Local Interpretation and LIME

David S. Rosenberg

Bloomberg ML EDU

December 1, 2021

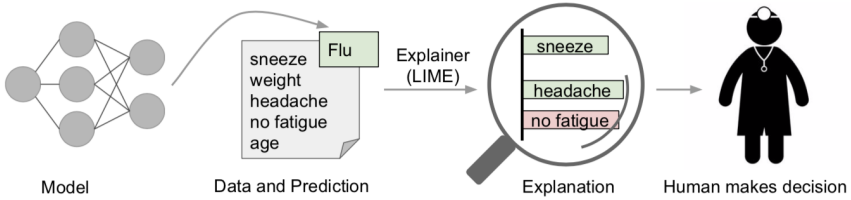
Contents

Introduction

Local Feature Importance

- I want to know ***why*** a particular prediction was made.
- Don't care that x_1(say age) is generally important to the prediction function.
- I want to know why $f(x) =$ DENY LOAN.
- e.g. Perhaps when $x_2 =$ Bad Credit, x_1 is irrelevant.
- If my loan was denied, I may also want to know
  - what can I change to improve my score, according to your function f?
- The methods we talk about today are about local feature importance
  - But they can be aggregated back to global importance scores

Idea of local interpretability



- Model makes prediction for diagnosis given observations.
- "Explanation" highlights symptoms that provide most evidence for and against the prediction.
- Helps human decide whether to trust prediction.
- By looking at many such explanations, can we build trust in the model?

Figure 1 from [RSG16b].

"Why Should I Trust You?"
Explaining the Predictions of Any Classifier

Marco Tulio Ribeiro
University of Washington
Seattle, WA 98105, USA
marcotcr@cs.uw.edu

Sameer Singh
University of Washington
Seattle, WA 98105, USA
sameer@cs.uw.edu

Carlos Guestrin
University of Washington
Seattle, WA 98105, USA
guestrin@cs.uw.edu

- This is the paper that introduced LIME [RSG16b].
- A popular **local** interpretation method.
- In other words, explain the prediction for a particular instance $x \in \mathcal{X}$.
- The paper is actually a bit difficult to understand, but we'll unpack it here.

Will the real LIME please stand up?

- First was the original LIME paper [RSG16b].
  - Presented a general approach for local interpretation...
  - But only really fleshed out the method for image and text inputs.
  - My thought: when it works, seems useful!
- The associated GitHub repo later added support for tabular data.
  - My thought: not as natural or compelling as for images / text
- Explanatory blogs and books (e.g. [Mol19, Tha21])
  - often introduce LIME for the tabular case, and give reasonable critique,
  - but not obvious the same critiques are as relevant for images and text

LIME for vision

Interpreting object detection

- Suppose we have a trained model $f(x)$ that predicts $\mathbb{P}(\text{frog})$.
- We want to explain the prediction for the following image:

Original Image

- One strategy: cover up various parts of the image, and see effect on prediction for $\mathbb{P}(\text{frog})$.
- Parts covered up should be "interpretable components".

Image from [RSG16a].

Interpretable components

Original Image

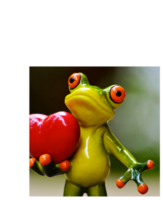
Interpretable Components

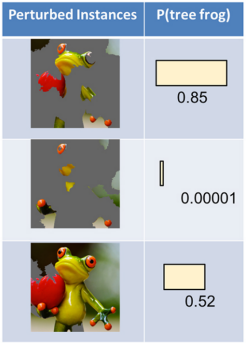
- A separate segmentation algorithm has broken the image into "interpretable components".

Image from [RSG16a].

On the left is the original image. On the right we see the image partitioned into regions. This partitioning is done using an independent algorithm designed to find "superpixels", which are groups of pixels that are perceptually related.

How does covering up parts affect predictions?





- $\mathbb{P}(\text{tree frog})$ column are the values predicted by model $f(x)$ on the perturbed instances.
- Can we build a model to predict $f(x)$ based on which parts are covered up?
- Then we can interpret this model to see which parts are the most important.

Image from [RSG16a].

- The perturbed instances are attained by covering up various interpretable components (i.e. superpixels) of the original image.
- The $\mathbb{P}(\text{tree frog})$ column is generated by our original prediction function $f(x)$ applied to these perturbed images.

Simplified feature representations

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

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

Image from [RSG16a].

LIME for vision

We want to interpret $f(x) \approx \mathbb{P}(\text{frog} \mid X = x)$ at $f(x_0)$.

1. Segment image x_0 into interpretable components.
2. Generate "perturbations" of x_0 with various components replaced by background color.
3. Use 0/1 indicator vector to indicate which parts of image x_0 are "on"
4. Each perturbation of x_0 can be represented by $x \in \mathcal{X}$ or $x' \in \{0,1\}^M$.
5. Make labeled training point $(x', \text{logit}[f(x)])$ for each perturbed image.
6. Train [weighted, sparse] linear regression to predict the logit of $f(x)$ from x'.
   - can use a smaller weight for examples that are larger perturbations of x_0.
7. "Explanation" of $f(x_0)$ is from the features with the largest positive weights.
   - These features correspond to image components most important for predicting $f(x_0)$.

- Recall that the logit function is $\text{logit}(p) = \ln\left(\frac{p}{1-p}\right)$ for $p \in (0,1)$. We use it for mapping from the [bounded] probability domain to a domain more appropriate for regression.
- We could also have done logistic regression with soft labels.

Interpreting predictions for frog picture

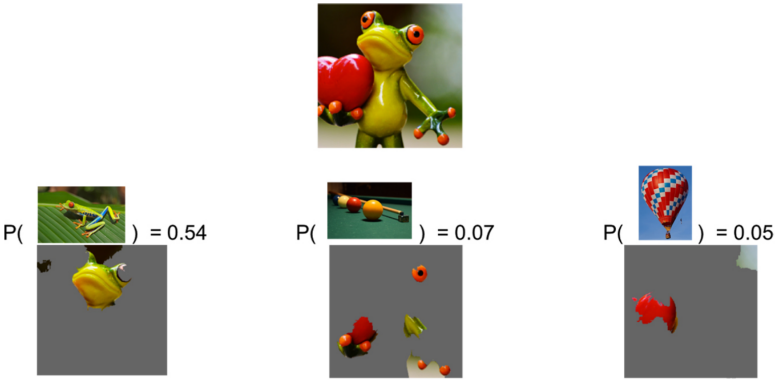
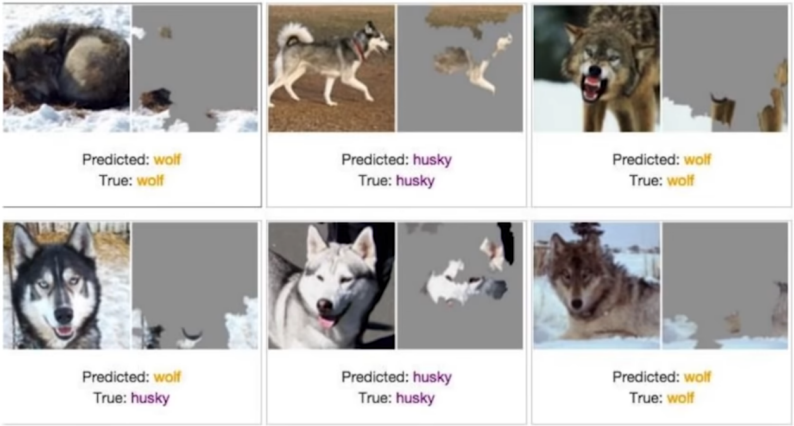


Figure 6. Explanation for a prediction from Inception. The top three predicted classes are "tree frog," "pool table," and "balloon." Sources: Marco Tulio Ribeiro, Pixabay (frog, billiards, hot air balloon).

Image from [RSG16a].

- Here we evaluate our prediction function $f(x)$ on x_0, which his the image of the frog with the heart.
- The prediction function gives a probability distribution over many image classes.
- Here are the image segments that most support the classes "tree frog", "pool table", and "balloon", respectively, from left to right.
- To get these explanations, we needed to
  - create one interpretable representation based on x_0 (including the segmentation) and
  - a separate model for each of the three classes.

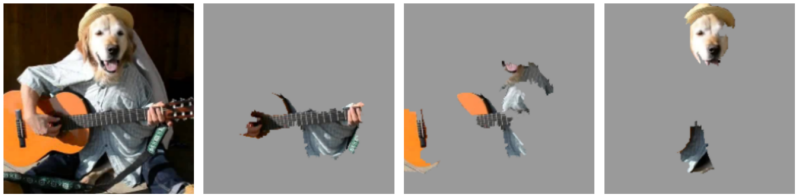
LIME: For finding generalization issues



Husky vs wolf? or "snow detector" attribution.

From [Kul17] slide 32.

Interpretation for dog playing guitar



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google's Inception neural network. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), "Acoustic guitar" ($p = 0.24$) and "Labrador" ($p = 0.21$)

Figure 4 from [RSG16b].

LIME for choosing classifiers[1]

Original Image

"Bad" Classifier

"Good" Classifier

- Presumably we'd be prefer the "Good" classifier.

[1] Slide 59 from Singh's "Explaining Black-Box Machine Learning Predictions" presentation

LIME (generic paper version)

LIME: general idea

- Given: prediction function f
- Given: instance $x_0 \in \mathcal{X}$ to be explained
- Build "interpretable" / "white box" model $\hat{f}$ that approximates f near x_0.
- Explain prediction $f(x_0)$ by explaining prediction $\hat{f}(x_0)$.
- How this is implemented varies by input domain.

What's interpretable, really?

- Some standard "white box" / interpretable models are
  - linear models
  - tree models
  - generalized additive models
- But for these to be interpretable, the features must be interpretable.
  - No word embeddings.
  - No pixels.
  - No complicated derived features or principal components.
  - etc...

Interpretable data representations

- [RSG16b] presents the idea of
  - "interpretable data representations"
- We'll also call them interpretable features.
- The SHAP paper [LL17] refers to them as "simplified features".
- [RSG16b] suggests that interpretable features should be binary.
- If $x \in \mathcal{X}$ is the original representation of some instance,
  - we'll use $x' \in \{0,1\}^{d'}$ as the **interpretable representation**.

Local interpretable representations

- Our main goal is to interpret $f(x)$ for a particular $x = x_0$.
- We can restrict our attention to f in a "local" neighborhood of x_0.
  - Neighborhood is often defined in terms of small "perturbations" of x_0.
- Need an interpretable representation in a neighborhood of x_0,
  - but **we don't need an interpretable representation that's defined globally**.
- Often the instance of interest x_0 is represented as $(1, \ldots, 1)^{d'}$,
  - and other elements of $\{0,1\}^{d'}$ represent various perturbations of x_0.
- Think of $x_0 \in \mathcal{X}$ as having d' interpretable aspects or components, and
  - get perturbations of x_0 by removing some of these components.

Formalizing the interpretable representation

- Define a mapping $h_{x_0} : \{0,1\}^{d'} \to \mathcal{X}$ such that
  - $h_{x_0}((1,1,\ldots,1)) = x_0$.
  - for any $x' \in \{0,1\}^{d'}$, $h_{x_0}(x') \in \mathcal{X}$ is some "perturbation" of x_0.
- For example
  - $h_{x_0}(x')$ is an image with regions blocked out.
  - $h_{x_0}(x')$ is a sentence with words eliminated or modified.

Sampling perturbations

- In the LIME codebase,
  - perturbations are attained by uniform sampling of x' from $\{0,1\}^{d'}$.
- Then $h_{x_0}(x') \in \mathcal{X}$ will be some perturbed version of x in the original space.
- (Note: the approach in the original LIME paper [RSG16b] was different,
  - but their codebase has the version described, and that will be our definition of LIME.

- In [RSG16b] they sample perturbations as follows:
  - Sample k uniformly at random from $1, \ldots, d'$.
  - Randomly choose k interpretable components to keep.
  - So if we choose $k = 3$ and $d' = 5$, we may end up with

$$x' = (1, 1, 0, 1, 0)\,.$$

Proximity measure

- Introduce proximity measure $\pi_{x_0}(x)$.
- Measures how "similar" x and x_0 are.
- Both x and x_0 are in the original input space $\mathcal{X}$.
- For perturbed interpretable representation $x' \in \{0,1\}^{d'}$,
  - we create its image in input space $h(x') \in \mathcal{X}$
  - then compute its "similarity" to x_0 with $\pi_{x_0}(h(x'))$.
- For $\mathcal{X} = \mathbb{R}^d$, [RSG16b] uses a standard RBF kernel

$$\pi_{x_0}(x) = \exp\left(-D(x, x_0)^2/\sigma^2\right),$$

with any distance function D and $\sigma > 0$.

  - e.g. D can be Euclidean distance or cosine distance.

Fidelity measure

- The idea is for g to be a good approximation of f around x_0.
- How can we assess that?
- Let $\mathcal{D}'_{x_0} \subset \{0,1\}^{d'}$ be drawn uniformly at random.
- [RSG16b] uses a square-loss fidelity measure:

$$\mathcal{L}(f, g, \pi_{x_0}, \mathcal{D}'_{x_0}) = \sum_{x' \in \mathcal{D}'_{x_0}} \pi_{x_0}(h(x')) \left(f(h(x')) - g(x')\right)^2.$$

- In words: g's predictions on $\mathcal{D}'_{x_0}$ are close to
  - f's predictions on the input space versions of elements of $\mathcal{D}'_x$,
  - weighted by the similarity between the perturbations and x.

- Recalling that $\mathcal{D}'_{x_0}$ is sampled uniformly from $\{0,1\}^{d'}$, we can view $\mathcal{L}(f, g, \pi_{x_0}, \mathcal{D}'_{x_0})$ as an unbiased estimate of

$$\mathbb{E}_{x' \sim \text{Unif}\left(\{0,1\}^{d'}\right)} \pi_{x_0}(h(x')) \left(f(h(x')) - g(x')\right)^2 \quad = \quad \sum_{x' \in \{0,1\}^{d'}} \pi_{x_0}(h(x')) \left(f(h(x')) - g(x')\right)^2.$$

- Going further, if we assume $\pi_{x_0}(h(x')) \geqslant 0$, then we can think of LIME as minimizing $\mathbb{E}_{x' \sim p(x')} \left[\left(f(h(x')) - g(x')\right)^2 \right]$, where $p(x') = \frac{1}{k} \pi_{x_0}(h(x'))$ for some $k > 0$.
- In general, $p(x')$ may not be easy to sample from, and we can think of $\mathbb{E}_{x' \sim \text{Unif}\left(\{0,1\}^{d'}\right)} \pi_{x_0}(h(x')) \left(f(h(x')) - g(x')\right)^2$ as importance sampling approach to calculating it.
- When we discuss SHAP, we'll end up with an objective function of the exact same form, but where sampling from $p(x')$ is straightforward (the Kernel SHAP method).

Fitting g

- They take G to be the class of linear models

$$g(x') = w_g^T x'$$

  for $x' \in \{0,1\}^{d'}$ and $w_g \in \mathbb{R}^{d'}$.
- Create perturbation set $\mathcal{D}'_{x_0} \subset \{0,1\}^{d'}$.
- For each $x' \in \mathcal{D}'_{x_0}$ we get a training example $(x', f(h(x')))$ with weight $\pi_{x_0}(h(x'))$.
- Fit g with this weighted training data so that w_g is sparse.
  - e.g. use Lasso to get at most K nonzero components of w_g.
- Resulting weights w_g can be "interpreted".
  - Large weights correspond to interpretable components supporting the prediction.

LIME: tabular version (from code)

Tabular case

- [RSG16b] doesn't discuss the tabular case.
- But the LIME codebase supports it.
- What changes?
  - perturbations
  - the interpretable representation
- Perturbations are done separately for each column/feature.

Perturbations for categorical variables

- For a categorical column,
  - we get the relative frequencies of the categories in a training set.
- Column is perturbed by resampling according to those frequencies.
- The "interpretable representation" is a binary indicator
  - 1 if resampled column is same as original value of point we're trying to interpret
  - 0 otherwise

Perturbations for continuous variables (1)

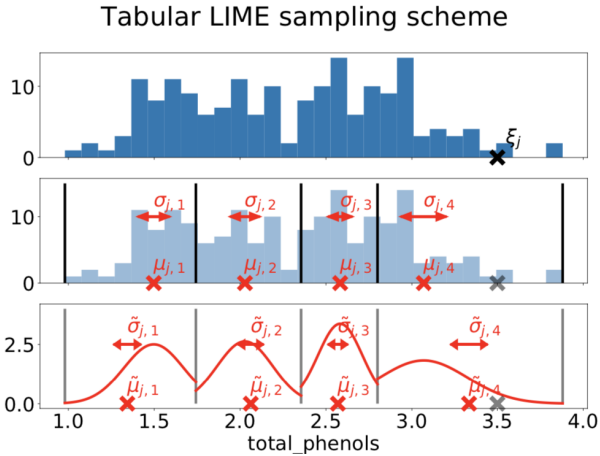
- Assume we have a training set.
- For each feature j,
  - Partition $\mathbb{R}$ into p parts using quantile binning of feature j in the training set.
  - The number of training points with feature j in a particular bin will be roughly the same for all bins.
- For each bin b for feature j,
  - compute the mean $\mu_{j,b}$ and variance $\sigma^2_{j,b}$ of jth feature values of training points in bin b.

Perturbations for continuous variables (2)

To generate a perturbed point,

- For each feature j,
  - Sample a bin b uniformly
  - Sample a value for feature j from $\mathcal{N}(\mu_{j,b}, \sigma^2_{j,b})$ **truncated** to the bin boundaries
- Let $x \in \mathbb{R}^d$ be a point sampled in this way.
  - Note the sampling had nothing to do with point x_0 of interest.
- The interpretable representation will be $x' \in \{0,1\}^d$, where
  - $x'_j = \mathbb{1}[x \text{ and } x_0 \text{ have their } j\text{'th feature in the same bin}]$

Continuous variable sampling scheme visualized[2]



[2]From [GL20, Fig. 2]

- What we're looking at here is for a single feature j.
- The top graph is a histogram of the values of feature j in the training set.
- The ξ_j marks the jth feature for the point x_0 that we want to interpret.
- In the second graph we see the quantile partitioning into 4 bins, as well as identification of the mean variance of feature values in each bin.
- In the bottom graph, we see the truncated Gaussian distributions that are drawn for from each bin, after first selecting a bin uniformly at random.
- This same procedure is repeated for all continuous features.

Surrogate model training set

- We sample n points using the scheme above.
- For every point we get a point $x \in \mathcal{X}$ and $x' \in \{0,1\}^d$.
  - Recall that the encoding x' depends on the point x_0.
- Our labeled training example for this point is $(x', f(x))$.
- We train a weighted linear regression model to get our surrogate model.
  - We can then interpret the feature weights
- What are the example weights used in training?

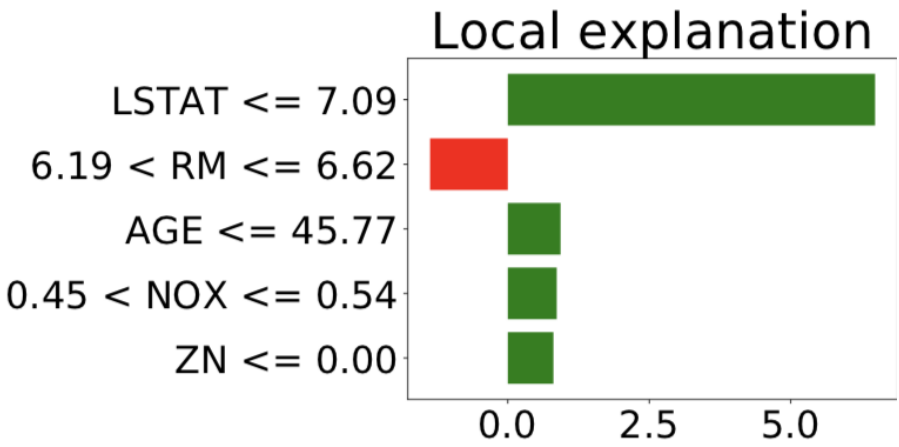
Weight samples by proximity to instance

- We want to build a model that's "local" to x_0, the point of interest.
- The Tabular LIME default is to define the weight on example $(x, x', f(x))$ by

$$\exp\left(-\frac{1}{2\nu^2}\sum_{j=1}^{d}(1-x_j')\right),$$

for ν some bandwidth parameter.

Typical output



- Each of these was originally a continuous variable.
- The ranges are coming from the feature quantile binning.
- Remember the input features to the surrogate model are binary indicators for whether the particular feature was in the range given.
- The takeaway here would be that LSTAT$\leqslant 7.09$ has strong a positive impact on the response.

References

Resources

- Authors of the LIME paper have a nice blog post [RSG16a].
- For understanding tabular LIME, the best resource besides the code in GitHub seems to be this paper [GL20], which is what our presentation is based on. Christoph Molnar's online book [Mol19], while generally quite useful, is pretty weak on LIME, and rather misleading for Tabular LIME (as of December 1, 2021).

References I

[GL20] Damien Garreau and Ulrike von Luxburg, *Looking deeper into tabular lime*, CoRR (2020).

[Kul17] Kasia Kulma, *Interpretable machine learning using lime framework*, https://www.slideshare.net/0xdata/interpretable-machine-learning-using-lime-framework-kasia-kulma-phd-da 2017, [Online; accessed 15-June-2021].

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

[Mol19] Christoph Molnar, *Interpretable machine learning*, bookdown.org, 2019, https://christophm.github.io/interpretable-ml-book/.

References II

[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/.

[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.

[Tha21] Ajay Thampi, *Interpretable ai*, Manning Publications, 2021, https://www.manning.com/books/interpretable-ai.