + |\text{num}_8| * \text{num}_9^2 + e$.

Simulated data is used again to illustrate the utility of tree shap. Shapley explanations are estimated at the 10th, median, and 90th percentiles of $g_{\text{GBM}}(\mathbf{X})$ for simulated $\mathbf{X}$ with known signal-generating function f. Results are presented in Figure 6. In contrast to the LIME explanations in Figure 5, the Shapley explanations are complete, giving a numeric local contribution value for each non-missing input feature. At the 10th percentile of $g_{\text{GBM}}(\mathbf{X})$ predictions, all feature contributions for $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 are negative as expected for this relatively low value of $g_{\text{GBM}}(\mathbf{X})$ and their contributions obviously outweigh those of noise features.

For the median prediction of $g_{\text{GBM}}(\mathbf{X})$, the Shapley explanations are somewhat aligned with the expectation of a split between positive and negative contributions. $\text{num}_1, \text{num}_4$, and num_9 are negative and the contribution for num_8 is positive. Like the LIME explanations at this percentile in Figure 5, the noise feature num_2 has a relatively high contribution, higher than that of num_1, likely indicating that g_{GBM} is over-emphasizing num_2 in the local region around the median prediction.

As expected at the 90th percentile of $g_{\text{GBM}}(\mathbf{X})$ all contributions from $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 are positive and much larger than the contributions from noise features. Unlike the LIME explanations at the 90th percentile of $g_{\text{GBM}}(\mathbf{X})$ in Figure 5, tree shap estimates only a small contribution from num_2. This discrepancy may reveal a pair-wise linear correlation between num_2 and $g_{\text{GBM}}(\mathbf{X})$ in the local region around the 90th percentile of $g_{\text{GBM}}(\mathbf{X})$ that fails to represent the true form of $g_{\text{GBM}}(\mathbf{X})$ in this region, which can be highly nonlinear and incorporate high-degree interactions. Partial dependence and ICE for num_2 and two-dimensional partial dependence between num_2 and $\text{num}_1, \text{num}_4, \text{num}_8$ and num_9 could be used to further investigate the form of $g_{\text{GBM}}(\mathbf{X})$ w.r.t. num_2, along with model debugging techniques discussed briefly in Section 9. URLs to the data and software used to generate Figure 6 are available in Section 10.

6.2 Recommendations

- Tree shap is ideal for estimating high-fidelity, consistent, and complete explanations of decision tree and decision tree ensemble models, perhaps even in regulated applications to generate regulator-mandated reason codes (also known as turn-down codes or adverse action codes).

- Because tree shap explanations are offsets from a global intercept, each ϕ_j can be interpreted as the difference in $g(\mathbf{x})$ and the average of $g(\mathbf{X})$ associated with some input feature x_j [21].

- Currently treeinterpreter may be inappropriate for some GBM models. Treeinterpreter is locally accurate for some decision tree and random forest models, but is known to be inconsistent like all other feature importance methods aside from Shapley approaches [19]. In experiments available in the supplemental materials of this text, treeinterpreter is seen to be locally inaccurate for some XGBoost GBM models.

7. General Recommendations

The following recommendations apply to several or all of the described explanatory techniques or to the practice of applied interpretable machine learning in general.

- Less complex models are typically easier to explain. Section 9 contains information about directly interpretable white-box machine learning models.

- Monotonicity is often a desirable characteristic in interpretable models. White-box, monotonically constrained XGBoost models along with the explanatory techniques described in this text are a direct and open source way to train and explain an interpretable machine learning model. A monotonically constrained XGBoost GBM is trained and explained in Section 8.

- Several explanatory techniques are usually required to create good explanations for any given complex model. Users should apply a combination global and local and low- and high-fidelity explanatory techniques to a machine learning model and seek consistent results across multiple explanatory techniques. Simpler low-fidelity or sparse explanations can be used to understand more accurate, and sometimes more sophisticated, high-fidelity explanations.

- Methods relying on surrogate models or generated data are sometimes unpalatable to users. Users sometimes *need* to understand *their* model on *their* data.

- Surrogate models can provide low-fidelity explanations for an entire machine learning pipeline in the original feature space if g is defined to include feature extraction or feature engineering steps.

- Both understanding and trust are crucial to interpretability. The discussed explanatory techniques should engender a greater understanding of model mechanisms and predictions. But can a model be trusted to perform as expected on unseen data? Its predictions probably do not extrapolate linearly outside of the training, validation, or test data domains. Always conduct sensitivity analysis on your trained machine learning model to understand how it will behave on out-of-domain data.

- Consider production deployment of explanatory methods carefully. Currently, the deployment of some open source software packages is not straightforward, especially for the generation of explanations on new data in real-time.

8. Credit Card Data Use Case

Some of the discussed explanatory techniques and recommendations will now be applied to a basic credit scoring problem using a monotonically constrained XGBoost binomial classifier and the UCI credit card dataset [17]. Referring back to Figure 1, a training set X and associated labels Y will be used to train a GBM with decision tree base learners, selected based on domain knowledge from many other types of hypotheses models $\mathcal{H}$, using a monotonic splitting strategy with gradient boosting as the training algorithm $\mathcal{A}_{\text{mono}}$, to learn a final hypothesis model g_{mono}, that approximates the true signal generating function f governing credit default in $\mathcal{X}$ and $\mathcal{Y}$ such that $g_{\text{mono}} \sim f$:

$$\mathbf{X}, \mathbf{Y} \xrightarrow{\mathcal{A}_{\text{mono}}} g_{\text{mono}} \tag{7}$$

g_{mono} is globally explainable with aggregated local Shapley values, decision tree surrogate models h_{tree}, partial dependence, and ICE plots. Additionally each prediction made by g_{mono} can be explained using local Shapley explanations.

To begin, Pearson correlation between g_{mono} inputs and the target, `default payment next month`, are calculated and stored. All other features except for the observation identifier, `ID`, are used as g_{mono} inputs. Then 30% of the credit card dataset observations are randomly partitioned into a labeled validation set. Pearson correlations are used to define monotonicity constraints w.r.t. each input feature. Input features with a positive

correlation to the target are constrained to a monotonically increasing relationship with the target under g_{mono}. Input features with a negative correlation to the target are constrained to a monotonically decreasing relationship. Along with the monotonicity constraints, the non-default hyperparameter settings used to train g_{mono} are presented in Table 2.

Table 2: g_{mono} hyperparameters for the UCI credit card dataset. Adequate hyperparameters were found by Cartesian grid search.

Hyperparameter	Value
`eta`	0.08
`subsample`	0.9
`colsample_bytree`	0.9
`maxdepth`	15

A maximum of 1000 iterations were used to train g_{mono}, with early stopping triggered after 50 iterations without validation AUC improvement. This configuration led to a final validation AUC of 0.781 after only 100 iterations.

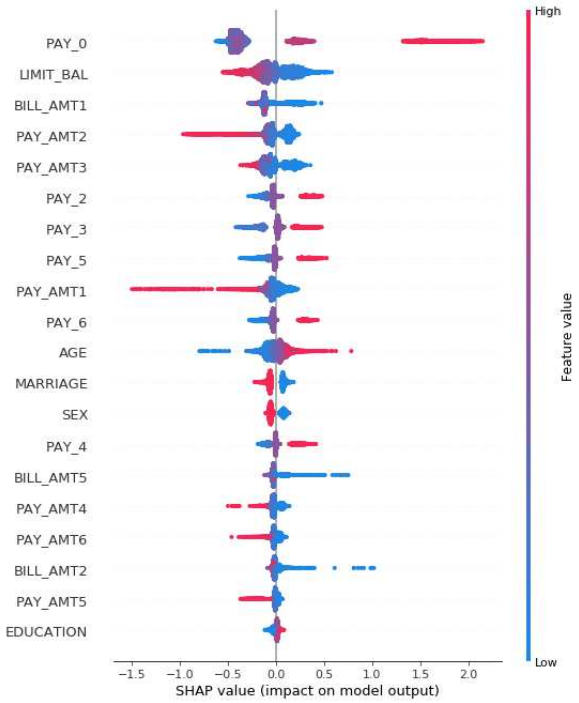

Figure 7: Shapley summary plot for g_{mono} in a 30% validation set randomly sampled from the UCI credit card dataset.

The global feature importance of g_{mono} evaluated in the validation set and ranked by mean absolute Shapley value is displayed in Figure 7. `PAY_0` – a customer's most recent repayment status, `LIMIT_BAL` – a customer's credit limit, and `BILL_AMT1` – a customer's most recent bill amount are the globally most important features, which aligns with reasonable expectations and basic domain knowledge. (A real-world credit scoring application would be unlikely to use `LIMIT_BAL` as an input feature because this feature could cause target leakage. `LIMIT_BAL` is used in this small data example to improve g_{mono} fit.) The monotonic relationship between each input feature and g_{mono} output is also visible in Figure

7. Numeric Shapley explanation values appear to increase only as an input feature value increases as for PAY_0, or vice versa, say for LIMIT_BAL.

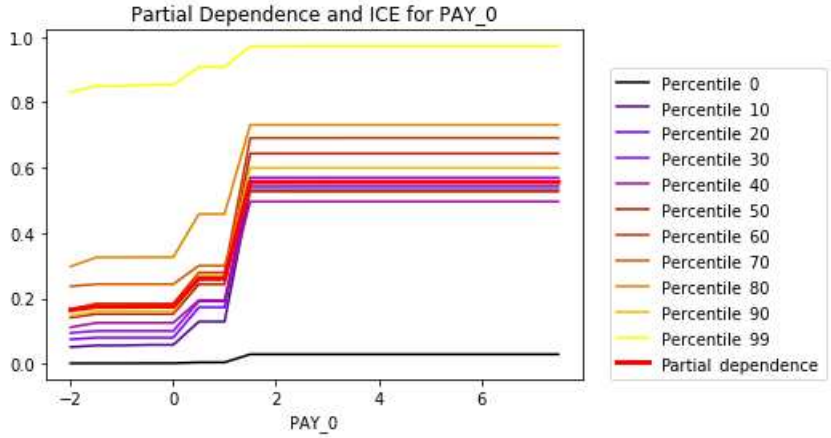

Figure 8: Partial dependence and ICE curves for learned GBM response function g_{mono}, and important input feature PAY_0.

Partial dependence and ICE for g_{mono} and the important input feature PAY_0 verify the monotonic increasing behavior of g_{mono} w.r.t. to PAY_0. For several percentiles of predicted probabilities and on average, the output of g_{mono} is low for PAY_0 values -2 – 1 then increases dramatically. PAY_0 values of -2 – 1 are associated with on-time or 1 month late payments. A large increase in predicted probability of default occurs at PAY_0 = 2 and predicted probabilities plateau after PAY_0 = 2. The lowest and highest predicted probability customers do not display the same precipitous jump in predicted probability at PAY_0 = 2. If this dissimilar prediction behavior is related to interactions with other input features, that may be evident in a surrogate decision tree model.

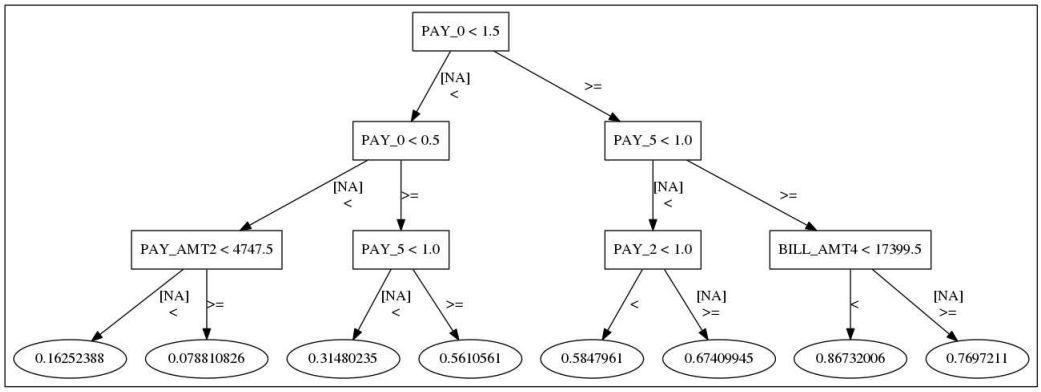

Figure 9: h_{tree} for g_{mono} in a 30% validation set randomly sampled from the UCI credit card dataset. An image of a depth-five h_{tree} directed graph is available in the supplementary materials described in Section 10.

To continue explaining g_{mono}, a simple depth-three h_{tree} model is trained to represent $g_{\text{mono}}(\mathbf{X})$ in the validation set. h_{tree} is displayed in Figure 9. h_{tree} has a mean R^2 across three random folds in the validation set of 0.86 with a standard deviation of 0.0011 and a mean RMSE across the same folds of 0.08 with a standard deviation of 0.0003, indicating

h_{tree} is likely accurate and stable enough to be a helpful explanatory tool. The global importance of PAY_0 and the increase in $g_{\text{mono}}(\mathbf{x})$ associated with PAY_0 = 2 is reflected in the simple h_{tree} model, along with several potentially important interactions between input features. For instance the lowest predicted probabilities from h_{tree} occur when a customer's most recent repayment status, PAY_0, is less than 0.5 and their second most recent payment amount, PAY_AMT2, is greater than or equal to NT$ 4747.5. The highest predicted probabilities from h_{tree} occur when PAY_0 $\geq$ 1.5, a customers fifth most recent repayment status, PAY_5, is 1 or more months late, and when a customer's fourth most recent bill amount, BILL_AMT4 is less than NT$ 17399.5. In this simple depth-three h_{tree} model, it appears that an interaction between PAY_0 and PAY_AMT2 may be leading to the very low probability of default predictions displayed in Figure 8, while interactions between PAY_0, PAY_5, and BILL_AMT4 are potentially associated with the highest predicted probabilities. A more complex and accurate depth-five h_{tree} model is available in the supplemental materials described in Section 10 and it presents greater detail regarding the interactions and decision paths that could lead to the modeled behavior for the lowest and highest probability of default customers.

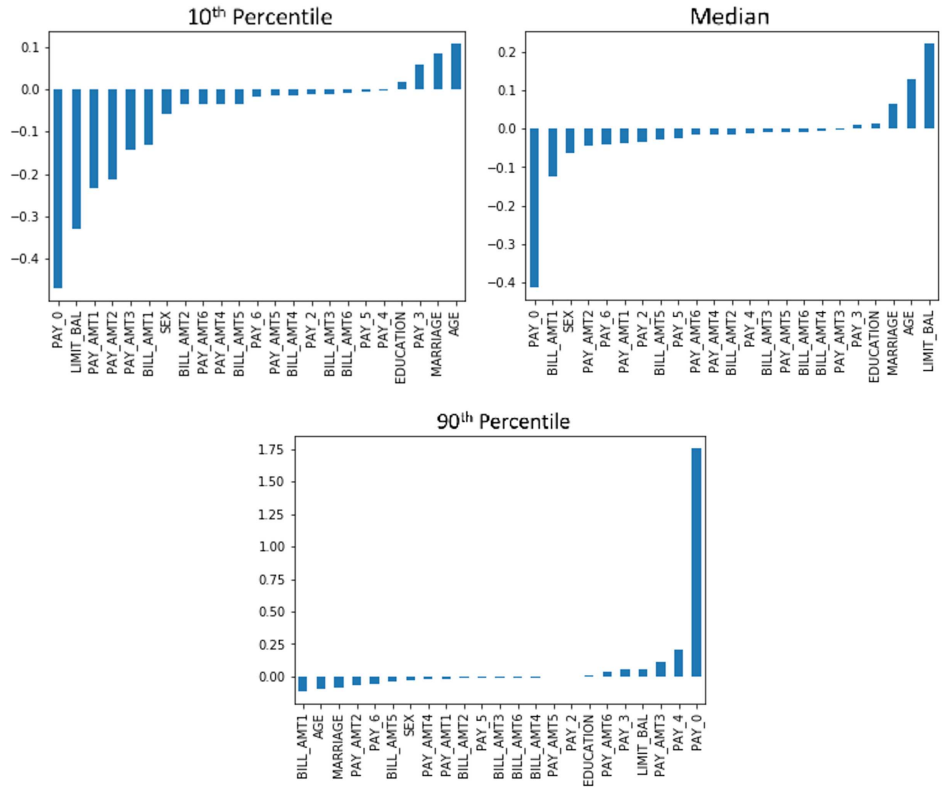

Figure 10: Complete, consistent local feature contributions found using tree shap at three percentiles of $g_{\text{mono}}(\mathbf{X})$ in a 30% validation set randomly sampled from the UCI credit card dataset.

Figure 10 displays local Shapley explanation values for three customers at the 10th, median, and 90th percentiles of $g_{\text{mono}}(\mathbf{X})$ in the validation set. The plots in Figure 10 are representative of the local Shapley explanations that could be generated for any $g_{\text{mono}}(\mathbf{x}), \mathbf{x} \in \mathcal{X}$. The values presented in Figure 10 are aligned with the general expectation that Shapley contributions will increase for increasing values of $g_{\text{mono}}(\mathbf{x})$. Reason codes to justify decisions based on $g_{\text{mono}}(\mathbf{x})$ predictions can also be generated for arbitrary $g_{\text{mono}}(\mathbf{x}), \mathbf{x} \in \mathcal{X}$ using

local Shapley explanation values and the values of input features in $\mathbf{x}$. For the customer at the 90$^{\text{th}}$ percentile of $g_{\text{mono}}(\mathbf{X})$ the likely top three reason codes to justify declining further credit are:

- Most recent payment is 2 months delayed.
- Fourth most recent payment is 2 months delayed.
- Third most recent payment amount is NT$ 0.

Analysis for an operational, mission-critical machine learning model would likely involve further investigation of partial dependence and ICE plots and perhaps deeper analysis of h_{tree} models following Hu et al [16]. Analysis would also probably continue on to diagnostic, or *model debugging*, and fairness techniques such as:

- **Disparate impact analysis**: to uncover any unfairness in model predictions or errors across demographic segments.
- **Residual analysis**: to check the fundamental assumptions of the model against relevant data partitions and to investigate outliers or observations exerting undue influence on $g(\mathbf{X})$.
- **Sensitivity analysis**: to explicitly test the trustworthiness of model predictions on simulated out-of-domain data or in other simulated scenarios of interest.

A successful explanatory and diagnostic analysis must also include remediating any discovered issues and documenting all findings. Examples of more detailed analyses along with the URLs to the data and software used to generate Figures 7 – 10 are available in Section 10.

9. Suggested Reading

As stated in the introduction, this text focuses on a fairly narrow but practical sub-discipline of machine learning interpretability. As interpretability truly is a diverse subject with many other practically useful areas of study, additional subjects are suggested for further reading.

9.1 White-box Models

The application of post-hoc explanatory techniques is convenient for previously existing machine learning models, workflows, or pipelines. However, a more direct approach may be to train an interpretable white-box machine learning model which may or may not require additional post-hoc explanatory analysis. Monotonic XGBoost is an excellent option to evaluate because the software is open source, readily available, easily installable and deployable, and highly scalable [7]. Acclaimed work by the Rudin group at Duke University is also likely of interest to many users. They have developed several types of rule-based models [3], [31], linear model variants [28], and many other novel algorithms suitable for use in high stakes, mission-critical prediction and decision-making scenarios.

9.2 Explainable Neural Networks (xNNs)

Often considered the least transparent of black-box models, recent work in xNN implementation and explaining artificial neural network (ANN) predictions may render that notion of ANNs obsolete. Many of the breakthroughs in ANN explanation stem from the straightforward calculation of accurate derivatives of the trained ANN response function w.r.t. to input features made possible by the proliferation of deep learning toolkits such as tensorflow [22]. These derivatives allow for the disaggregation of the trained ANN response function prediction, $g_{ANN}(\mathbf{X})$, into input feature contributions for any observation in the domain of $\mathcal{X}$. Popular techniques have names like DeepLIFT and integrated gradients [26], [27], [2]. Explaining ANN predictions is impactful for at least two major reasons. While most users will be familiar with the wide-spread use of ANNs in pattern recognition, they are also used for more traditional data mining applications such as fraud detection, and even for regulated applications such as credit scoring [14]. Moreover, ANNs can now be used as accurate and explainable surrogate models, potentially increasing the fidelity of both global and local surrogate model techniques. For an excellent discussion of xNNs in a practical setting see *Explainable Neural Networks based on Additive Index Models* by the Wells Fargo Corporate Model Risk group [29].

9.3 Fairness

Fairness is yet another important facet of interpretability, and an admirable goal for any machine learning project whose outcomes will affect human lives. Traditional checks for fairness include assessing the average prediction, accuracy, and error across demographic segments. Today the study of fairness in machine learning is widening and progressing rapidly. Users who would like to stay abreast of developments in the fairness space should follow the free online book by leading researchers, *Fairness and Machine Learning* [4]. The book's website currently includes references to other fairness materials. Users may also be interested in the broader organization for fairness, accountability, and transparency in machine learning or FATML. FATML maintains a list of pertinent scholarship on their website: `https://www.fatml.org/`.

9.4 Model Debugging

As alluded to in previous sections, the techniques in this text enable practitioners to *explain* and hopefully understand complex models. Trust is a related and somewhat orthogonal concept in interpretability, because models can certainly be explained and not trusted, and conversely, be trusted and not explainable. The growing field of model debugging empowers users to trust their complex model predictions and to diagnose and treat any discovered undesirable behaviors. While residual and sensitivity analysis typically play a role in model debugging exercises, readers are encouraged to investigate newer methods such as *anchors* and burgeoning work in adversarial examples [24], [32].

10. Supplementary Materials and Software Resources

To make the discussed results useful and reproducible for practitioners, several online supporting materials and software resources are freely available.

- Supplementary materials and any corrections or updates to this text are available at: `https://github.com/jphall663/jsm_2018_paper`.

- Simulated data experiments, including experiments on random data, and the UCI credit card dataset use case are available at: https://github.com/h2oai/mli-resources/tree/master/lime_shap_treeint_compare. General instructions for using these resources, including a Dockerfile which builds the complete runtime environment with all dependencies, are available here: https://github.com/h2oai/mli-resources.
- In-depth example explanatory use cases for the UCI credit card dataset are available at: https://github.com/jphall663/interpretable_machine_learning_with_python.
- A curated list of interpretability software is available at: https://github.com/jphall663/awesome-machine-learning-interpretability.

11. Acknowledgements

The author wishes to thank Sri Ambati, Arno Candel, Mark Chan, Doug Deloy, Lauren DiPerna, Mateusz Dymczyk, Navdeep Gill, Megan and Michal Kurka, Lingyao Meng, Mattias Müller, Wen Phan and other makers past and present at H2O.ai who turned explainability into a software application and learned the lessons discussed in this text along with me. The author thanks Leland Wilkinson at H2O.ai and the University of Illinois at Chicago (UIC) for his continued mentorship and guidance.

References

[1] Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. ***Learning from Data***. AMLBook, New York, 2012. URL: https://work.caltech.edu/textbook.html.

[2] Marco Ancona, Enea Ceolini, Cengiz Oztireli, and Markus Gross. Towards Better Understanding of Gradient-based Attribution Methods for Deep Neural Networks. In *6th International Conference on Learning Representations (ICLR 2018)*, 2018. URL: https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/249929/Flow_ICLR_2018.pdf.

[3] Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning Certifiably Optimal Rule Lists for Categorical Data. *Journal of Machine Learning Research*, 18(234):1–78, 2018. URL: https://corels.eecs.harvard.edu/.

[4] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2018. URL: http://www.fairmlbook.org.

[5] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. Interpreting Blackbox Models via Model Extraction. *arXiv preprint arXiv:1705.08504*, 2017. URL: https://arxiv.org/pdf/1705.08504.pdf.

[6] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Routledge, 1984.

[7] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM, 2016. URL: https://arxiv.org/pdf/1603.02754.pdf.

[8] Mark W. Craven and Jude W. Shavlik. Extracting Tree-structured Representations of Trained Networks. *Advances in Neural Information Processing Systems*, 1996. URL: http://papers.nips.cc/paper/1152-extracting-tree-structured-representations-of-trained-networks.pdf.

[9] Finale Doshi-Velez and Been Kim. Towards a Rigorous Science of Interpretable Machine Learning. *arXiv preprint arXiv:1702.08608*, 2017. URL: https://arxiv.org/pdf/1702.08608.pdf.

[10] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. ***The Elements of Statistical Learning***. Springer, New York, 2001. URL: https://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII_print12.pdf.

[11] Leilani H Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining Explanations: An Approach to Evaluating Interpretability of Machine Learning. *arXiv preprint arXiv:1806.00069*, 2018. URL: https://arxiv.org/pdf/1806.00069.pdf.

[12] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation. *Journal of Computational and Graphical Statistics*, 24(1), 2015. URL: https://arxiv.org/pdf/1309.6392.pdf.

[13] Riccardo Guidotti, Anna Monreale, Franco Turini, Dino Pedreschi, and Fosca Giannotti. A Survey of Methods for Explaining Black Box Models. *arXiv preprint arXiv:1802.01933*, 2018. URL: https://arxiv.org/pdf/1802.01933.pdf.

[14] Patrick Hall and Navdeep Gill. *An Introduction to Machine Learning Interpretability*. O'Reilly Media, 2018. URL: https://www.safaribooksonline.com/library/view/an-introduction-to/9781492033158/.

[15] Patrick Hall, Navdeep Gill, Megan Kurka, and Wen Phan. *Machine Learning Interpretability with H2O Driverless AI*. H2O.ai, 2017. URL: http://docs.h2o.ai/driverless-ai/latest-stable/docs/booklets/MLIBooklet.pdf.

[16] Linwei Hu, Jie Chen, Vijayan N. Nair, and Agus Sudjianto. Locally Interpretable Models and Effects Based on Supervised Partitioning (LIME-SUP). *arXiv preprint arXiv:1806.00663*, 2018. URL: https://arxiv.org/ftp/arxiv/papers/1806/1806.00663.pdf.

[17] M. Lichman. UCI Machine Learning Repository, 2013. URL: http://archive.ics.uci.edu/ml.

[18] Zachary C. Lipton. The Mythos of Model Interpretability. *arXiv preprint arXiv:1606.03490*, 2016. URL: https://arxiv.org/pdf/1606.03490.pdf.

[19] Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent Individualized Feature Attribution for Tree Ensembles. *arXiv preprint arXiv:1802.03888*, 2018. URL: https://arxiv.org/pdf/1706.06060.pdf.

[20] Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. URL: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

[21] Christoph Molnar. *Interpretable Machine Learning*. christophm.github.io/interpretable-ml-book, 2018. URL: https://christophm.github.io/interpretable-ml-book/.

[22] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations. *arXiv preprint arXiv:1711.10561*, 2017. URL: https://arxiv.org/pdf/1711.10561.pdf.

[23] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why Should I Trust You?: Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM, 2016. URL: http://www.kdd.org/kdd2016/papers/files/rfp0573-ribeiroA.pdf.

[24] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-Precision Model-agnostic Explanations. In *AAAI Conference on Artificial Intelligence*, 2018. URL: https://homes.cs.washington.edu/~marcotcr/aaai18.pdf.

[25] Ando Saabas. Interpreting Random Forests, 2014. URL: http://blog.datadive.net/interpreting-random-forests/.

[26] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning Important Features Through Propagating Activation Differences. *arXiv preprint arXiv:1704.02685*, 2017. URL: https://arxiv.org/pdf/1704.02685.pdf.

[27] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic Attribution for Deep Networks. *arXiv preprint arXiv:1703.01365*, 2017. URL: https://arxiv.org/pdf/1703.01365.pdf.

[28] Berk Ustun and Cynthia Rudin. Supersparse Linear Integer Models for Optimized Medical Scoring Systems. *Machine Learning*, 102(3):349–391, 2016. URL: https://users.cs.duke.edu/~cynthia/docs/UstunTrRuAAAI13.pdf.

[29] Joel Vaughan, Agus Sudjianto, Erind Brahimi, Jie Chen, and Vijayan N Nair. Explainable Neural Networks Based on Additive Index Models. *arXiv preprint arXiv:1806.01933*, 2018. URL: https://arxiv.org/pdf/1806.01933.pdf.

[30] Mike Williams et al. *Interpretability*. Fast Forward Labs, 2017. URL: https://www.cloudera.com/products/fast-forward-labs-research.html.

[31] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable Bayesian Rule Lists. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017. URL: https://arxiv.org/pdf/1602.08610.pdf.

[32] Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating Natural Adversarial Examples. *arXiv preprint arXiv:1710.11342*, 2017. URL: https://arxiv.org/pdf/1710.11342.pdf.