Shapley Values, LIME, and SHAP

David S. Rosenberg

Bloomberg ML EDU

December 8, 2021

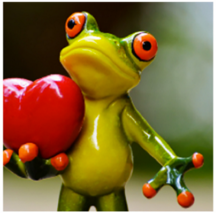
Contents

Recap: interpretable representations

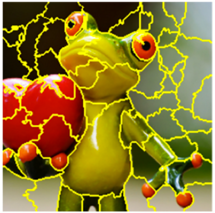
Simplified features / interpretable representation

- LIME introduced the idea of an "interpretable representation." [RSG16b].
- They ask: what good is interpreting a model f if we can't interpret its features?
- The SHAP paper calls these things "simplified features." [LL17]
- Each interpretable representation is designed to interpret
  - the prediction **for a particular example** $x_0 \in \mathcal{X}$.
- The idea is **not** to build a single simplified representation for all of $\mathcal{X}$.
- But rather, to represent $x \in \mathcal{X}$ that are "near" to $x_0 \in \mathcal{X}$ in some sense.

Interpretable components



Original Image

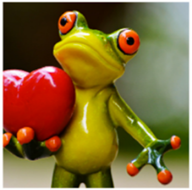
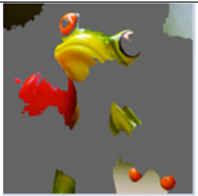
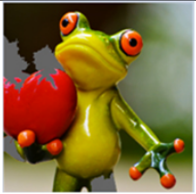


Interpretable Components

- A segmentation algorithm has broken the image into "interpretable components".
- There is a "simplified feature" for each component ($=\mathbb{1}$ [component is "included"]).

Image from [RSG16a].

Simplified feature representations

$x_0 \in \mathcal{X}$				
$x' \in \{0,1\}^M$	$(1,1,\ldots,1,1)$	$(0,0,1,0,\ldots,1)$	$(1,0,0,0,\ldots,0)$	$(1,0,1,1\ldots,0,1)$

- The number or interpretable components M is specific to a particular $x_0 \in \mathcal{X}$.
- The mapping from $x' \mapsto x$ is also specific to the particular $x_0 \in \mathcal{X}$.

Image from [RSG16a].

Example simplified features

- Fix some $x_0 \in \mathcal{X}$ in our original feature space.
- Let $\{0,1\}^M$ be our simplified feature space (M generally depends on x).
- Define a mapping $h_{x_0} : \{0,1\}^M \to \mathcal{X}$ such that
  - $h_{x_0}((1,1,\ldots,1)) = x_0$.
  - for any $x' \in \{0,1\}^M$, $h_{x_0}(x') \in \mathcal{X}$ is some variation of x_0.
- For example
  - $h(x')$ is an image with regions blocked out.
  - $h(x')$ is a sentence with words eliminated.

LIME is approximating a set function

Additive feature attribution methods

Definition

[LL17]. **Additive feature attribution methods** have an explanation model that is a linear function of binary variables:

$$g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i,$$

where $x' \in \{0,1\}^M$, where M is the number of simplified features, and $\phi_i \in \mathbb{R}$.

- The idea is to use $g(x')$ to interpret the prediction $f(x)$.
- In particular, $g(x')$ is so simple that we can read off
  - the importance of each interpretable feature x'_i by the size of ϕ_i.

Linear LIME

- Suppose we're trying to interpret the prediction $f(x_0)$.
- In LIME, we try to find an **interpretable** g that approximates f near x.
- Let $h_x : \{0,1\}^M \to \mathcal{X}$ be our simplified feature map.
- Consider linear models on $\{0,1\}^M$ as our interpretable model class:

  $$g(x') = w_0 + w_1 x'_1 + \cdots + w_M x'_M$$

  for $x' \in \{0,1\}^M$.
- This will give us an additive feature attribution method.

Linear LIME optimization problem

- In LIME, we sample a set of "perturbations" $\mathcal{D} \subset \{0,1\}^M$, uniformly at random.
- And the LIME optimization problem is

$$\arg\min_g \sum_{x' \in \mathcal{D}} \pi_x(h_x(x'))\left(f(h(x')) - g(x')\right)^2$$

or (for Tabular LIME) it's

$$\arg\min_g \sum_{x' \in \mathcal{D}} \pi(x')\left(f(h(x')) - g(x')\right)^2.$$

- Here our objective sums over binary vectors in $\{0,1\}^M$.
- The quadratic objective for generalized Shapley values summed over subsets of $\{1,\ldots,M\}$.
- These are exactly equivalent!

Let $\{1,\ldots,M\}$ index a set of features.

- Let $\{0,1\}^M$ be any simplified feature space.
- Note the obvious correspondence between
  - subsets $S \subset \{1,\ldots,M\}$ and
  - binary vectors $x' \in \{0,1\}^M$.
- Namely, $S = \left\{ j \mid x'_j = 1 \right\} \iff x'_j = \mathbb{1}[j \in S]$.
- Thus we can view $(f \circ h)(x')$ as a set function on features $\{1,\ldots,M\}$.
- Similarly, we can view $g(x') = w_0 + w_1 x'_1 + \cdots + w_M x'_M$ as a set function.
- So [linear] LIME is trying to approximate one set function by another, simpler one.
- Sound familiar?

LIME and generalized Shapley values

- Consider the Tabular LIME objective

$$\underset{w \in \mathbb{R}^M}{\arg\min} \sum_{x' \in \mathcal{D}} \pi(x') \left(f(h(x')) - g(x') \right)^2,$$

  where $g(x') = w_1 x'_1 + \cdots + w_M x'_M$.
- Suppose we
  - take $\mathcal{D} = \{0,1\}^M - \{(0,\ldots,0),(1,\ldots,1)\}$— i.e., $2^M - 2$ perturbations,
  - assume $\pi(x')$ depends only on the number of 1's in x' (as in Tabular LIME),
  - constrain the optimization so that $g((1,\ldots,1)) = f(x_0)$, and
  - shift f so that $f(h(0)) = 0$,
- then we exactly have a generalized Shapley objective function, with
  - weight function $\pi(x')$ and
  - "set function" $(f \circ h)(x')$.

Shapley version of LIME

- In addition to the previous conditions on

$$\underset{w \in \mathbb{R}^M}{\arg\min} \sum_{x' \in \mathcal{D}} \pi(x') \left(f(h(x')) - g(x') \right)^2,$$

- let's use the **Shapley kernel** $\pi(x') = \binom{M-2}{|x'|-1}^{-1}$, where $|x'|$ is the number of 1's in x'.
- We'll call this the **Shapley-LIME objective.**
- Then the minimizing $w_1, \ldots, w_M$ are Shapley values.
- Thus using the w's to allocate "credit" to each interpretable feature,
  - has all the theoretical properties of Shapley values

Comparing Shapley-LIME to LIME

- The Tabular LIME weight function $\pi(x')$
  - puts more weight on x' with more 1's (because they're closer to x_0)
- (The more features you "cover up", the more different the result is from x_0).
- But with the Shapley kernel,
  - we have large weights when x is mostly 1's
  - AND large weights when x is mostly 0's.
- A Shapley-ist might say that this is this the most justifiable form of LIME.

SHAP (SHapley Additive exPlanation) values

SHAP: original definition

- Consider the set function

$$v(S) = \mathbb{E}[f(x_S, X_C) \mid X_S = x_S] - \mathbb{E}[f(X)].$$

- Note that $v(\emptyset) = 0$.
- [LL17] defines the **SHAP values** as the Shapley values for $v(S)$.
- Let $\phi_1, \ldots, \phi_M$ be those Shapley values.
- We can define $\phi_0 = \mathbb{E}[f(X)]$, and then

$$g(x') = \phi_0 + \sum_{i=1}^{M} \phi_i x_i'.$$

- This is another additive feature attribution method.

Shapley-LIME vs SHAP

- The only difference between Shapley-LIME and SHAP is the set function.
- In Shapley-LIME, the set function for a subset S of the M interpretable features is
    - found by first mapping S to an element of the original feature space $\mathcal{X}$,
    - then applying the function f that we're trying to interpret.
    - The resulting value is the set function evaluation on S.
- In SHAP, the set function involves various expectations over features.
- 2 different set functions give 2 different sets of Shapley values and feature interpretations.

SHAP in practice

- As discussed in previous modules
  - it's not easy to compute $\mathbb{E}[f(x_S, X_C) \mid X_S = x_S]$.
- So not only are Shapley values hard to compute,
  - but in the case of SHAP, even computing the set function is hard.
- [LL17] presents a method called "**Kernel SHAP**":
  - Replace the conditional expectation with the marginal expectation $\mathbb{E}[f(x_s, X_C)]$, and
  - compute Shapley values using the hybrid Monte Carlo method described in the previous module.
- [LL17] suggests thinking about this as an approximation to the conditional approach.
- But... we've already thought a lot about these two different approaches.
- And they're quite different.

References

Resources

- If you want to read about SHAP from the original authors, the presentation in [LEC+20] is much more clear than the original [LL17].
- There are a huge number of papers discussing and building on SHAP. One recent paper that connects to many of the on-manifold/off-manifold issues we've discussed is [FMB+20].

[FMB+20] Christopher Frye, Damien de Mijolla, Tom Begley, Laurence Cowton, Megan Stanley, and Ilya Feige, *Shapley explainability on the data manifold*, CoRR (2020).

[LEC+20] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee, *From local explanations to global understanding with explainable ai for trees*, Nature Machine Intelligence **2** (2020), no. 1, 56–67.

[LL17] Scott Lundberg and Su-In Lee, *A unified approach to interpreting model predictions*, 2017, pp. 4765–4774.

[RSG16a] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, *Local interpretable model-agnostic explanations (lime): An introduction*, Aug 2016, https://www.oreilly.com/content/introduction-to-local-interpretable-model-agnostic-explanations-lime/

References II

[RSG16b] ______, *"why should i trust you?": Explaining the predictions of any classifier*, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (New York, NY, USA), KDD '16, Association for Computing Machinery, 2016, pp. 1135–1144.