A Practical Discussion of Machine Learning Explanations with Recommendations and a Use Case

Patrick Hall
patrick.hall@h2o.ai
H2O.ai
Mountain View, CA

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 public, in-depth software examples for reproducibility.

CCS CONCEPTS

• **Human-centered computing** → *Visual analytics*; • **Computing methodologies** → *Supervised learning by classification*; *Supervised learning by regression*; *Learning linear models*; *Classification and regression trees*; *Ensemble methods.*

KEYWORDS

Machine learning, interpretability, explanations, transparency, FATML, XAI

ACM Reference Format:
Patrick Hall. 2019. A Practical Discussion of Machine Learning Explanations with Recommendations and a Use Case. In *KDD '19: 25th ACM SIGKDD Conference On Knowledge Discovery and Data Mining, August 04–08, 2019, Anchorage, AK*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The subject of interpretable machine learning is both multifaceted and evolving. Others have previously defined key terms, put forward general motivations for the broader goal of better interpretability, and advocated for stronger scientific rigor for the burgeoning field [5], [8], [10], [14], [22]. 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 [8]. 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.

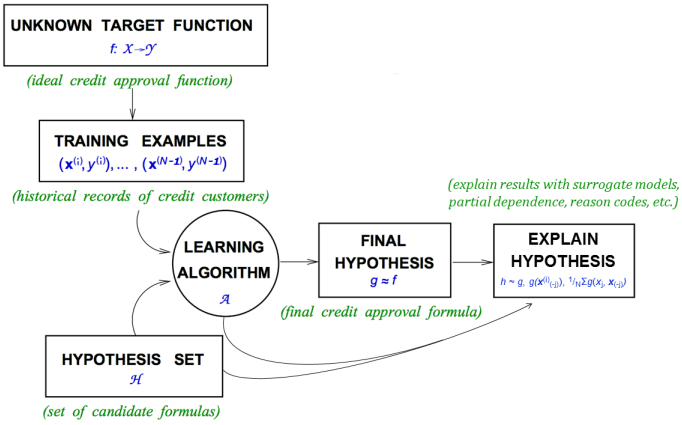

Figure 1: An augmented learning problem diagram in which several techniques create explanations for a credit scoring model. Adapted from Figure 1.2 of the open textbook *Learning From Data* [1].

Some have criticized the sub-discipline of explanatory methods like those described herein [19], but model explanations and the documentation they enable are an important and often mandatory aspect of predictive modeling in industries like financial services and healthcare[1]. Moreover explanations can, and likely should, be used along with interpretable models and fairness assessments or

KDD '19, August 04–08, 2019, Anchorage, AK

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1] In the U.S., explanations and model documentation may be required under the Civil Rights Acts of 1964 and 1991, the Americans with Disabilities Act, the Genetic Information Nondiscrimination Act, the Health Insurance Portability and Accountability Act, the Equal Credit Opportunity Act, the Fair Credit Reporting Act, the Fair Housing Act, Federal Reserve SR 11-7, and the European Union (EU) Greater Data Privacy Regulation (GDPR) Article 22 [23].

bias remediation to further enhance transparency and accountability in high-stakes, life- or mission-critical machine learning workflows.

The primary discussion of this text will focus on augmenting popular machine learning models with explanations for various purposes, including meeting regulatory requirements. The use case will then focus on blending explanations with a constrained, interpretable variant of a popular machine learning model. Discussions of the explanatory methods begin below by defining notation. Then Sections 3 – 6 describe explanatory methods and present usage recommendations for each 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 [13]. Finally, Section A 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, $\mathcal{X} \subset \mathbb{R}^P$. An arbitrary, potentially unobserved, or future instance of $\mathcal{X}$ is denoted $\mathbf{x}$, $\mathbf{x} \in \mathcal{X}$.
- 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 $\mathbf{Y}$ consist 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 $X_j = [x_j^{(0)}, x_j^{(1)}, \ldots, x_j^{(N-1)}]^T$.

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 $\mathcal{A}$: $\mathbf{X}, \mathbf{Y} \xrightarrow{\mathcal{A}} g$, such that $g \approx f$.
- 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* [4], [23], [2].

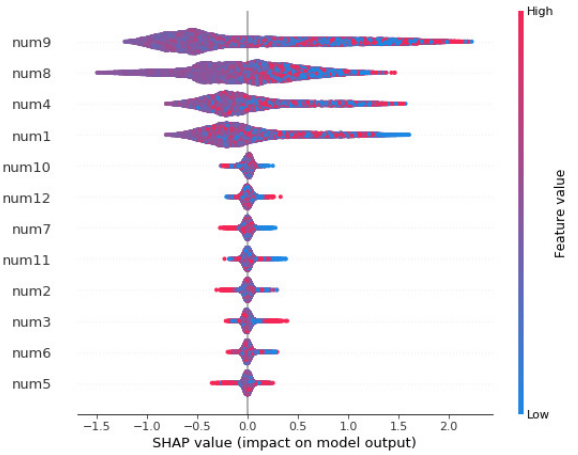

Figure 2: Globally consistent Shapley summary plot for known signal-generating function $f(\mathbf{X}) = X_{\mathbf{num1}} * X_{\mathbf{num4}} + |X_{\mathbf{num8}}| * X^2_{\mathbf{num9}} + e$ and for learned GBM response function $g_{\mathbf{GBM}}$ in a validation dataset.

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 [3]. 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 numeric input features with interactions and with eight noise features is simulated such that:

$$f(\mathbf{X}) = X_{\text{num1}} * X_{\text{num4}} + |X_{\text{num8}}| * X^2_{\text{num9}} + e \tag{2}$$

where e signifies the injection of random noise in the form of label switching for roughly 15% of the training and validation observations.

g_{GBM} is then trained: $\mathbf{X}, \mathbf{f}(\mathbf{X}) \xrightarrow{\mathcal{A}} g_{\text{GBM}}$, such that $g_{\text{GBM}} \approx f$, and 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, which accurately measure a feature's impact on each $g_{\text{GBM}}(\mathbf{x})$ prediction, for observations in the validation data. Analyzing local Shapley values can be a more holistic and consistent feature importance metric than traditional single-value quantities [16]. Features are ordered from top to bottom by their mean absolute Shapley value across observations in Figure 2, and as expected, X_{num9} and X_{num8} tend to make the largest contributions to $g_{\text{GBM}}(\mathbf{X})$ followed by X_{num4} and X_{num1}. Also as expected, noise features make minimal contributions to $g_{\text{GBM}}(\mathbf{X})$. Expected values are calculated by training g_{GBM} with no validation set on f with no error term and are available in materials listed in Section A. 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 X_{num9} and X_{num8} along with X_{num4} and X_{num1}. 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 A.

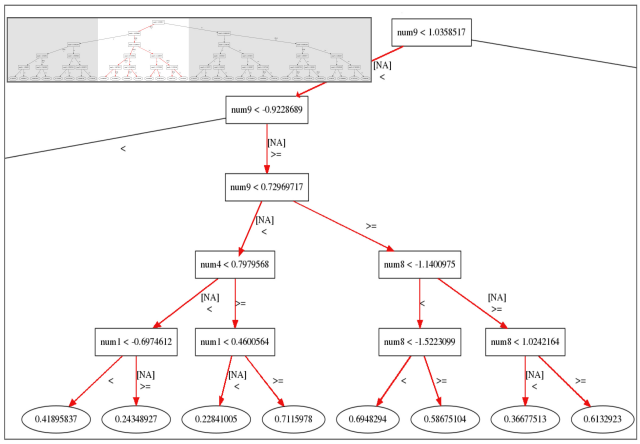

Figure 3: h_{tree} for previously defined known signal-generating function f and learned GBM response function g_{GBM} in a validation dataset. An image of the entire h_{tree} directed graph is available in the supplementary materials described in Section A.

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 guarantees that h_{tree} truly represents g, always use error measures to assess the trustworthiness of h_{tree}, e.g. RMSE, MAPE, R^2.
- Prescribed methods for training h_{tree} do exist [4] [2]. 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 [12].

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 [7]. 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 compensate for known weaknesses of partial dependence, to identify interactions modeled by g, and to create a holistic portrait of the predictions of a complex model for some X_j [9].

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 observations 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 $x \in \mathcal{X}$.

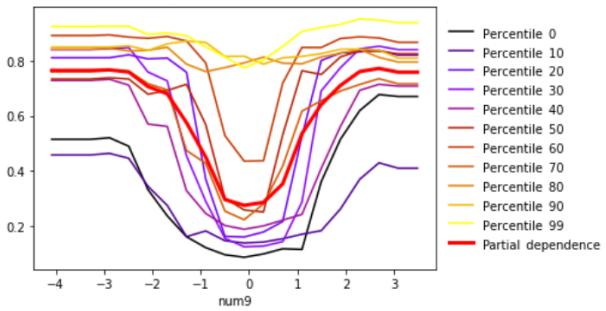

Figure 4: Partial dependence and ICE curves for previously defined known signal-generating function f, learned GBM response function g_{GBM}, and important input feature X_{num9} in a validation dataset.

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 X_{num9} is plainly visible, except for high value predictions, the 80^{th} percentiles of $g_{\text{GBM}}(\mathbf{X})$ and above and for $\sim -1 < X_{\text{num9}} <\sim 1$. 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 X_{num9} and X_{num8}. 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} confirms an interaction between X_{num9} and X_{num8} and shows additional modeled interactions between X_{num9}, X_{num4}, and X_{num1} for $\sim -0.92 \leq X_{\text{num9}} <\sim 1.04$. URLs to the data and software used to generate Figure 4 are available in Section A.

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.
- Partial dependence can be misleading in the presence of strong correlation or interactions. Plotting partial dependence with ICE provides a direct visual indication as to whether the displayed partial dependence is credibly representative of individual predictions.
- Comparing partial dependence and ICE plots with a histogram for the input feature of interest can give a basic qualitative measure of epistemic uncertainty by enabling visual discovery of $g(\mathbf{x})$ values that are based on only small amounts of training data.

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 [18]. 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}: X_{num1}, X_{num4}, X_{num8} and X_{num9}. 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 R^2 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 in a validation dataset.

$g_{\text{GBM}}(\mathbf{X})$ Percentile	$g_{\text{GBM}}(\mathbf{x}^{(i)})$ Prediction	$h_{GLM}(\mathbf{x}^{(i)})$ Prediction	h_{GLM} Intercept	h_{GLM} R^2
10^{th}	0.16	0.13	0.53	0.72
Median	0.30	0.47	0.70	0.57
90^{th}	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 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^{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 $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} outweigh all other noise feature contributions and the $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} 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 $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} 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 $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} 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 X_{num2} 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^{th} percentile of $g_{\text{GBM}}(\mathbf{X})$ predictions, the local contributions for $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} 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 X_{num2} has the highest local feature contribution. This large attribution to the noise feature X_{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.

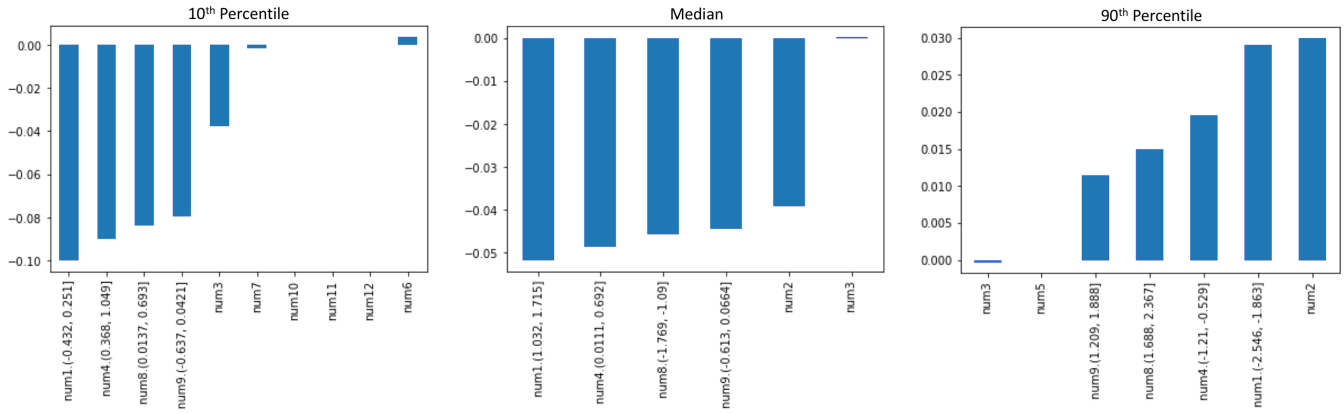

Figure 5: Sparse, low-fidelity local feature contributions found using LIME at three percentiles of $g_{\text{GBM}}(\mathbf{X})$ for known signal-generating function $f(\mathbf{X}) = X_{\mathbf{num1}} * X_{\mathbf{num4}} + |X_{\mathbf{num8}}| * X^2_{\mathbf{num9}} + e$ in a validation dataset.

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 A.

5.2 Recommendations

- Always use fit measures to assess the trustworthiness of LIMEs, e.g. RMSE, MAPE, R^2.
- 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 [11], [12].
- 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. Analyze h_{tree} to construct potential interaction terms.
- Use cross-validation to estimate standard deviations or even confidence intervals for local feature contribution values.
- Poor fit or inaccuracy of local linear models is itself informative, often indicating extreme nonlinearity or high-degree interactions. However, explanations from such local linear models are likely unacceptable and other types of local models with model-specific explanatory mechanisms, such as decision trees or neural networks [21], can be used in these cases to generate high-fidelity explanations.

6 TREE SHAP

Shapley explanations, including tree SHAP (SHapley Additive exPlanations) and even certain implementations of LIME, are a class of additive, consistent local feature contribution measures with long-standing theoretical support [16]. Shapley explanations are the only possible locally accurate and globally 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 $g(\mathbf{x})$ [16].

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 [15], [20].

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

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 $g(\mathbf{x})$.

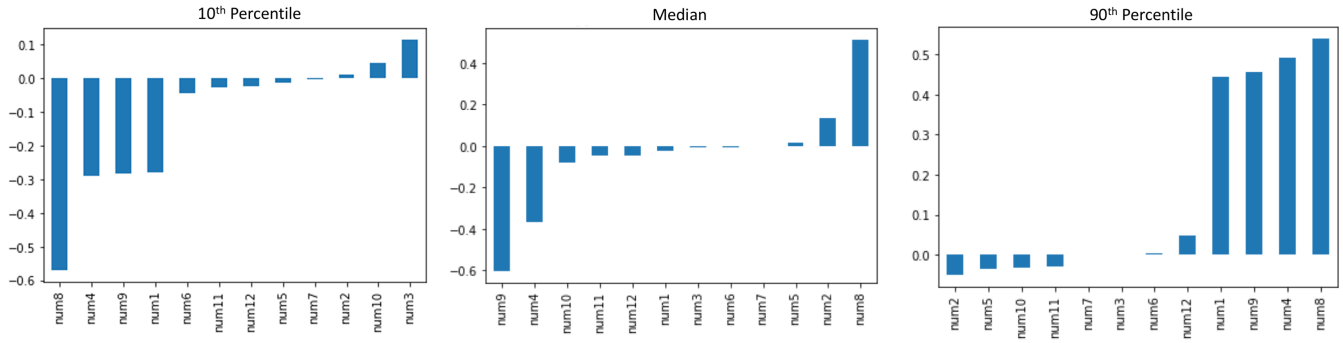

Figure 6: Complete, locally accurate feature contributions found using tree SHAP at three percentiles of $g_{\mathbf{GBM}}(\mathbf{X})$ and for known signal generating function $f(\mathbf{X}) = X_{\mathbf{num1}} * X_{\mathbf{num4}} + |X_{\mathbf{num8}}| * X^2_{\mathbf{num9}} + e$.

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 $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} are negative as expected for this relatively low value of $g_{\text{GBM}}(\mathbf{X})$ and their contributions obviously outweigh those of the 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. $X_{\text{num1}}, X_{\text{num4}}$, and X_{num9} are negative and the contribution for X_{num8} is positive. Like the LIME explanations at this percentile in Figure 5, the noise feature X_{num2} has a relatively high contribution, higher than that of X_{num1}, likely indicating that g_{GBM} is over-emphasizing X_{num2} in the local region around the median prediction.

As expected at the 90th percentile of $g_{\text{GBM}}(\mathbf{X})$ all contributions from $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} 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 X_{num2}. This discrepancy may reveal a spurious pair-wise linear correlation between X_{num2} 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 X_{num2} and two-dimensional partial dependence between X_{num2} and $X_{\text{num1}}, X_{\text{num4}}, X_{\text{num8}}$ and X_{num9} could be used to further investigate the form of $g_{\text{GBM}}(\mathbf{X})$ w.r.t. X_{num2}. URLs to the data and software used to generate Figure 6 are available in Section A.

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 [17].
- 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 [15]. 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 and several types of machine learning models are directly interpretable, e.g. scalable Bayesian rule lists [24]. For maximum transparency in life- or mission-critical decision support scenarios, consider pairing interpretable white-box machine learning models with post-hoc explanatory techniques.

- Monotonicity is often a desirable characteristic in interpretable models. (Of course it should not be enforced when a modeled relationship is known to be non-monotonic.) 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.
- Conceptually, both understanding and trust are crucial to interpretability. The discussed explanatory techniques should engender a greater understanding of model mechanisms and predictions. When appropriate conduct sensitivity, residual, and disparate impact analysis on your trained machine learning model to foster greater trust in its behavior.
- 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 [13]. Referring back to Figure 1, a training set $\mathbf{X}$ and associated labels $\mathbf{Y}$ 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 $g_{\text{mono}} \approx f$:

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

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

Thirty percent of the credit card dataset observations are randomly partitioned into a labeled validation set and Pearson correlation between g_{mono} inputs and the target, `default payment next month`, are calculated and stored. All features except for the target and the observation identifier, `ID`, are used as g_{mono} inputs. The signs of the stored Pearson correlations are used to define the direction monotonicity constraints w.r.t. each input feature. (Features with small magnitude correlations or known non-monotonic behavior could also be left unconstrained.) Additional non-default hyperparameter settings used to train g_{mono} are presented in Table 2. 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.

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

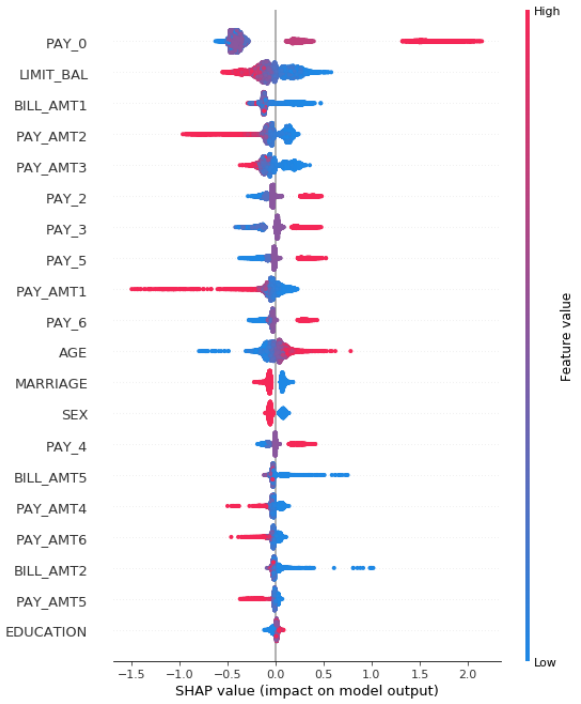

Figure 7: Globally consistent Shapley summary plot for g_{mono} in a 30% validation set randomly sampled from the UCI credit card dataset.

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 most important features globally, 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`.

Partial dependence and ICE for g_{mono} and the important input feature `PAY_0` verify the monotonic increasing behavior of g_{mono}

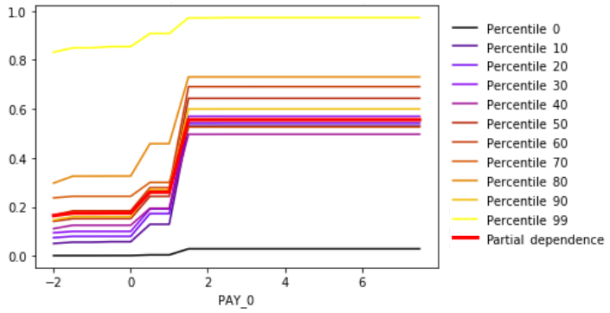

Figure 8: Partial dependence and ICE curves for learned GBM response function g_{mono} and important input feature `PAY_0` in a 30% validation set randomly sampled from the UCI credit card dataset.

w.r.t. `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.

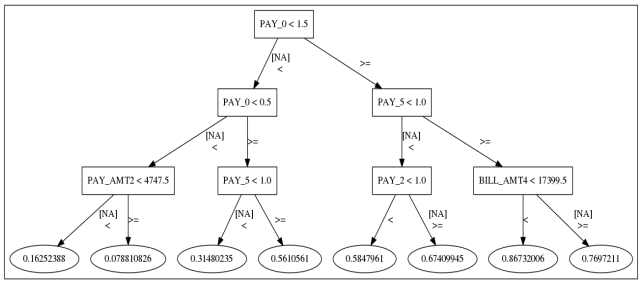

Figure 9: h_{tree} for g_{mono} in a 30% validation set randomly sampled from the UCI credit card dataset. An image of a depth-five h_{tree} directed graph is available in the supplementary materials described in Section A.

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 A 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 displays local Shapley explanation values for three customers at the 10$^{\text{th}}$, median, and 90$^{\text{th}}$ 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 $g_{\text{mono}}(\mathbf{x})$ using local Shapley explanation values and the values of input features in $\mathbf{x}$. Observed values of $\mathbf{X}$ are available in the supplementary materials presented in Section A. 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 [12]. Analysis would also probably continue on to diagnostic, or *model debugging*, and fairness techniques such as:

- **Disparate impact analysis and remediation**: to uncover and remediate any disparate impact in model predictions or errors across demographic segments [6].
- **Residual analysis**: to check the fundamental assumptions of the model against relevant data partitions and investigate outliers or observations exerting undue influence on g.
- **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 A.

9 CONCLUSION

This text aspires to hasten adoption of explanatory techniques and also to bring interpretable models and model debugging and fairness methodologies to the attention of thoughtful practitioners. Future work will analyze and test combinations of interpretable models and explanatory, model debugging, and fairness techniques

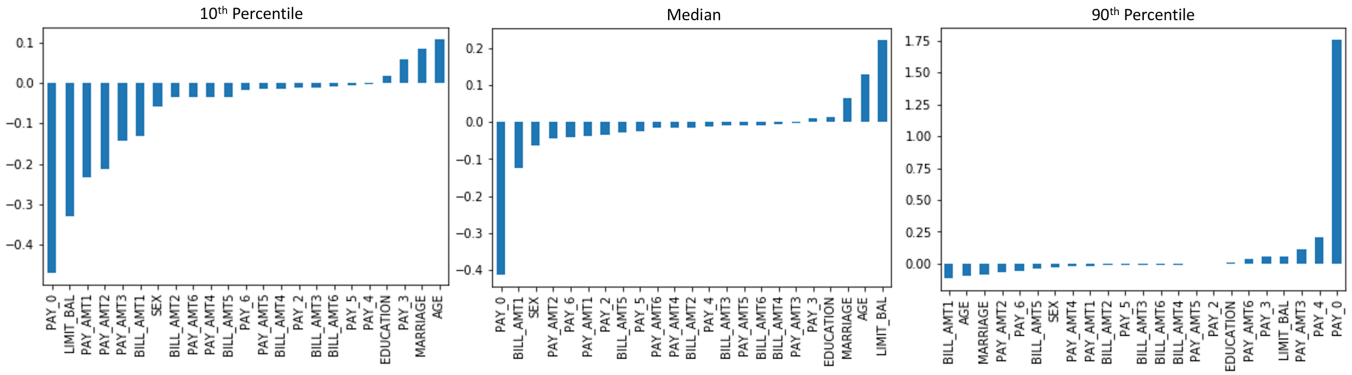

Figure 10: Complete, locally accurate feature contributions found using tree SHAP at three percentiles of $g_{\text{mono}}(\mathbf{X})$ in a 30% validation set randomly sampled from the UCI credit card dataset.

in the context of creating accurate and transparent systems for life- or mission-critical decision support applications.

10 ACKNOWLEDGMENTS

The author wishes to thank makers past and present at H2O.ai, and also thanks Mike Williams of Cloudera Fast Forward Labs for his review of this text.

REFERENCES

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

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

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

[4] Mark W. Craven and Jude W. Shavlik. 1996. 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.

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

[6] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 259–268. https://arxiv.org/pdf/1412.3756.pdf.

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

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

[9] Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. 2015. 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.

[10] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A Survey of Methods for Explaining Black Box Models. *ACM Computing Surveys (CSUR)* 51, 5 (2018), 93. URL: https://arxiv.org/pdf/1802.01933.pdf.

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

[12] Linwei Hu, Jie Chen, Vijayan N. Nair, and Agus Sudjianto. 2018. 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.

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

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

[15] Scott M. Lundberg, Gabriel G. Erion, and Su-In Lee. 2017. Consistent Individualized Feature Attribution for Tree Ensembles. In *Proceedings of the 2017 ICML Workshop on Human Interpretability in Machine Learning (WHI 2017)*, Been Kim, Dmitry M. Malioutov, Kush R. Varshney, and Adrian Weller (Eds.). ICML WHI 2017, 15–21. URL: https://openreview.net/pdf?id=ByTKSo-m-.

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

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

[18] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. 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*. ACM, 1135–1144. URL: http://www.kdd.org/kdd2016/papers/files/rfp0573-ribeiroA.pdf.

[19] Cynthia Rudin. 2018. Please Stop Explaining Black Box Models for High Stakes Decisions. *arXiv preprint arXiv:1811.10154* (2018). URL: https://arxiv.org/pdf/1811.10154.pdf.

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

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

[22] Adrian Weller. 2017. Challenges for Transparency. In *Proceedings of the 2017 ICML Workshop on Human Interpretability in Machine Learning (WHI 2017)*, Been Kim, Dmitry M. Malioutov, Kush R. Varshney, and Adrian Weller (Eds.). ICML WHI 2017, 55–62. URL: https://openreview.net/pdf?id=SJR9L5MQ-.

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

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

A ONLINE SOFTWARE RESOURCES

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

- Simulated data experiments, including additional experiments on random data, supplementary figures, and the UCI credit card dataset use case are available at:

  https://github.com/h2oai/mli-resources/tree/master/lime_shap_treeint_compare/README.md.

- 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 disparate impact, explanatory, and model debugging 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.