Intro to SHAP and New Ideas

SHapley Additive exPlanations

Presenter: wdai

November 21, 2025

College of Engineering and Computing
STATISTICS
George Mason University

SHapley Additive exPlanations

Outline

Why: The Problem Statement

Many of the highest-performing machine learning models are complex "black boxes."

- Ensembles (Random Forests, Gradient Boosting)
- Deep Neural Networks

The Challenge: When a model predicts $\hat{f}(\mathbf{x}) = y$, how do we know *why*?

The Goal: We need a rigorous method to explain individual predictions by attributing the outcome to each input feature.

Core Concept: Shapley Values

The core idea comes from cooperative game theory (**shapleyStochasticGames1953**).

- **Goal:** To fairly distribute a total "payout" among a group of collaborating "players."

The Central Question

How much did each player *individually* contribute to the final outcome of the team?

The Machine Learning “Game” Metaphor

We can frame a single model prediction as a game:

- **The “Game”:** Explaining the prediction $\hat{f}(\mathbf{x})$ for a single instance $\mathbf{x}$.
- **The “Players”:** The feature values of the instance $(x_1, x_2, \ldots, x_p)$.
- **The “Payout”:** The model's prediction for this instance minus the average (baseline) prediction over the whole dataset.

$$\text{Payout} = \hat{f}(\mathbf{x}) - \mathbb{E}[\hat{f}(\mathbf{X})]$$

Formal Definition: Shapley Value ϕ_j

The Shapley value ϕ_j for a feature j is its average marginal contribution, weighted and summed over all possible coalitions (subsets) S of features that *don't* include j.

Shapley Value Definition

$$\phi_j(val) = \sum_{S \subseteq \{1,\ldots,p\} \setminus \{j\}} \frac{|S|!(p - |S| - 1)!}{p!} (val(S \cup \{j\}) - val(S))$$

- p is the total number of features.
- The value function val is the payout function for coalitions of feature values.
- $val(S)$ is the "payout" of the coalition S (i.e., the model's prediction using only features in S).

Example: One model works with 4 features X_1, X_2, X_3, X_4. The prediction for feature values in S that are marginalized over features X_2 and X_4 is

$$val_{\mathbf{x}}(S) = val_{\mathbf{x}}(\{1,3\}) = \int_{\mathbb{R}} \int_{\mathbb{R}} \hat{f}(x_1, X_2, x_3, X_4) \, d\mathbb{P}_{X_2 X_4} - \mathbb{E}[\hat{f}(\mathbf{X})].$$

The Axioms of a "Fair" Payout

Shapley values are the *only* attribution method that satisfies three key properties:

1. **Efficiency (Local Accuracy):** The sum of all feature contributions (ϕ_j) equals the total "payout" (the prediction minus the average).

$$\sum_{j=1}^{p} \phi_j = \hat{f}(\mathbf{x}) - \mathbb{E}[\hat{f}(\mathbf{X})]$$

2. **Symmetry:** If two features j and k contribute identically to all possible coalitions, their attributions are the same ($\phi_j = \phi_k$).
3. **Dummy:** If a feature j has no impact on the prediction (it contributes 0 to all coalitions), its attribution is zero ($\phi_j = 0$).

From Shapley Values to SHAP

SHAP (SHapley Additive exPlanations) connects many explanation methods (like LIME) using Shapley values as a unifying framework.

Key Idea: SHAP defines a new class of "Additive Feature Attribution Methods."

- All explanations are based on a simple, linear *explanation model* g that approximates the original complex model $\hat{f}$ for a single prediction.

Class: Additive Feature Attribution Methods

The explanation model g is a linear function of binary variables $\mathbf{z}'$:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

- $g(\mathbf{z}')$: The explanation for the simplified input $\mathbf{z}'$.
- $\mathbf{z}' \in \{0, 1\}^M$: A binary "coalition vector" representing which features are "present" (1) or "absent" (0).
- $\phi_j \in \mathbb{R}$: The attribution for feature j. **This is the SHAP value.**
- ϕ_0: The base value, or $\mathbb{E}[\hat{f}(\mathbf{X})]$.

The SHAP Properties (The Axioms Re-stated)

SHAP re-frames the Shapley axioms for this explanation model:

1. **Local Accuracy (Efficiency):** The explanation model g must match the original model's output $f(x)$ when all features are present (i.e., $\mathbf{z}'$ is all 1s).

$$f(x) = g(x') = \phi_0 + \sum_{j=1}^{M} \phi_j$$

2. **Missingness (Dummy):** A missing feature ($z'_j = 0$) has no attribution ($\phi_j = 0$).
3. **Consistency (Symmetry/Additivity):** If a model $\hat{f}$ changes so a feature's marginal contribution *always* increases or stays the same (regardless of other features), its SHAP value ϕ_j should not decrease.

The Unifying Theorem

Theorem 1 (lundbergUnifiedApproachInterpreting2017)

There is **only one** possible explanation model g (i.e., one set of ϕ_j values) that satisfies all three properties (Local Accuracy, Missingness, Consistency).

The unique solution for ϕ_i is:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!} \left[f_x(z') - f_x(z' \setminus i)\right]$$

- $f_x(z')$ is the prediction of the original model f for the coalition z'.

The Statistical Definition: What is $f_x(z')$?

How do we calculate the prediction $f_x(z')$ for a coalition z' (e.g., $z'_1 = 1, z'_2 = 0$)? We can't just "remove" x_2. We must account for its effect.
SHAP defines $f_x(z')$ as the **conditional expectation**:

$$f_x(z') = \mathbb{E}[f(\mathbf{X}) \mid \mathbf{X}_S = \mathbf{x}_S]$$

- S is the set of "present" features (where $z'_j = 1$).
- In words: "The expected model prediction, given the values of the features we know (in S)."

This is the main statistical challenge. Computing this expectation is computationally very difficult.

Estimation Methods (How to Compute SHAP)

Exact computation is $O(2^M)$, which is infeasible. We need approximations(**molnarInterpretingMachineLearning2023**).

- **KernelSHAP (Model-Agnostic):**
  - A clever, model-agnostic approximation.
  - Connects SHAP to LIME by using a specific weighted linear regression (the "Shapley kernel") to solve for the ϕ_j values.
- **TreeSHAP (Model-Specific):**
  - A highly optimized, fast algorithm specifically for tree-based models (Decision Trees, Random Forests, XGBoost, etc.).
  - For trees, it can compute the *exact* SHAP values efficiently.

Usage: From Local to Global Insights

We can aggregate the local SHAP values ($\phi_j^{(i)}$ for all n instances) to get global model insights.

- **SHAP Feature Importance:** The mean absolute SHAP value for each feature. This is a more robust importance measure than simple permutation importance.

$$I_j = \frac{1}{n} \sum_{i=1}^{n} |\phi_j^{(i)}|$$

- **SHAP Summary Plot:** Combines feature importance with feature effects. It plots the SHAP value for every feature for every instance, often colored by the feature's original value.
- **SHAP Dependence Plot:** A scatter plot of a feature's value (x_j) vs. its SHAP value (ϕ_j). This is excellent for revealing interactions.

Conclusion - Sec 1

- SHAP provides a **unified theory** for feature attributions, connecting many disparate methods.
- It is founded on solid game theory axioms, ensuring explanations are **fair and accurate** (Efficiency, Symmetry, Dummy).
- It is the **only** additive method satisfying Local Accuracy, Missingness, and Consistency.
- It provides powerful **local** (single prediction) and **global** (model-wide) insights.

Recap: Generalized Additive Models (GAMs)

A GAM is an extension of a GLM.

- **Core Idea:** It replaces the simple linear term $\beta_j x_j$ with a flexible, non-linear function $f_j(x_j)$ for each feature.
- **Key Property:** The model is *still additive*. The functions are summed together.

GAM Model Formula

$$g[\mathbb{E}(Y|\mathbf{x})] = \beta_0 + f_1(x_1) + f_2(x_2) + \cdots + f_p(x_p)$$

- g is the link function (e.g., logit, log).
- f_j is a non-linear "spline" function learned from the data.

The Key Insight: SHAP on an Additive Model

What happens when we use an *additive explanation* (SHAP) on an *additive model* (GAM)?

- The math simplifies.
- We don't need complex, model-agnostic approximations (like KernelSHAP).
- The SHAP value ϕ_j for a feature j has an exact, closed-form solution.

The "SHAP-GAM" Connection

For an additive model, the SHAP value for a feature is simply its local function value, $f_j(x_j)$, centered by its average function value, $\mathbb{E}[f_j(X_j)]$.

Formal Definition: SHAP for GAMs

Let's assume a simple GAM (identity link g, $f(x) = \beta_0 + \sum f_j(x_j)$):

1. **The Model:** $f(\mathbf{x}) = \beta_0 + f_1(x_1) + \cdots + f_p(x_p)$
2. **The Baseline (ϕ_0):** $\phi_0 = \mathbb{E}[f(\mathbf{X})] = \mathbb{E}[\beta_0 + \sum f_j(X_j)]$

$$\phi_0 = \beta_0 + \sum_{j=1}^{p} \mathbb{E}[f_j(X_j)]$$

3. **The SHAP Value (ϕ_j):**

$$\phi_j(\mathbf{x}) = f_j(x_j) - \mathbb{E}[f_j(X_j)]$$

This perfectly satisfies the SHAP "Efficiency" (Local Accuracy) property:

$$f(\mathbf{x}) - \mathbb{E}[f(\mathbf{X})] = \sum_{j=1}^{p} (f_j(x_j) - \mathbb{E}[f_j(X_j)]) = \sum_{j=1}^{p} \phi_j(\mathbf{x})$$

Example: Bike Rental GAM

Use the bike rental example(**molnarInterpretingMachineLearning2023**).

- **Goal:** Predict ‘rentals‘ (count), so we use a Poisson GLM / GAM.
- **Model:** ‘log(E[rentals])‘ $= \beta_0 + f_1(\text{temp}) + f_2(\text{workday})$
- For simplicity, ‘workday‘ is linear, so $f_2(x_{\text{work}}) = \beta_{\text{work}} \cdot x_{\text{work}}$.

We want to explain a single prediction:

- **Instance x:** A hot day (e.g., 35°C) that is a workday.
- **Prediction $f(x)$:** 4000 rentals (example value).
- **Baseline ϕ_0:** Average prediction is 5500 rentals (example value).
- **Total Payout to Explain:** $4000 - 5500 = -1500$ rentals.

Example: Calculating the SHAP Values

We need to find ϕ_{temp} and ϕ_{workday} that sum to -1500. (Note: SHAP values are on the scale of the linear predictor, 'log(rentals)', but can be transformed back.)

1. **Temperature Contribution ϕ_{temp}**
   - $\phi_{\text{temp}} = f_1(35°\text{C}) - \mathbb{E}[f_1(\text{Temp})]$
   - We get $f_1(35°\text{C})$ directly from the spline plot for temperature.
   - High temperatures have a negative effect. Say the centered plot f_1 is at -0.4 for 35°C.
   - $\phi_{\text{temp}} = -0.4$ (in log-count space).
2. **Workday Contribution ϕ_{workday}**
   - $\phi_{\text{workday}} = f_2(1) - \mathbb{E}[f_2(\text{Workday})]$
   - $= (\beta_{\text{work}} \cdot 1) - \mathbb{E}[\beta_{\text{work}} \cdot X_{\text{work}}]$
   - $= \beta_{\text{work}}(1 - \text{mean}(X_{\text{work}}))$
   - This is the standard, exact contribution for a linear feature.

Why Use SHAP on a "White-Box" Model?

If GAMs are already interpretable, what does SHAP add?

- **Standard GAM Interpretation (Global):**
  - "The model is a sum of functions. Here is the plot for temperature, here is the plot for workday..."
  - This is a *global* view of feature effects.
- **SHAP Interpretation (Local):**
  - "For *this specific prediction*, the high temperature contributed -0.4, and the workday contributed -0.1..."
  - This is a *local* view, explaining one decision.
- **A Unified Currency:**
  - SHAP provides a single "currency" (ϕ) to compare the magnitude of a complex non-linear spline effect (e.g., $f_1(35)$) with a simple linear effect (e.g., $\beta_2 \cdot 1$).
  - This is extremely powerful for communicating results.

SHAP Summary by Model Type

Table: Comparison of SHAP Estimation by Model Type

Model Type	**Package (`shap`)**	**Speed**	**Exactness**
Linear	`LinearExplainer`	Instant	Exact
GAM	`AdditiveExplainer`	Very Fast	Exact
XGBoost	`TreeExplainer`	Fast	Exact
FFNN	`DeepExplainer`	Slow	Approximate

Conclusion - Sec 2

- SHAP is not just for "black-box" models like deep networks.
- When applied to additive models (GLMs, GAMs), SHAP values have an **exact, analytic solution**.
- $\phi_j(\mathbf{x}) = f_j(x_j) - \mathbb{E}[f_j(X_j)]$
- This bridges the gap between traditional *global* model interpretation (plotting f_j) and modern *local* explanation methods (attributing ϕ_j).
- It allows us to directly compare the local impact of linear and non-linear components on a single prediction.