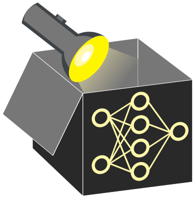
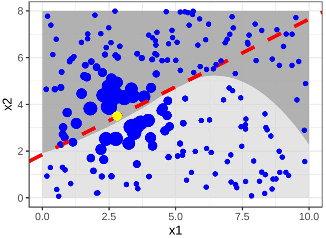
Interpretable Machine Learning



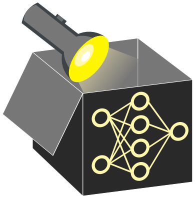
Local Interpretable Model-agnostic Explanations (LIME)



Learning goals

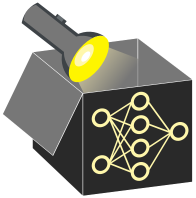
- Understand motivation for LIME
- Develop a mathematical intuition

LIME



- **Locality assumption:** $\hat{f}$ behaves similarly simple in small neighborhood of $\mathbf{x}$
  $\rightsquigarrow$ Approximate $\hat{f}$ near $\mathbf{x}$ using an interpretable surrogate model $\hat{g}$
- **Interpretation strategy:** Use $\hat{g}$'s simple internal structure to explain $\hat{f}(\mathbf{x})$ locally
  $\rightsquigarrow$ **Common surrogates:** Sparse linear models, shallow decision trees
- **Applicability:** Model-agnostic; supports tabular, image, and text data
- **In practice:** Generate samples near $\mathbf{x}$, predict with $\hat{f}$, and fit $\hat{g}$ to these samples using $\hat{f}$'s outputs as targets, weighting samples by their proximity/closeness to $\mathbf{x}$

LIME: CHARACTERISTICS



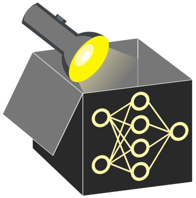
Definition: LIME provides a local explanation for a black-box model $\hat{f}$ in form of a surrogate model $\hat{g} \in \mathcal{G}$, where $\mathcal{G}$ is a class of interpretable models

Surrogate model $\hat{g}$ should satisfy two characteristics:

1. **Interpretable**: Provide human-understandable insights into the relationship between input features and prediction (e.g. via coefficients, model structure)
2. **Local fidelity** / **faithfulness**: $\hat{g}$ closely approximates $\hat{f}$ in the vicinity of the input $\mathbf{x}$ being explained

Goal: Find $\hat{g}$ with **minimal complexity and maximal local fidelity**

MODEL COMPLEXITY



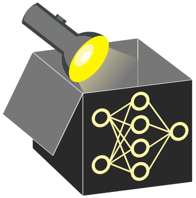
We can measure the complexity of $\hat{g} \in \mathcal{G}$ using a complexity measure $J : \mathcal{G} \to \mathbb{R}_0$

Example: (Sparse) Linear Models

- Let $\mathcal{G} = \left\{g : \mathcal{X} \to \mathbb{R} \mid g(\mathbf{x}) = s(\boldsymbol{\theta}^\top \mathbf{x})\right\}$ be the class of linear models
- $s(\cdot)$ is identity (linear model) or logistic sigmoid function (logistic regression)

$\rightsquigarrow$ $J(g) = \sum_{j=1}^{p} \mathcal{I}_{\{\theta_j \neq 0\}}$: Count number of non-zero coefficients (via L_0-norm of $\boldsymbol{\theta}$)

MODEL COMPLEXITY



We can measure the complexity of $\hat{g} \in \mathcal{G}$ using a complexity measure $J : \mathcal{G} \to \mathbb{R}_0$

Example: (Sparse) Linear Models

- Let $\mathcal{G} = \left\{g : \mathcal{X} \to \mathbb{R} \mid g(\mathbf{x}) = s(\boldsymbol{\theta}^\top \mathbf{x})\right\}$ be the class of linear models
- $s(\cdot)$ is identity (linear model) or logistic sigmoid function (logistic regression)

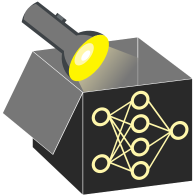
$\rightsquigarrow$ $J(g) = \sum_{j=1}^{p} \mathcal{I}_{\{\theta_j \neq 0\}}$: Count number of non-zero coefficients (via L_0-norm of $\boldsymbol{\theta}$)

Example: Decision Trees

- Let $\mathcal{G} = \left\{g : \mathcal{X} \to \mathbb{R} \mid g(\mathbf{x}) = \sum_{m=1}^{M} c_m \mathcal{I}_{\{\mathbf{x} \in Q_m\}}\right\}$ be the class of trees
- Q_m are disjoint axis parallel regions (leaves) and $c_m \in \mathbb{R}$ constant predictions

$\rightsquigarrow$ $J(g) = M$: Count number of terminal/leaf nodes

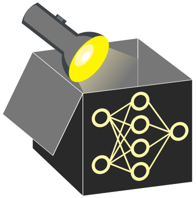
LOCAL FIDELITY OF SURROGATE MODELS



- Surrogate $\hat{g}$ is **locally faithful** to a black-box model $\hat{f}$ around an input $\mathbf{x}$ if

$$\hat{g}(\mathbf{z}) \approx \hat{f}(\mathbf{z}) \quad \text{for synthetic samples } \mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^p \text{ generated around } \mathbf{x}$$

LOCAL FIDELITY OF SURROGATE MODELS

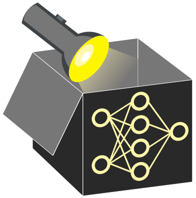


- Surrogate $\hat{g}$ is **locally faithful** to a black-box model $\hat{f}$ around an input $\mathbf{x}$ if

$$\hat{g}(\mathbf{z}) \approx \hat{f}(\mathbf{z}) \quad \text{for synthetic samples } \mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^p \text{ generated around } \mathbf{x}$$

- **Optimization principle:** The closer $\mathbf{z}$ is to $\mathbf{x}$, the more $\hat{g}(\mathbf{z})$ should match $\hat{f}(\mathbf{z})$

LOCAL FIDELITY OF SURROGATE MODELS



- Surrogate $\hat{g}$ is **locally faithful** to a black-box model $\hat{f}$ around an input $\mathbf{x}$ if

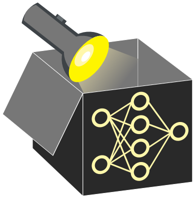
$$\hat{g}(\mathbf{z}) \approx \hat{f}(\mathbf{z}) \quad \text{for synthetic samples } \mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^p \text{ generated around } \mathbf{x}$$

- **Optimization principle:** The closer $\mathbf{z}$ is to $\mathbf{x}$, the more $\hat{g}(\mathbf{z})$ should match $\hat{f}(\mathbf{z})$
- To operationalize this optimization, we need:
  1. **A proximity (similarity) measure $\phi_{\mathbf{x}}(\mathbf{z})$** between $\mathbf{z}$ and $\mathbf{x}$, e.g.:

     $$\phi_{\mathbf{x}}(\mathbf{z}) = \exp(-d(\mathbf{x}, \mathbf{z})^2/\sigma^2) \text{ (exponential kernel), where}$$

     - d is a distance metric (e.g., Euclidean or Gower for mixed types)
     - σ is the kernel width that controls locality

LOCAL FIDELITY OF SURROGATE MODELS



- Surrogate $\hat{g}$ is **locally faithful** to a black-box model $\hat{f}$ around an input $\mathbf{x}$ if

$$\hat{g}(\mathbf{z}) \approx \hat{f}(\mathbf{z}) \quad \text{for synthetic samples } \mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^p \text{ generated around } \mathbf{x}$$

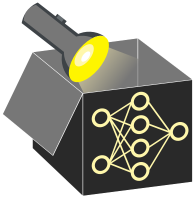
- **Optimization principle:** The closer $\mathbf{z}$ is to $\mathbf{x}$, the more $\hat{g}(\mathbf{z})$ should match $\hat{f}(\mathbf{z})$
- To operationalize this optimization, we need:
  1. **A proximity (similarity) measure** $\phi_{\mathbf{x}}(\mathbf{z})$ between $\mathbf{z}$ and $\mathbf{x}$, e.g.:

     $$\phi_{\mathbf{x}}(\mathbf{z}) = \exp(-d(\mathbf{x}, \mathbf{z})^2/\sigma^2) \text{ (exponential kernel), where}$$

     - d is a distance metric (e.g., Euclidean or Gower for mixed types)
     - σ is the kernel width that controls locality
  2. **A loss function** $L(\hat{f}(\mathbf{z}), \hat{g}(\mathbf{z}))$, e.g. the L_2 loss/squared error:

$$L(\hat{f}(\mathbf{z}), \hat{g}(\mathbf{z})) = \left(\hat{g}(\mathbf{z}) - \hat{f}(\mathbf{z})\right)^2$$

LOCAL FIDELITY OF SURROGATE MODELS



- Surrogate $\hat{g}$ is **locally faithful** to a black-box model $\hat{f}$ around an input $\mathbf{x}$ if

$$\hat{g}(\mathbf{z}) \approx \hat{f}(\mathbf{z}) \quad \text{for synthetic samples } \mathbf{z} \in \mathcal{Z} \subseteq \mathbb{R}^p \text{ generated around } \mathbf{x}$$

- **Optimization principle:** The closer $\mathbf{z}$ is to $\mathbf{x}$, the more $\hat{g}(\mathbf{z})$ should match $\hat{f}(\mathbf{z})$
- To operationalize this optimization, we need:
  1. **A proximity (similarity) measure** $\phi_{\mathbf{x}}(\mathbf{z})$ between $\mathbf{z}$ and $\mathbf{x}$, e.g.:

     $$\phi_{\mathbf{x}}(\mathbf{z}) = \exp(-d(\mathbf{x}, \mathbf{z})^2/\sigma^2) \text{ (exponential kernel), where}$$

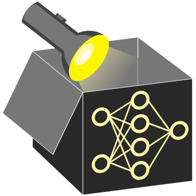
     - d is a distance metric (e.g., Euclidean or Gower for mixed types)
     - σ is the kernel width that controls locality
  2. **A loss function** $L(\hat{f}(\mathbf{z}), \hat{g}(\mathbf{z}))$, e.g. the L_2 loss/squared error:

     $$L(\hat{f}(\mathbf{z}), \hat{g}(\mathbf{z})) = \left(\hat{g}(\mathbf{z}) - \hat{f}(\mathbf{z})\right)^2$$

- The overall **local fidelity objective** is measured by a weighted loss:

$$L(\hat{f}, \hat{g}, \phi_{\mathbf{x}}) = \sum_{\mathbf{z} \in \mathcal{Z}} \phi_{\mathbf{x}}(\mathbf{z}) \cdot L(\hat{f}(\mathbf{z}), \hat{g}(\mathbf{z}))$$

LIME OPTIMIZATION TASK

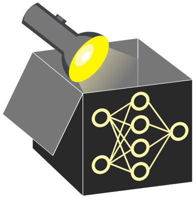


- Optimization problem of LIME:

$$\arg\min_{\hat{g}\in\mathcal{G}} L(\hat{f}, \hat{g}, \phi_{\mathbf{x}}) + J(\hat{g})$$

- **In practice** LIME uses a two-stage approach:
  1. User specifies complexity $J(\hat{g})$ beforehand (e.g., LASSO with k features)
  2. Optimize $L(\hat{f}, \hat{g}, \phi_{\mathbf{x}})$ (model fidelity) for fixed complexity
- **Goal:** Build a **model-agnostic** explainer
  - $\rightsquigarrow$ Optimize $L(\hat{f}, \hat{g}, \phi_{\mathbf{x}})$ without making any assumptions on the form of $\hat{f}$
  - $\rightsquigarrow$ Surrogate $\hat{g}$ approximates $\hat{f}$ locally through sampling and fitting

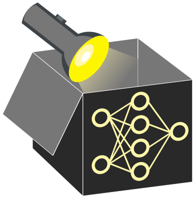
LIME ALGORITHM: OUTLINE ▸ Ribeiro. 2016



Input:

- Pre-trained black-box model $\hat{f}$
- Observation $\mathbf{x}$ whose prediction $\hat{f}(\mathbf{x})$ we want to explain
- Interpretable model class $\mathcal{G}$ for local surrogate (to limit complexity)

LIME ALGORITHM: OUTLINE ▸ Ribeiro. 2016



Input:

- Pre-trained black-box model $\hat{f}$
- Observation $\mathbf{x}$ whose prediction $\hat{f}(\mathbf{x})$ we want to explain
- Interpretable model class $\mathcal{G}$ for local surrogate (to limit complexity)

Algorithm:

1. Independently sample new points $\mathbf{z} \in \mathcal{Z}$
2. Retrieve predictions $\hat{f}(\mathbf{z})$ for obtained points $\mathbf{z}$
3. Weight $\mathbf{z} \in \mathcal{Z}$ by their proximity $\phi_{\mathbf{x}}(\mathbf{z})$ to quantify closeness to $\mathbf{x}$
4. Train interpretable surrogate model $\hat{g}$ on data points $\mathbf{z} \in \mathcal{Z}$ using weights $\phi_{\mathbf{x}}(\mathbf{z})$
   $\rightsquigarrow$ Predictions $\hat{f}(\mathbf{z})$ are used as target of this model
5. Return $\hat{g}$ as the local explanation for $\hat{f}(\mathbf{x})$

LIME ALGORITHM: EXAMPLE

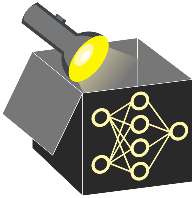
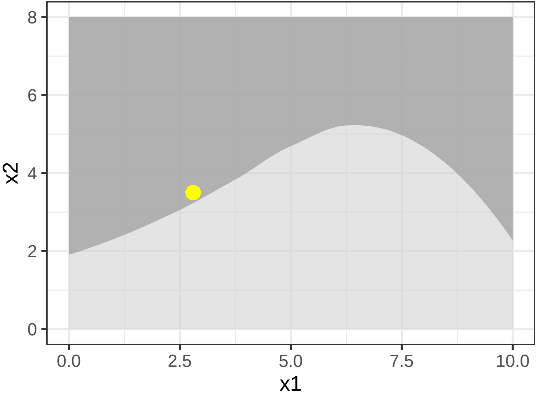
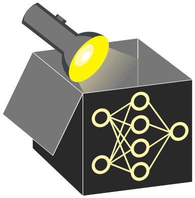


Illustration of LIME based on a classification task:

- Light/dark gray background: prediction surface of a classifier
- Yellow point: $\mathbf{x}$ to be explained
- $\mathcal{G}$: class of logistic regression models



LIME ALGORITHM: EXAMPLE (STEP 1+2: SAMPLING)



Strategies for sampling:

- Uniformly sample new points from the feasible feature range
- Use the training data set with or without perturbations
- Draw samples from the estimated univariate distribution of each feature
- Create an equidistant grid over the supported feature range

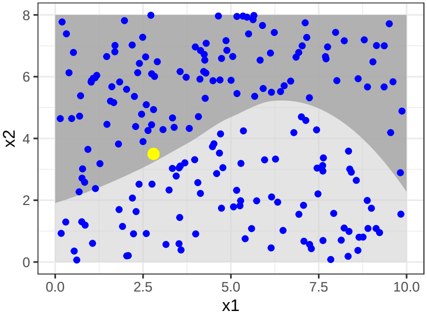


Figure: Uniformly sampled

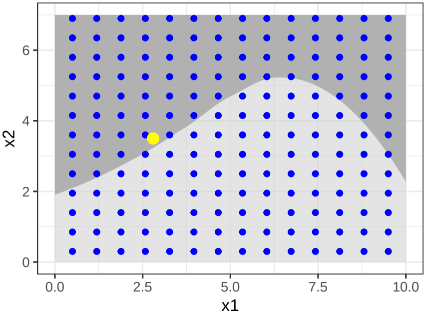
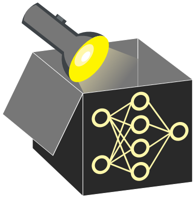


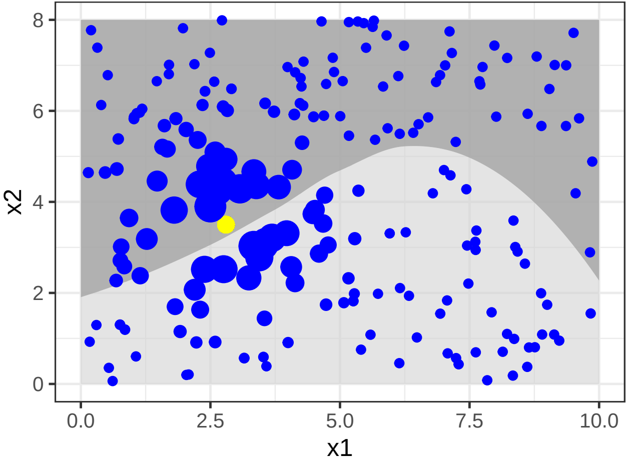
Figure: Equidistant grid

LIME ALGORITHM: EXAMPLE (STEP 3: PROXIMITY)



In this example, we use the exponential kernel defined on the Euclidean distance d

$$\phi_{\mathbf{x}}(\mathbf{z}) = exp(-d(\mathbf{x}, \mathbf{z})^2/\sigma^2).$$



LIME ALGORITHM: EXAMPLE (STEP 4: SURROGATE)

In this example, we fit a **logistic regression** model
$\rightsquigarrow L(\hat{f}(\mathbf{z}), \hat{g}(\mathbf{z}))$ is the Bernoulli loss