Interpretable Machine Learning

Wen Phan, Lauren DiPerna, Patrick Hall, and Navdeep Gill

Outline

Foundations

Introduction & Purpose

- **Goal:** Ability to interpret complex "black box" models, deemed uninterpretable for decades.
- **Solution:** Some approaches that increase a complex model's transparency, accountability, and fairness are the following:
  - Decision tree surrogate models [Craven and Shavlik, 1996; Bastani, Kim, and Bastani, 2017]
  - Partial dependence plots [Friedman, Hastie, and Tibshirani, 2001]
  - Individual conditional expectation (ICE) plots [Goldstein et al., 2015]
  - Random forest feature importance [Friedman, Hastie, and Tibshirani, 2001]
  - Leave-one-covariate-out (LOCO) local feature importance [Lei et al., 2017]
  - Local Interpretable Model-agnostic Explanations (LIME) [Ribeiro, Singh, and Guestrin, 2016]
  - Shapley Explanations [Lundberg and Lee, 2017; Lundberg, Erion, and Lee, 2018]

Introduction & Purpose

- The following slides will detail the mathematics behind each of these approaches and techniques.

Notation & Preliminaries

- **Spaces**.
  - The input features come from a set $\mathcal{X}$ contained in a P-dimensional input space (i.e. $\mathcal{X} \subset \mathbb{R}^P$).
  - The output responses come from a set $\mathcal{Y}$ contained in a C-dimensional output space (i.e. $\mathcal{Y} \subset \mathbb{R}^C$).

- **Dataset**. A dataset $\mathbf{D}$ consists of N tuples of observations: $[(\mathbf{x}^{(0)}, \mathbf{y}^{(0)}), (\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathbf{x}^{(N-1)}, \mathbf{y}^{(N-1)})], \mathbf{x}^{(i)} \in \mathcal{X}, \mathbf{y}^{(i)} \in \mathcal{Y}$.
  - The input data $\mathbf{X}$ is composed of the set of row vectors $\mathbf{x}^{(i)}$.
    - let $\mathcal{P}$ be the set of features $\{X_0, X_1, \ldots, X_{P-1}\}$, where $X_j = \left[x_j^{(0)}, x_j^{(1)}, \ldots, x_j^{(N-1)}\right]^T$.
    - then each i-th observation denoted as $\mathbf{x}^{(i)} = \left[x_0^{(i)}, x_1^{(i)}, \ldots, x_{P-1}^{(i)}\right]$ is an instance of $\mathcal{P}$.

Preliminaries (Cont.)

- **Learning Problem**. We want to discover some *unknown target function* $f : \mathcal{X} \to \mathcal{Y}$ from our data $\mathbf{D}$, which we assume is identically and independently distributed (i.i.d). To do so, we explore a *hypothesis set* $\mathcal{H}$ and use a given learning algorithm $\mathcal{A}$ to find a function g that we hope sufficiently approximates our target function: $\mathbf{D} \xrightarrow{\mathcal{A}} g \approx f$. For a given training example $(\mathbf{x}, \mathbf{y})$ in $\mathbf{D}$, we hope that $g(\mathbf{x}) = \hat{\mathbf{y}} \approx \mathbf{y}$ and g generalizes well for unseen observations.

Preliminaries (Cont.)

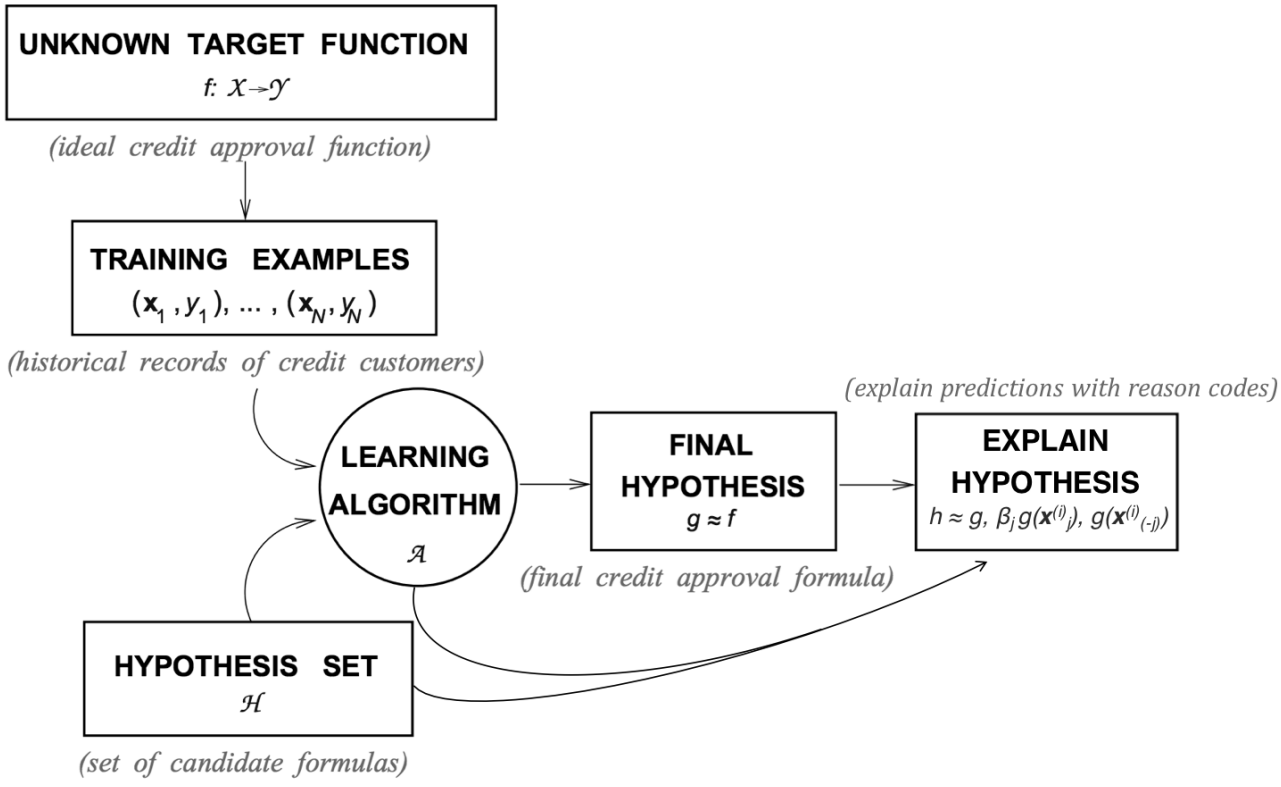

Figure: The learning problem. Adapted from **Learning From Data**.

Preliminaries (Cont.)

- **Explanation**. To justify the predictions of $g(\mathbf{x})$, we may resort to a number of techniques. Some techniques will be global in scope and simply seek to generate an interpretable approximation, h, for g itself, such that $h(\mathbf{x}) \approx g(\mathbf{x}) = \hat{\mathbf{y}}(\mathbf{x})$. Other techniques will be more local in scope and attempt to rank local contributions for each feature $X_j \in \mathcal{P}$ for some observation $\mathbf{x}^{(i)}$; this can create reason codes for $g(\mathbf{x}^{(i)})$. Local contributions are often estimated by evaluating the product of a learned parameter β_j in g with a corresponding observed feature value $x_j^{(i)}$ (i.e. $\beta_j x_j^{(i)}$), or by seeking to remove the contribution of some X_j in a prediction, $g(\mathbf{x}_{(-j)}^{(i)})$.

Outline

Interpretability

Model Complexity

- The more complex a function, the more difficult it is to explain. Simple functions can be used to explain more complex functions, however not all explanatory techniques are a good match for all types of models.

- When interpreting a complex model, it is important to verify that your interpretation techniques:
  - are appropriate for the response function complexity of the model.
  - are appropriate for global and local interpretation.
  - make sense given real-world context (aka match domain expertise).
  - provide explanations that match the explanations of other interpretation techniques.

Model Complexity (Cont.)

- **7** different techniques are presented to help interpret the results of your complex model. These techniques should be reviewed individually and within the general context of all other techniques. Depending on the relationship between your input training data and your target, certain interpretation techniques will be more reliable than others. These 7 techniques fall into two categories **model-agnostic** and **model-specific**, explained in the next section.

- The **7** techniques were chosen based on the essential questions they could answer about a complex model, including but not limited to:
  - how features are handled
  - how interactions are handled
  - the impact of individual features
  - how to create reason codes

Specific Interpretability Techniques

- **Model Agnostic:** Techniques to interpret the inner workings of any model. Requires surrogate models (detailed in next section), which can degrade the interpretation's quality. Since there is no guarantee that a surrogate model reflects the decisions of a complex model, it is important to verify how well a surrogate model fits the results of the complex model before using it as a method of interpretation.
  - Decision Tree **Surrogate** Models
  - Partial Dependence Plots
  - Individual Conditional Expectation Plots
  - LOCO

- **Model Specific:** Techniques that are attributed to and require specific models.
  - Random Forest or Gradient Boosting Variable Importance.
  - Shapley: Tree Shap Implementation.
  - Local Interpretable Model-agnostic Explanations (LIME).

Outline

Surrogate Models

Surrogate Models

- A *surrogate model* is a data mining and engineering technique in which a simple model is used to explain another complex model.
- Given our learned function g and set of predictions, $g(\mathbf{X}) = \hat{\mathbf{Y}}$, we can train a surrogate model h:

$$\mathbf{X}, \hat{\mathbf{Y}} \xrightarrow{\mathcal{A}_{\text{surrogate}}} h \tag{1}$$

- Ideally $h(\mathbf{X}) \approx g(\mathbf{X})$, however there exist few guarantees that $h(\mathbf{X})$ accurately represents g.
- To preserve interpretability, the hypothesis set for h is often restricted to linear models or decision trees.

Surrogate Models (Cont.)

- While any model can act as a surrogate model, surrogate models are chosen because they are easy for a human to interpret and explain. Surrogate models enhance transparency by providing:
  - Specific insights into the mechanism and results of a complex model.
  - Global or local interpretations of a complex model.
  - Visualizations that are easy to understand and compare.
- Surrogate models are limited to linear models, decision trees, and random forests.

Outline

Decision Tree Surrogate Model

Decision Tree Surrogate Model

- Given our learned function g and set of predictions, $g(\mathbf{X}) = \hat{\mathbf{Y}}$, we can train a decision tree surrogate model:

$$h_{\text{tree}} : \mathbf{X}, \hat{\mathbf{Y}} \xrightarrow{\mathcal{A}_{\text{surrogate}}} h_{\text{tree}} \tag{2}$$

- The decision tree surrogate model displays an approximate flow chart of g's decision making process to increase model transparency.
- It also shows likely important features and the most important interactions in g.

Outline

Partial Dependence Plots

Review of Marginal Expectation

- Recall that in a P-dimensional feature space, we can consider a single feature $X_j \in \mathcal{P}$ and its complement set $\mathcal{P}_{(-j)}$ (where $\{X_j\} \cup \mathcal{P}_{(-j)} = \mathcal{P}$).
- To describe the partial dependence $g(\mathbf{X})$ on X_j, we go through the following enumerated explanation:
  1. Expected value is $\mathbb{E}[g(X)] = \sum_{i=0}^{N-1} g(x^{(i)})p(x^{(i)})$.
  2. Let $g(\mathbf{X}) = g(X_j, X_{(-j)})$ and set $X_j = [x_j^0, \ldots, x_j^{P-1}]^T$).
  3. Then the marginal expectation over $X_{(-j)}$ is
  $$\mathbb{E}_{X_{(-j)}}\left[g(X_j, X_{(-j)})\right] = \sum_{i=0}^{N-1} g(X_j, X_{(-j)})p(x^{(i)}).$$
  4. Given that $\sum_{i=0}^{N-1} p(x^{(i)}) = 1$, and equal probabilities, $p(x^{(i)}) = \frac{1}{N}$.
  5. Thus $\mathbb{E}_{X_{(-j)}}\left[g(X_j, X_{(-j)})\right] = \frac{1}{N}\sum_{i=0}^{N-1} g(x_j, \mathbf{x}_{(-j)}^{(i)})$.

Partial Dependence Plots (Cont.)

- The partial dependence of a given feature X_j is the average of the response function g, where all the components of X_j are set to x_j ($X_j = [x_j^{(0)}, \ldots, x_j^{(N-1)}]^T$), and all other feature vectors of the complement set $\mathbf{x}_{(-j)}^{(i)}$ are left as the original dataset specified.
- Thus, the *one-dimensional partial dependence* of a function g on X_j is the marginal expectation:

$$\begin{aligned} \text{PD}(X_j, g) &= \mathbb{E}_{X_{(-j)}} \left[g(X_j, X_{(-j)}) \right] \\ &= \frac{1}{N} \sum_{i=0}^{N-1} g(x_j, \mathbf{x}_{(-j)}^{(i)}) \end{aligned} \tag{3}$$

Partial Dependence Plots (Cont.)

- Partial dependence plots show the partial dependence as a function of *specific values* of our feature subset.
- The plots show how machine-learned response functions change based on the values of an input feature of interest, while taking nonlinearity into consideration and averaging out the effects of all other input features.
- Partial dependence plots enable increased transparency in g and enable the ability to validate and debug g by comparing a feature's average predictions across its domain to known standards and reasonable expectations.

Conceptual Example

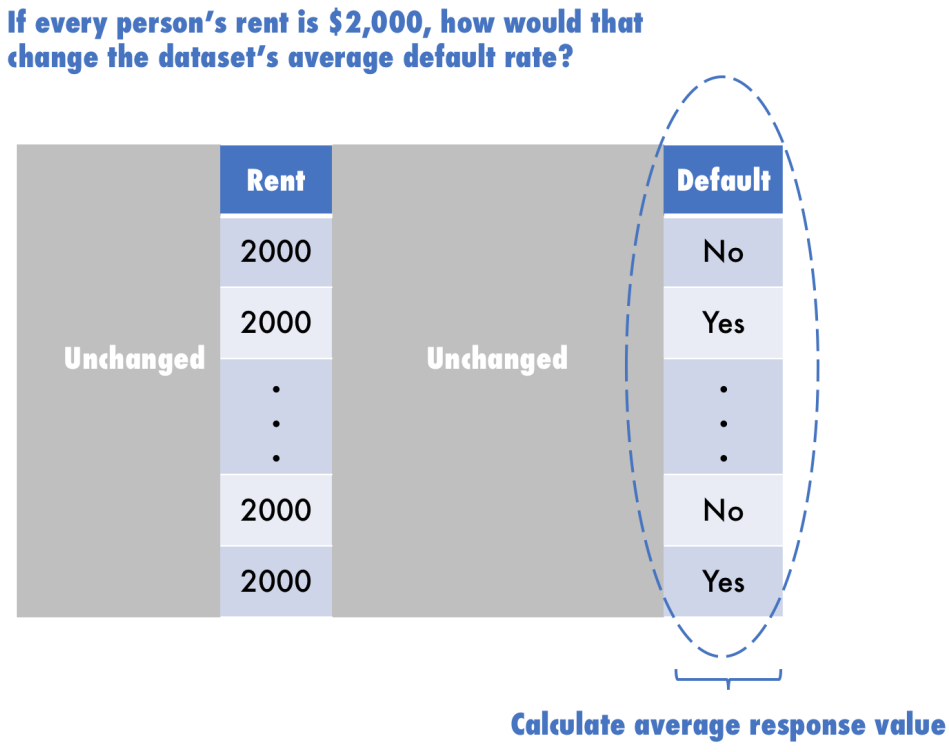

Figure: We want to predict whether someone will default on their monthly utility bill. The image shows one feature that we fixed ($Rent = 2000$), keeping the rest unchanged, to see its impact on the dataset's average default rate.

Outline

Individual Conditional Expectation

Individual Conditional Expectation

- Individual conditional expectation (ICE) plots can create localized explanations for a single observation of data using the same basic ideas as partial dependence plots.
- ICE is a type of nonlinear sensitivity analysis in which a model's predictions for a single observation are measured while a feature of interest is varied over its domain.
- To create an ICE plot for a row of interest $x^{(i)}$ and a feature of interest $x_j^{(i)}$, we vary $x_j^{(i)}$ within the range of the feature's original domain, and call a single such value $x_{j,q}$ (where q specifies an index within the domain):

  Specifically, we plot $g(x_{j,q}, \mathbf{x}_{(-j)}^{(i)})$ versus $x_{j,q}$ for each fixed $\mathbf{x}_{(-j)}^{(i)}$.

Outline

Feature Importance

Feature Importance

- Feature Importance (FI) measures the effect that a feature has on the predictions of a model.
- Unlike regression parameters, it is often unsigned and typically not directly related to the numerical predictions of the model.

Feature Importance

- **Global Importance**
  - measures the overall impact of an input feature on a model's predictions while taking nonlinearity and interactions into consideration.
  - values give an indication of the magnitude of a feature's contribution to model predictions for all observations.

- **Local Importance**
  - describes how the combination of the learned model rules or parameters and an individual observation's attributes affect a model's prediction for that observation while taking nonlinearity and interactions into effect.

Random Forest Feature Importance

To get the Global Feature Importance of a model on the original features, train a random forest model h_{RF} with some target outcome.

$$h_{\mathsf{RF}}(\mathbf{x}^{(i)}) = \frac{1}{B} \sum_{b=1}^{B} h_{\mathsf{tree},b} \left(\mathbf{x}^{(i)}; \Theta_b \right) \tag{4}$$

B : number of decision trees
Θ_b : set of splitting rules for each tree $h_{\mathsf{tree},b}$

Random Forest Feature Importance (cont.)

- At each split in each tree $h_{\text{tree},b}$, the improvement in the split-criterion is the importance measure attributed to the splitting feature, X_j. Specifically, the improvement is the difference between the squared error, SE, of the parent node pn and the child nodes cn at a given split. The importance measure is accumulated over all trees separately for each feature.

Random Forest Feature Importance (cont.)

- H2O splits a node to reduce the response variance in that node.
- Then SE is calculated by assuming an unbiased estimator (i.e. the mean squared error MSE is equal to the variance VAR). Given that $SE = MSE \times N$, Then $SE = MSE \times N = VAR \times N$.

$$VAR = \frac{1}{N} \sum_{i=0}^{N-1} (y^{(i)} - \bar{y})^2$$

$$VAR \times N = \left[\frac{1}{N} \times \sum_{i=0}^{N-1} (y^{(i)})^2 - N \times (\bar{y})^2 \right] \times N \tag{5}$$

$$= \left[\sum_{i=0}^{N-1} (\frac{(y^{(i)})^2}{N}) - \bar{y}^2 \right] \times N = SE$$

Random Forest Feature Importance (cont.)

- at each split in each tree $h_{\text{tree},b}$, the improvement in the split-criterion is the importance measure attributed to the splitting feature, X_j. Specifically, the improvement is the difference between the squared error, SE, of the parent node pn and the child nodes cn at a given split. The importance measure is accumulated over all trees separately for each feature.

$$FI(X_j) = \sum_{b=1}^{B} \sum_{tl=1}^{TD_b} \kappa \times SE_{tl}$$

$$\text{where } \kappa = \kappa(X_{j,tl}) = \begin{cases} 1 & \text{if } X_{j,tl} = X_j \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$SE_{tl} = SE_{pn} - SE_{cn}$$

SE_{tl}: Squared Error at tree level tl
TD_b: Max tree depth of each b
tl: tree level

Random Forest Feature Importance

- The aggregated feature importance values are then scaled between 0 and 1, such that the most important feature has an importance value of 1.

  Let $m = \max_{0 \leq l \leq P-1} FI(X_l)$, $l =$ index of feature X_l,
  then

$$AF(X_j) = \frac{FI(X_j)}{m} \tag{7}$$

  Where $AF(X_j)$ is the aggregated feature importance.

Outline

Leave One Covariate Out

Leave One Covariate Out

- Leave-one-covariate-out (LOCO) provides the feature importance values at a per-observation level (i.e. what is the feature importance of X_j for observation $\mathbf{x}^{(i)}$).

- LOCO is calculated by subtracting a model's prediction, $g(\mathbf{x}^{(i)})$, for a row-observation with all its features, from that model's prediction for the same row *without* the input feature X_j of interest.

$$g(\mathbf{x}^{(i)}_{(-j)}) - g(\mathbf{x}^{(i)}), \tag{8}$$

  where $\mathbf{x}^{(i)}_{(-j)} \in \mathcal{P}_{(-j)}$, given that $\mathcal{P}_{(-j)}$ is the complement set to $\{X_j\} \in \mathcal{P}$ (i.e $\{X_j\} \cup \mathcal{P}_{(-j)} = \mathcal{P}$).

Outline

LIME

Ribeiro, Singh, and Guestrin, 2016 defines LIME for some observation $\mathbf{x} \in \mathcal{X}$:

$$\underset{h \in \mathcal{H}}{\arg\max}\, \mathcal{L}(g, h, \pi_{\mathbf{X}}) + \Omega(h) \tag{9}$$

Here g is the function to be explained, 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.

Typically, h_{GLM} is constructed such that

$$\mathbf{X}^{(*)}, g(X^{(*)}) \xrightarrow{\mathcal{A}_{\text{surrogate}}} h_{\text{GLM}} \tag{10}$$

where $\mathbf{X}^{(*)}$ is a generated sample, $\pi_{\mathbf{X}}$ weighs $\mathbf{X}^{(*)}$ samples by their Euclidean similarity to $\mathbf{x}$, local feature importance is estimated using $\beta_j x_j$, and L_1 regularization is used to induce a simplified, sparse h_{GLM}.

LIME (Cont.)

- LIME is ideal for creating highly interpretable explanations for non-decision tree models and for neural network models trained on unstructured data, e.g. deep learning models.
- Use regression fit measures to assess the trustworthiness of LIME explanations.
- Local feature importance values are offsets from a local intercept.
  - Note that the intercept in LIME can account for the most important local phenomena.
  - Generated LIME samples can contain large proportions of out-of-range data that can lead to unrealistic intercept values.

LIME (Cont.)

- To increase the trustworthiness of LIME explanations, try LIME on discretized input features and on manually constructed interactions.
- Use cross-validation to construct standard deviations or even confidence intervals for local feature importance values.
- LIME can fail, particularly in the presence of extreme nonlinearity or high-degree interactions.

Outline

Shapley Feature Importance

Overview of SHAP: SHapley Additive exPlanations

- **What:** SHAP values provide feature importance for a complex model, through numeric values that are easy for humans to understand.
- **Why:** SHAP is a unified framework that reveals the relationship between other published interpretability techniques [citations needed], as well as solutions to these techniques that are easier to interpret and satisfy the SHAP properties.
- **Examples:** Interpretability methods that fall under the SHAP framework. *Shapley Feature Importances* explicitly apply SHAP framework and satisfy its constraints, while *LOCO* and *Feature Importance* fall under the SHAP framework but are not designed to satisfy the SHAP Properties.

SHAP: SHapley Additive exPlanations (Cont.)

The following slides summarize and explain the work found in [Lundberg and Lee, 2017].

Explanation Models

- **Explanation models:** a form of surrogate models, explanation models h are designed to provide an interpretable approximation of a complex model.

- **Simplified inputs:** a binary vector $\mathbf{x}'$, of interpretable values, that maps to a given training observation $\mathbf{x}$, through a mapping function $M_{\mathbf{x}}$ specific to $\mathbf{x}$, such that $\mathbf{x} = M_{\mathbf{x}}(\mathbf{x}')$.

- **Local methods:** are techniques to explain a model's prediction for a specific observation, as oppose to observations in general. When a local method works well, $h(\mathbf{z}') \approx g(M_{\mathbf{x}}(\mathbf{z}'))$ whenever $\mathbf{z}' \approx \mathbf{x}'$.

Additive Feature Attribution Methods

Additive feature attribution methods sum up the feature attributions of an observation $\mathbf{x}$ to approximate and thereby explain the prediction $g(\mathbf{x})$ corresponding to that observation.

- **Definition 1** additive feature attribution methods use an explanation model that is a linear function of binary variables:

$$h(\mathbf{z}') = \phi_0 + \sum_{j=1}^{P} \phi_j z'_j, \tag{11}$$

ϕ_j: each feature's attributed impact, where $\phi_j \in \mathbb{R}$.
P: the number of predictors.
$\mathbf{z}'$: the binary vector where $\mathbf{z}' \in \{0, 1\}^P$.

Review of Cooperative Game Theory's Shapley Number

- **Cooperative Games:** a game in which players benefit from teaming up rather than playing independently.
- **The Shapley Number:** a numeric value that determines how much a player should gain or lose, given the number of players and the value attributed to each if they were to play independently.
  - the Shapley Number is easy to calculate when there only a few players: it's the average value of all possible combinations for each player.

Shapley Number Axioms

The Shapley Number calculation is based on the following three axioms:

- **Efficiency**: the sum of the amounts attributed to each player must equal the total amount attributed to the players when they play together.
- **Symmetry**: changing the names of players doesn't affect the attributed importance of each player.
- **Monotonicity:** If the total value of a game increases the value attributed to each player will not decrease. This axiom includes the linearity axiom - the sum of players playing individually has to equal the sum of them playing together - and the null effect axiom - players that don't participate don't contribute to the game.

Game Theory Axiom Applied to Machine Learning

The previous Shapley Number axioms can be applied as constraints to solve a SHAP explanation model. Below are the SHAP properties and their mapping to the Shapley axioms.

- Local Accuracy $\sim$ Efficiency Axiom
- Missingness $\sim$ Null Effect Axiom - adapted to handle missing data
- Consistency $\sim$ Monotonicity Axiom

Local Accuracy Property

To satisfy the local accuracy property, an explanation model h should return the same prediction as the complex model g, given a simplified input $\mathbf{x}'$ that maps back to $\mathbf{x}$ via the mapping function $M_{\mathbf{x}}$. Specifically,

$$\begin{aligned} &\text{if } \mathbf{x} = M_{\mathbf{x}}(\mathbf{x}'), \phi_0 = g(M_{\mathbf{x}}(\mathbf{0})), \\ &\text{and } h(\mathbf{x}') = \phi_0 + \sum_{j=1}^{P} \phi_j x'_j, \\ &\text{then } g(\mathbf{x}) = g(M_{\mathbf{x}}(\mathbf{x}')) = h(\mathbf{x}'). \end{aligned} \tag{12}$$

Missingness Property

If a feature value is missing during training, that implies it shouldn't have an attributed effect value. Remember that $x'_j = 0$ means the original input value x_j is not present.

$$x'_j = 0 \implies \phi_j = 0 \tag{13}$$

Consistency Property

If a model changes and the change causes a feature to have a larger impact on the model's predicted value for a simplified input (or keep it the same), the feature's new attributed value cannot decrease.

Let $g_{\mathbf{x}}(\mathbf{z}') = g(M_{\mathbf{x}}(\mathbf{z}'))$, and $\mathbf{z}' \setminus j$ denote setting $z'_j = 0$.

Let g and g' be any two models, if

$$g'_{\mathbf{x}}(\mathbf{z}') - g'_{\mathbf{x}}(\mathbf{z}' \setminus j) \geq g_{\mathbf{x}}(\mathbf{z}') - g_{\mathbf{x}}(\mathbf{z}' \setminus j), \, \forall \, \mathbf{z}' \in \{0,1\}^P, \tag{14}$$

then

$$\phi_j(g', \mathbf{x}) \geq \phi_j(g, \mathbf{x}) \tag{15}$$

Inconsistency

Several of the other attributed feature importance calculations (gain, permutation, etc) assign inconsistent attribution values, under different models:

$$g'_{\mathbf{x}}(\mathbf{z}') - g'_{\mathbf{x}}(\mathbf{z}' \setminus j) \geq g_{\mathbf{x}}(\mathbf{z}') - g_{\mathbf{x}}(\mathbf{z}' \setminus j), \; \forall \, \mathbf{z}' \in \{0,1\}^P, \tag{16}$$

and

$$\phi_j(g', \mathbf{x}) < \phi_j(g, \mathbf{x}) \tag{17}$$

Even though the presence of z'_j shows a larger difference for g' then g, its attributed effect for the original input $\phi_j(g', \mathbf{x})$, is less.

SHAP Theorem 1

SHAP Theorem 1: only one possible explanation model h follows from SHAP Definition 1 and satisfied Properties 1-3:

$$\phi_j(g, \mathbf{x}) = \sum_{\mathbf{z}' \subseteq \mathbf{x}'} \frac{|\mathbf{z}'|!(P - |\mathbf{z}'| - 1)!}{P!} \left[g_{\mathbf{x}}(\mathbf{z}') - g_{\mathbf{x}}(\mathbf{z}' \setminus j) \right] \tag{18}$$

$|\mathbf{z}'|$: the number of non-zero entries in $\mathbf{z}'$
$\mathbf{z}' \subseteq \mathbf{x}'$: represents all $\mathbf{z}'$ vectors where the non-zero entries are a subset of the non-zero entries in $\mathbf{x}'$.

- This theorem implies that methods which fall under the explanation model but are not solved in consideration of the Shapley values likely break the local accuracy and/or consistency properties.

SHAP Values Overview

As stated in [Lundberg and Lee, 2017]: "SHAP values are the Shapley values of the complex model's conditional expectation function. This means that SHAP values attribute to each feature the change in the expected model prediction when conditioning on that feature."

SHAP Approximations

SHAP values makes the following simplifications and approximations:

- Defined simplified input mapping:

$$g(M_{\mathbf{x}}(\mathbf{z}')) = \mathbb{E}\left[g(\mathbf{z})|\, \mathbf{z}_s\right] = \mathbb{E}_{\mathbf{z}_{\bar{s}}|\mathbf{z}_s}\left[g(\mathbf{z})\right] \tag{19}$$

- Assumptions of feature independence:

$$g(M_{\mathbf{x}}(\mathbf{z}')) \approx \mathbb{E}_{\mathbf{z}_{\bar{s}}}\left[g(\mathbf{z})\right] \tag{20}$$

- Assumptions of model linearity:

$$g(M_{\mathbf{x}}(\mathbf{z}')) \approx g([\mathbf{z}_s, \mathbb{E}\left[\mathbf{z}_{\bar{s}}\right]]) \tag{21}$$

Where $M_{\mathbf{x}}(\mathbf{z}') = \mathbf{z_s}$, S is the set of non-zero indexes in $\mathbf{z}'$, and $\bar{S}$ is the set of features not in S.

SHAP Values For Tree Ensemble Feature Attributions

Given

$$g_{\mathbf{x}}(S) = g(M_{\mathbf{x}}(\mathbf{z}')) = \mathbb{E}\left[g(\mathbf{x})|\,\mathbf{x}_s\right] \tag{22}$$

where S is the set of non-zero indexes in $\mathbf{z}'$ and $\mathbb{E}\left[g(\mathbf{x})|\,\mathbf{x}_s\right]$ is the expected value of the function conditioned on a subset of S input features. We can calculate the attribute value ϕ_j of each feature:

$$\phi_j(g, \mathbf{x}) = \sum_{S \subseteq P \setminus \{j\}} \frac{|S|!(P - |S| - 1)!}{P!} \left[g_{\mathbf{x}}(S \cup \{j\}) - g_{\mathbf{x}}(S)\right] \tag{23}$$

again, P is the set of all features, so $P \setminus \{j\}$ removes the jth feature from the set.

Some techniques that fall under Additive Feature Attribution Methods

Each of the following interpretability techniques, are specific implementations of the SHAP explanation model.

- LOCO
- Tree-Interpreter
- LIME
- DeepLIFT
- Layer-Wise Relevance Propagation
- Shapley Regression Values
- Shapley Sampling Values
- Quantitative Input Influence

Outline

References

References I

Bastani, Osbert, Carolyn Kim, and Hamsa Bastani (2017). "Interpreting blackbox models via model extraction". In: *arXiv preprint arXiv:1705.08504*. URL: https://arxiv.org/pdf/1705.08504.pdf.

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

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

Goldstein, Alex et al. (2015). "Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation". In: *Journal of Computational and Graphical Statistics* 24.1.

Lei, Jing et al. (2017). "Distribution-Free Predictive Inference for Regression". In: *arXiv preprint arXiv:1604.04173*. URL: https://arxiv.org/abs/1604.04173.

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

Lundberg, Scott M and Su-In Lee (2017). "A Unified Approach to Interpreting Model Predictions". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., pp. 4765–4774. URL: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

References II

Ribeiro, Marco Tulio, 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, pp. 1135–1144. URL: http://www.kdd.org/kdd2016/papers/files/rfp0573-ribeiroA.pdf.