Local Model-agnostic Methods

2023.01.31

Jiyeon Han
KAIST Graduate School of AI

KAIST XAI Tutorial Series
2023. 1. 26 – 2. 16

Slides courtesy of AI702@KAIST GSAI

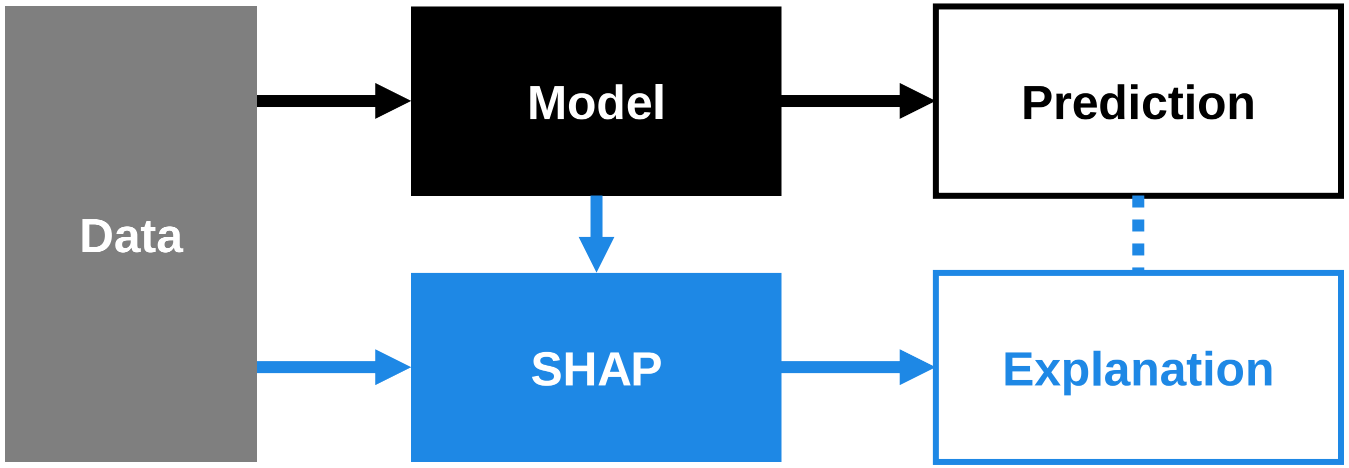
Overview

1. Local Model-Agnostic Approaches
2. LIME (Local Interpretable Model-agnostic Explanations)
3. Shapley Value
4. SHAP

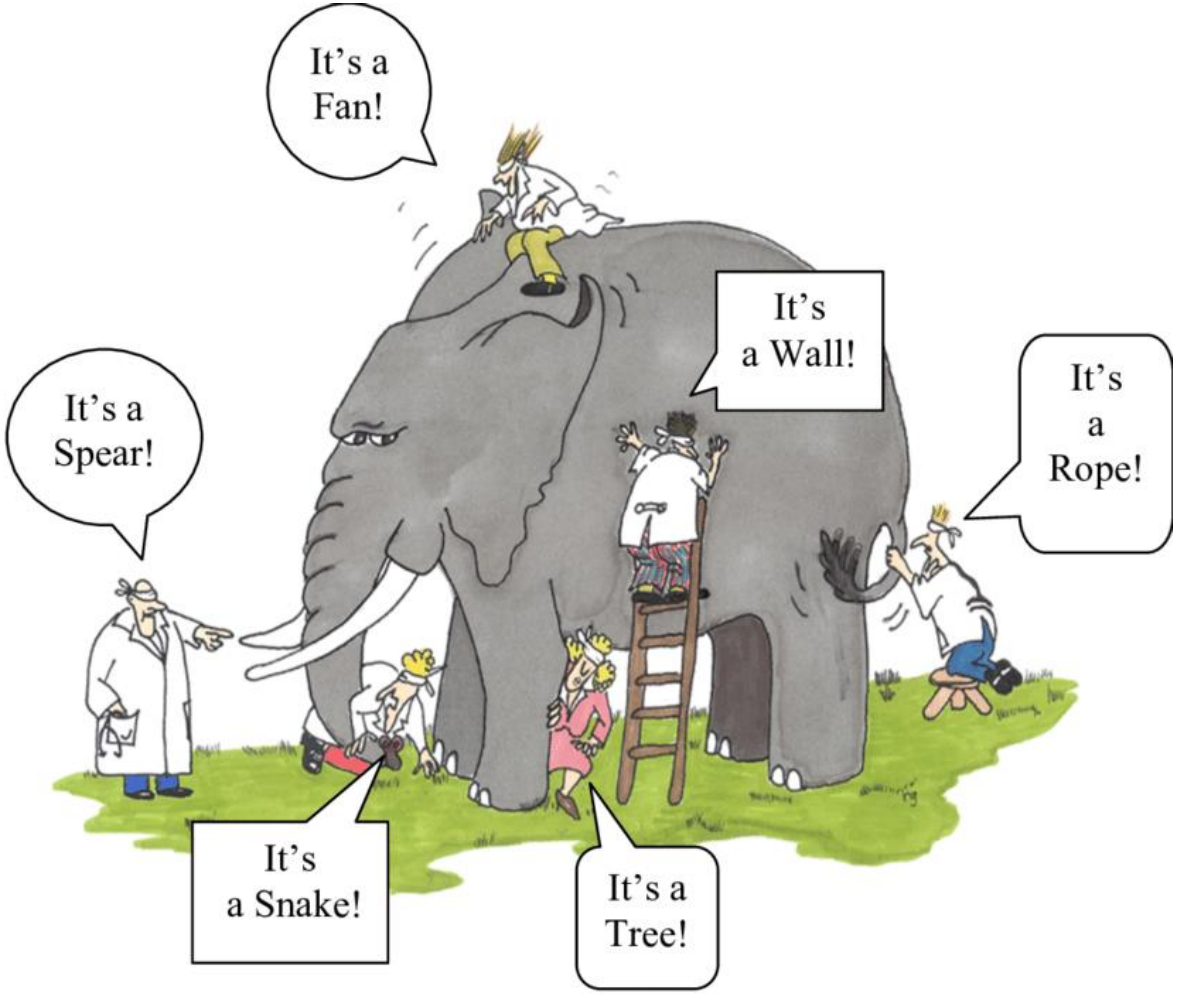
1. Model Agnostic Approaches

Model Agnostic Approaches

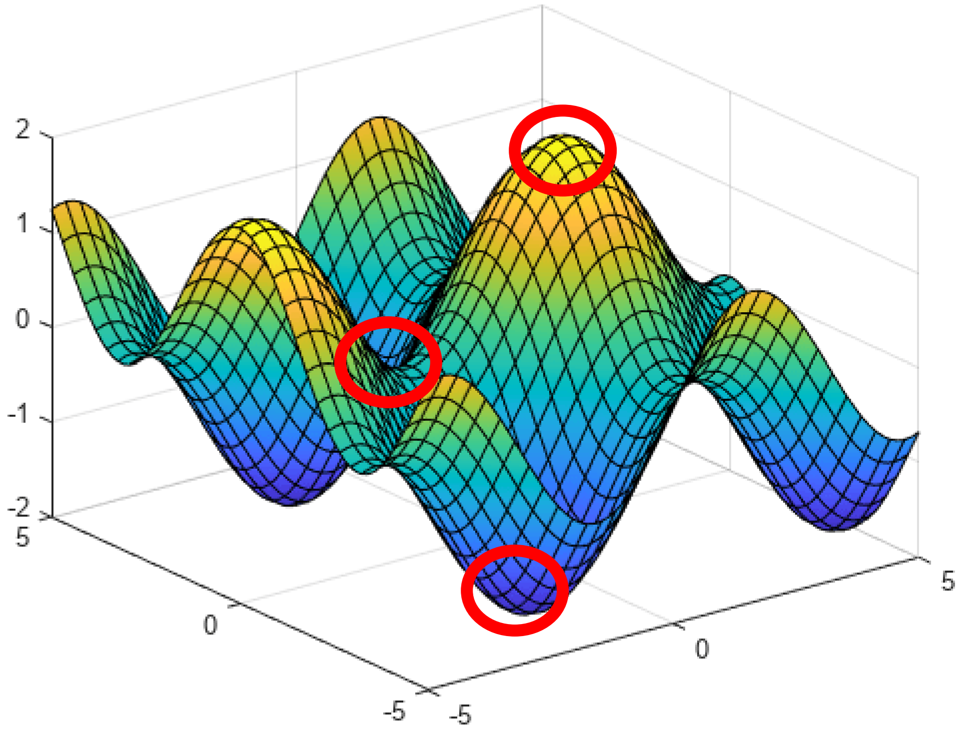
- Given
  - A already trained model (e.g., modern machine learning models)
  - A set of multi-featured data points (training or validation)
- Goal:
  - Compute the contributions of individual features of a data point



Model Agnostic Approaches



Model Agnostic Approaches



Source: https://kr.mathworks.com/help/matlab/ref/fsurf.html

2. LIME

LIME Local Interpretable Model-agnostic Explanations

- Mathematical formulation

$$\text{explanation}(x) = \arg\min_{g \in G} [L(f, g, \pi_x) + \Omega(g)]$$

loss function

original function

explainable model
(Lasso or Decision Tree)

proximity measure
(how large the neighborhood around instance x?)

model complexity

The recipe for training in LIME

1. **Select your instance** of interest for which you want to have an explanation of its black box prediction.
2. **Perturb your dataset** and get the black box predictions for these new points.
3. **Weight** the new samples according to their **proximity** to the instance of interest.
4. **Train a weighted, interpretable model** on the dataset with the variations.
5. **Explain** the prediction by interpreting the **local** model.

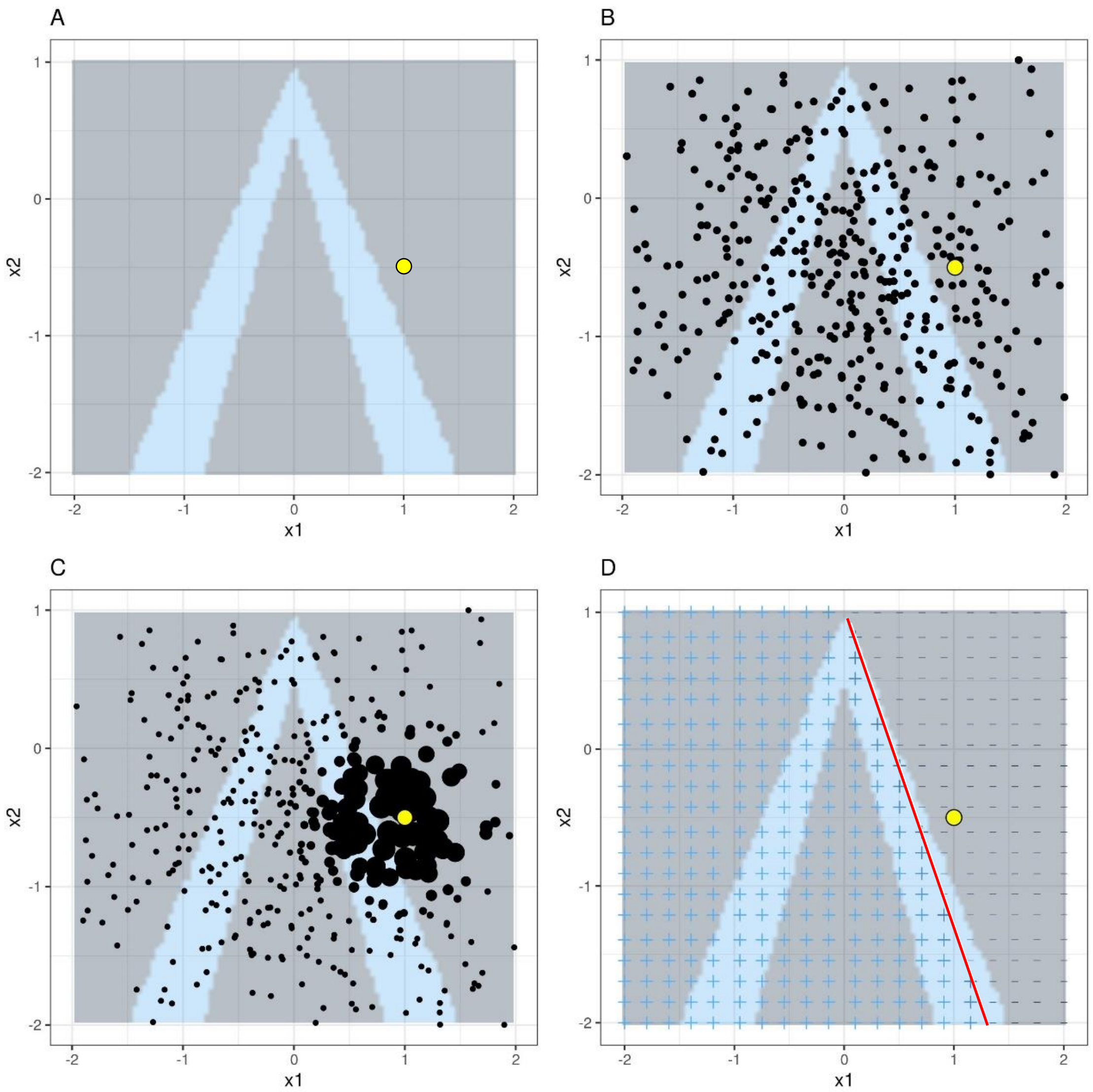
Then, how do you get the variations of the data?

https://christophm.github.io/interpretable-ml-book/lime.html

Example 1: LIME for Tabular Data

1. **Select your instance** of interest for which you want to have an explanation of its black box prediction.

2. **Perturb your dataset** and get the black box predictions for these new points.



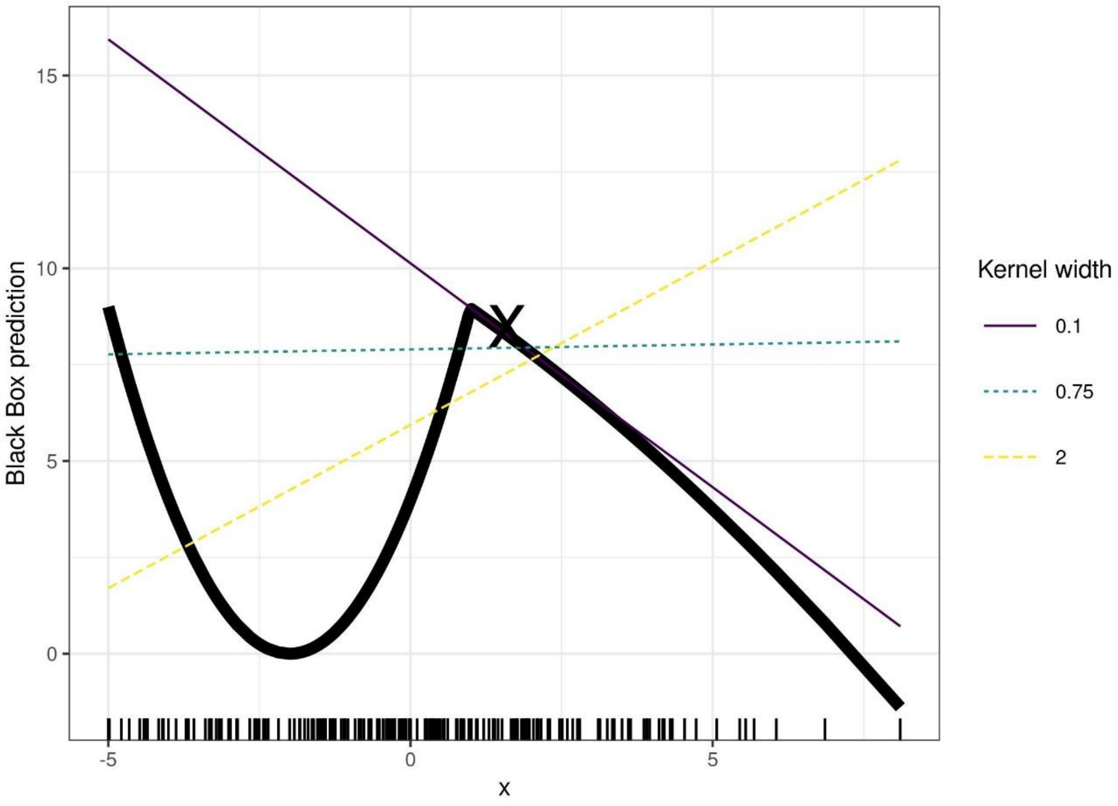
3. **Weight** the new samples according to their **proximity** to the instance of interest.

4. **Train a weighted, interpretable model** on the dataset with the variations.

https://christophm.github.io/interpretable-ml-book/lime.html

Example 1: LIME for Tabular Data

- LIME depends on kernel width: How do we set the neighborhood?



https://christophm.github.io/interpretable-ml-book/lime.html

Example 2: LIME for Text Data

- Classify YouTube comments as spam or normal

- How to perturb
  - Randomly **remove words** and observe the results!
  - **Weight** is calculated as 1-(1/# of removed words)

	CONTENT	CLASS
267	PSY is a good guy	0
173	For Christmas Song visit my channel! ;)	1

For	Christmas	Song	visit	my	channel!	;)	prob	weight
1	0	1	1	0	0	1	0.17	0.57
0	1	1	1	1	0	1	0.17	0.71
1	0	0	1	1	1	1	0.99	0.71
1	0	1	1	1	1	1	0.99	0.86
0	1	1	1	0	0	1	0.17	0.57

https://christophm.github.io/interpretable-ml-book/lime.html

Example 2: LIME for Text Data

- Classify YouTube comments as spam or normal

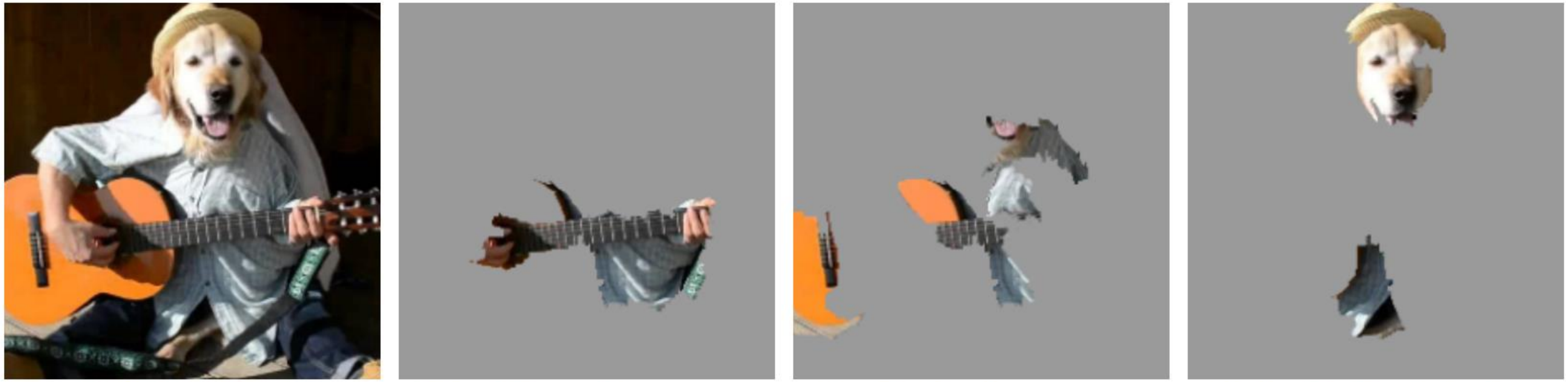
case	label_prob	feature	feature_weight
1	0.1701170	is	0.000000
1	0.1701170	good	0.000000
1	0.1701170	a	0.000000
2	0.9939024	channel!	6.180747
2	0.9939024	;)	0.000000
2	0.9939024	visit	0.000000

LIME algorithm shows that the word “channel” indicates a high probability of spam.

https://christophm.github.io/interpretable-ml-book/lime.html

Example 3: LIME for Images

- Explaining an image classification prediction made by neural Google's Inception neural network



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

Image regions are selected by the superpixel methods

Pros and Cons for LIME

Pros:

1. LIME is model-agnostic
2. Explanations are human-friendly
3. It works for tabular data, text and images
4. The fidelity measure proves the reliability of the interpretable model
5. Very easy to use
6. Other interpretable features are able to be used instead of original model features

Cons:

1. Finding a good neighborhood is unsolved problem
2. Sampling can be wrong (e.g. Gaussian)
3. The complexity should be pre-defined
4. Explanations can be instable

https://christophm.github.io/interpretable-ml-book/lime.html

3. Shapley Value

Shapley Values

- The Shapley value is the average marginal contribution of a feature value across all possible coalitions.

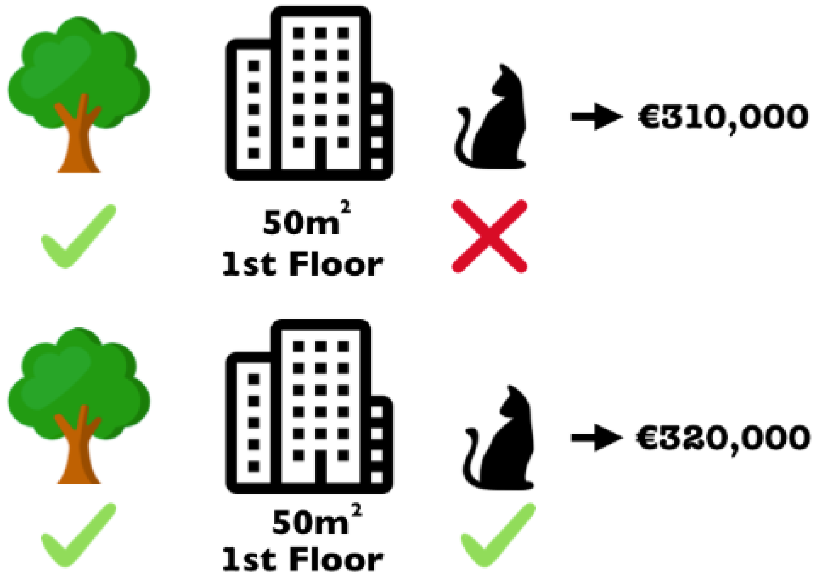


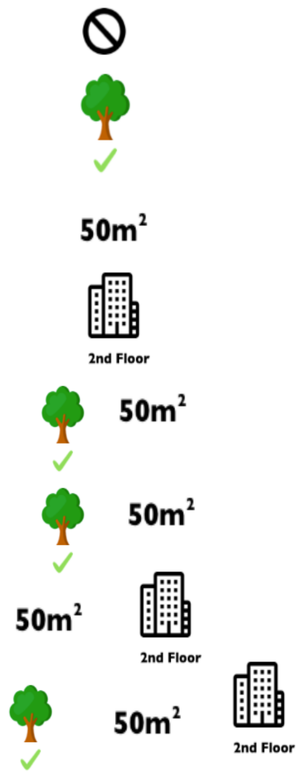
FIGURE 5.44: One sample repetition to estimate the contribution of `cat-banned` to the prediction when added to the coalition of `park-nearby` and `area-50`.

https://christophm.github.io/interpretable-ml-book/shapley.html

Shapley Values

- The Shapley value is the average marginal contribution of a feature value across all possible coalitions.

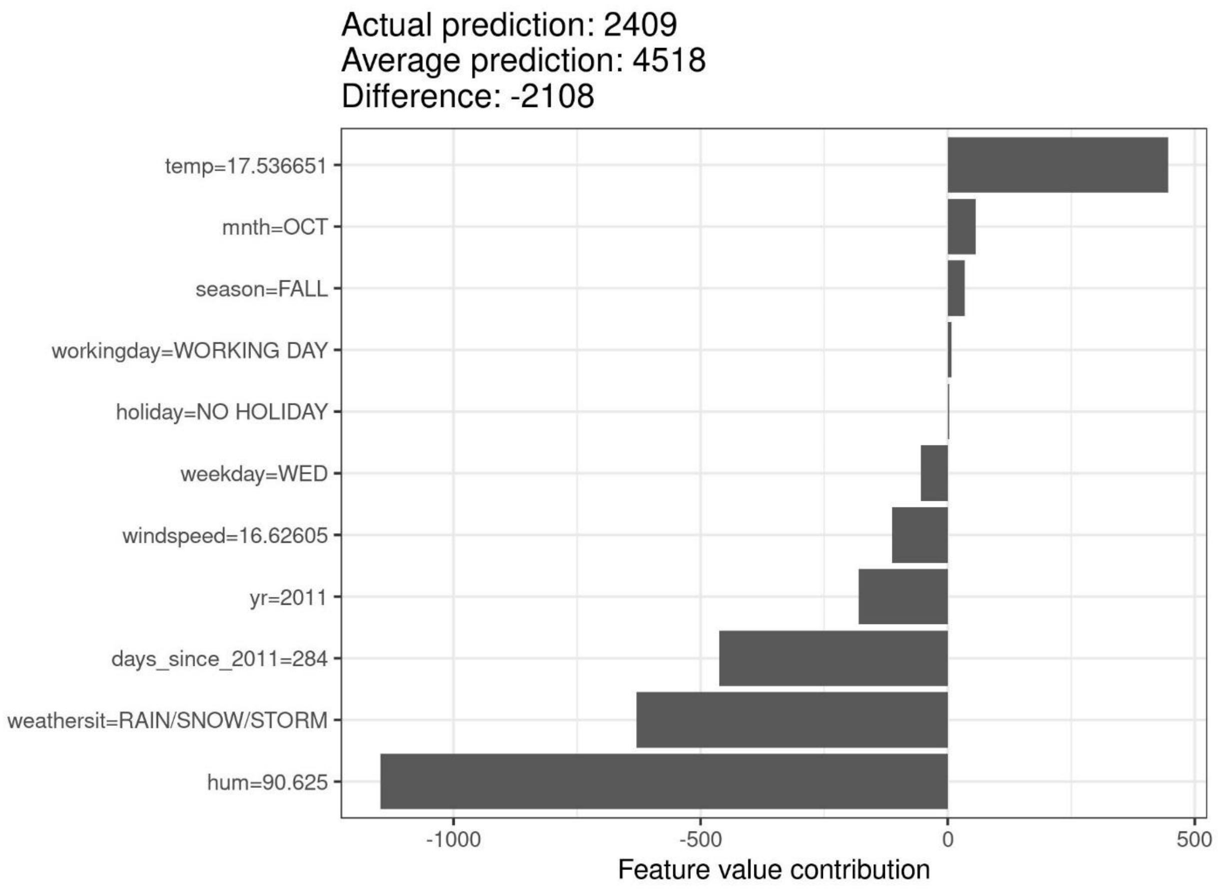
  - `No feature values`
  - `park-nearby`
  - `size-50`
  - `floor-2nd`
  - `park-nearby` + `size-50`
  - `park-nearby` + `floor-2nd`
  - `size-50` + `floor-2nd`
  - `park-nearby` + `size-50` + `floor-2nd`



https://christophm.github.io/interpretable-ml-book/shapley.html

Shapley Values

- Bike Rental Example



https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Definition

- The Shapley Value of a feature value is its contribution to the payout, weighted and summed over all possible feature value combinations

$val_x(S)$ is the prediction for feature values in set S that are marginalized over features that are not included in set S:

$$val_x(S) = \int \hat{f}(x_1, \ldots, x_p) d\mathbb{P}_{x \notin S} - E_X(\hat{f}(X))$$

Note that this is a function of S

$$val_x(S) = val_x(\{x_1, x_3\}) = \int_{\mathbb{R}} \int_{\mathbb{R}} \hat{f}(x_1, X_2, x_3, X_4) d\mathbb{P}_{X_2 X_4} - E_X(\hat{f}(X))$$

https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Definition

- The Shapley Value of a feature value is its contribution to the payout, weighted and summed over all possible feature value combinations

$$\phi_j(val) = \sum_{S \subseteq \{x_1,\ldots,x_p\} \setminus \{x_j\}} \underbrace{\frac{|S|!\,(p-|S|-1)!}{p!}}_{\text{weight}} \underbrace{\left(val\left(S \cup \{x_j\}\right) - val(S)\right)}_{\text{marginal contribution}}$$

https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Definition

- The Shapley Value of a feature value is its contribution to the payout, weighted and summed over all possible feature value combinations

Let $P_j = \{x_i \in \{x_1, \dots, x_p\} | \sigma(x_i) < \sigma(x_j)\}$.

$$\phi_j(val) = E_{\sigma \in \Pi(\{x_1,\dots,x_p\})}\left[val\left(P_j(\sigma \cup \{x_j\})\right) - val\left(P_j(\sigma)\right)\right]$$
$$= \frac{1}{n!}\left[val\left(P_j(\sigma \cup \{x_j\})\right) - val\left(P_j(\sigma)\right)\right]$$

$$\phi_j(val) = \sum_{S \subseteq \{x_1,\dots,x_p\} \setminus \{x_j\}} \frac{|S|!\,(p - |S| - 1)!}{p!}\left(val\left(S \cup \{x_j\}\right) - val(S)\right)$$

EX) $\sigma = x_2, x_6, x_7, x_3, x_1, x_4, x_5$

https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Properties

- The Shapley Value is the only attribution method that satisfies the properties 1) Efficiency, 2) Symmetry, 3) Dummy and 4) Additivity.

1) Efficiency

The feature contributions must add up to the difference of prediction for x and the average.

$$\sum_{j=1}^{p} \phi_j = \hat{f}(x) - E_X(\hat{f}(X))$$

https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Properties

- The Shapley Value is the only attribution method that satisfies the properties 1) Efficiency, 2) Symmetry, 3) Dummy and 4) Additivity.

2) Symmetry

The contributions of two feature values j and k should be the same if they contribute equally to all possible coalitions

If $val(S \cup \{x_j\}) = val(S \cup \{x_k\})$ for all $S \subseteq \{x_1, \dots, x_p\} \setminus \{x_j, x_k\}$,

$$\phi_j = \phi_k$$

https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Properties

- The Shapley Value is the only attribution method that satisfies the properties 1) Efficiency, 2) Symmetry, 3) Dummy and 4) Additivity.

3) Dummy

A feature j that does not change the predicted value (regardless of coalition) should have a Shapley value of 0

$$\phi_j = 0 \quad \text{If} \quad val(S \cup \{x_j\}) = val(S) \quad \text{for all} \quad S$$

https://christophm.github.io/interpretable-ml-book/shapley.html

The Shapley Value Properties

- The Shapley Value is the only attribution method that satisfies the properties 1) Efficiency, 2) Symmetry, 3) Dummy and 4) Additivity.

4) Additivity

For a game with combined payouts val+val^{+}, the respective Shapley values are as follows:

$$\phi_j + \phi_j^{+}$$

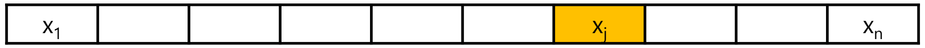
https://christophm.github.io/interpretable-ml-book/shapley.html

Estimating the Shapley Value

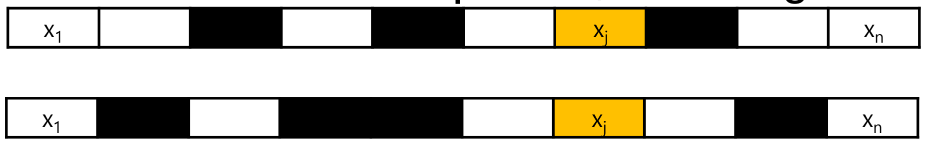
- Computing exact Shapley value is expensive
- Thus Monte-Carlo sampling is used in practice.

$$\phi_j(val) = \frac{1}{n!}\left[val\left(P_j(\sigma \cup \{x_j\})\right) - val\left(P_j(\sigma)\right)\right]$$

$$\hat{\phi}_j = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{f}(x_{+j}^m) - \hat{f}(x_{-j}^m)\right)$$



x_{+j}^m x with a random number of features values replaced by feature values from a random data point z, including for the value of feature j.



x_{-j}^m x with a random number of features values replaced by feature values from a random data point z, except for the value of feature j.

Estimating the Shapley Value

- The Pseudo Code for Estimating the Shapley Value

Approximate Shapley estimation for single feature value:

- Output: Shapley value for the value of the j-th feature
- Required: Number of iterations M, instance of interest x, feature index j, data matrix X, and machine learning model f
- For all m = 1,…,M:
  - Draw random instance z from the data matrix X
  - Choose a random permutation o of the feature values
  - Order instance x: $x_o = \left(x_{(1)}, \ldots, x_{(j)}, \ldots, x_{(p)}\right)$
  - Order instance z: $z_o = \left(z_{(1)}, \ldots, z_{(j)}, \ldots, z_{(p)}\right)$
  - Construct two new instances
    - With feature j: $x_{+j} = \left(x_{(1)}, \ldots, x_{(j-1)}, x_{(j)}, z_{(j+1)}, \ldots, z_{(p)}\right)$
    - Without feature j: $x_{-j} = \left(x_{(1)}, \ldots, x_{(j-1)}, z_{(j)}, z_{(j+1)}, \ldots, z_{(p)}\right)$
  - Compute marginal contribution: $\phi_j^m = \hat{f}\left(x_{+j}\right) - \hat{f}\left(x_{-j}\right)$
- Compute Shapley value as the average: $\phi_j(x) = \frac{1}{M} \sum_{m=1}^{M} \phi_j^m$

Pros and Cons for Shapley Value

Pros:

1. The prediction is fairly distributed among the features (no guarantee in LIME)
2. Contrastive Explanations are allowed
3. The Shapley value is the only explanation method with a solid theory
4. It is mind-blowing to explain a prediction as a game

Cons:

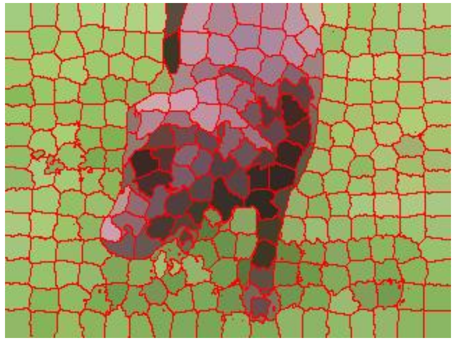
1. It requires a lot of computing time
2. Easy to be misinterpreted (It is NOT a feature value difference after removing the feature)
3. Always use all the features, thus not a selective explanation
4. Need access to the data
5. It suffers from inclusion of unrealistic data instances

4. SHAP

SHAP (Shapley Additive exPlanations)

- In SHAP, the Shapley value explanation is represented as an additive feature attribution method, a linear model.

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



g : explanation model
z' ∈ $\{0,1\}^M$: coalition vector (e.g. images in super-pixel level)
M : maximum coalition size
ϕ_j : feature attribution for a feature j, the Shapley values

Source: https://github.com/PSMM/SLIC-Superpixels

SHAP Properties

- SHAP describes the following three desirable properties:

1) Local accuracy

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

2) Missingness

$$z'_j = 0 \Rightarrow \phi_j = 0$$

3) Consistency

Let $f_x(z') = f\left(h_x(z')\right)$ and $z'_{\setminus j}$ indicates that $z'_j = 0$. For any two models f and f' that satisfy:

$$f'_x(z') - f'_x\left(z'_{\setminus j}\right) \geq f_x(z') - f_x(z'_{\setminus j})$$

for all inputs $z' \in \{0,1\}^M$, then:

$$\phi_j(f', x) \geq \phi_j(f, x)$$

Kernel SHAP: Example of h_x

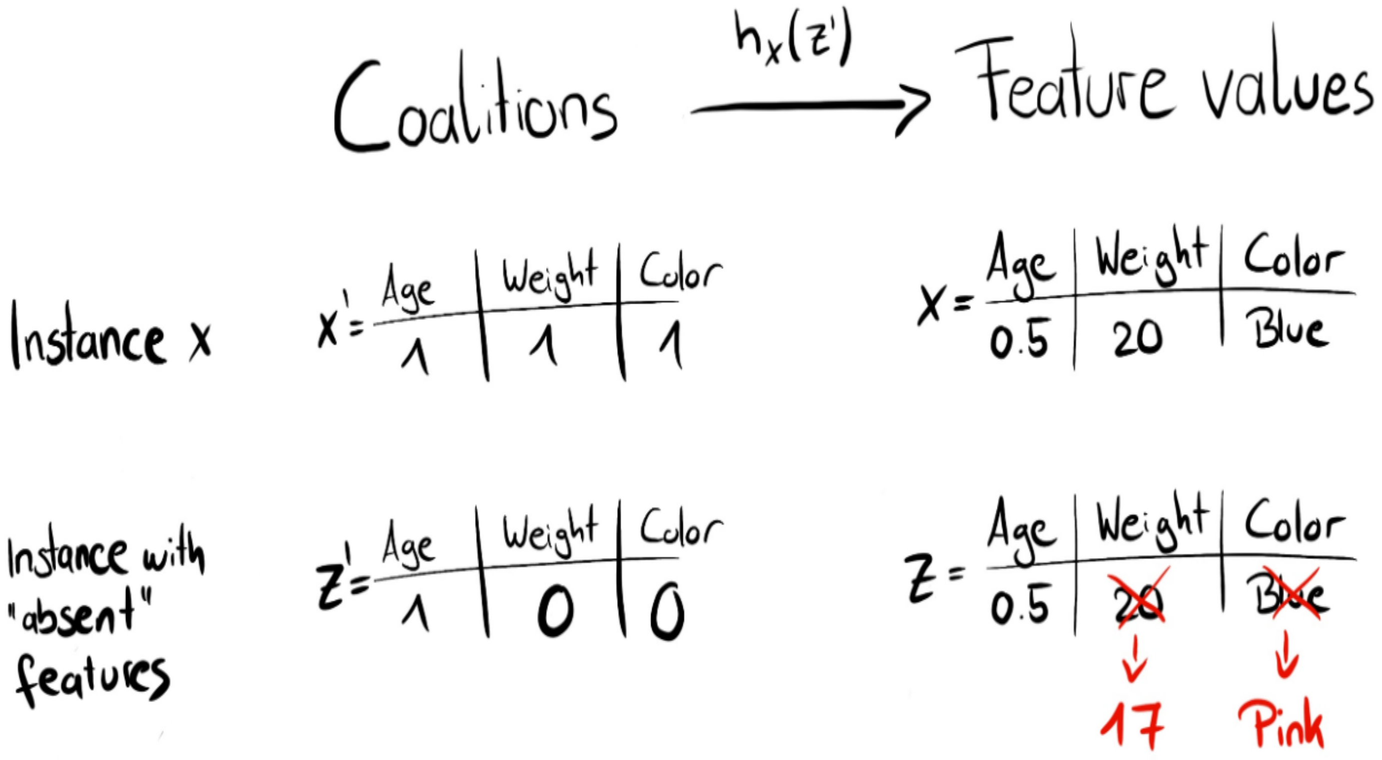


FIGURE 5.48: Function h_x maps a coalition to a valid instance. For present features (1), h_x maps to the feature values of x. For absent features (0), h_x maps to the values of a randomly sampled data instance.

Kernel SHAP: Example of h_x

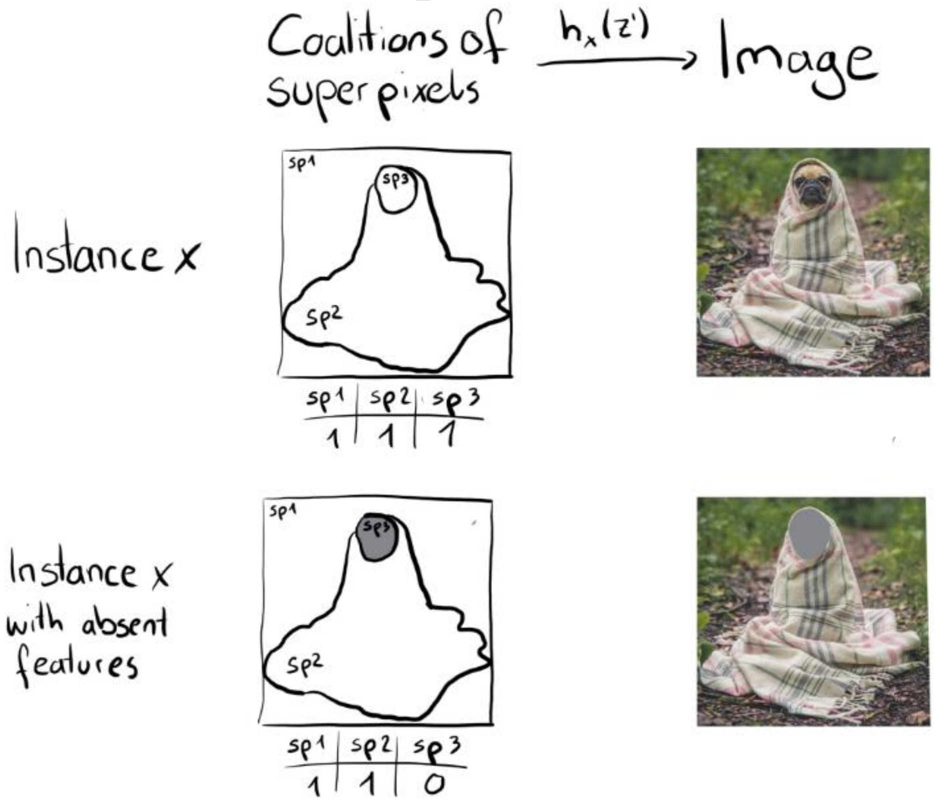


FIGURE 5.49: Function h_x maps coalitions of super pixels (sp) to images. Super-pixels are groups of pixels. For present features (1), h_x returns the corresponding part of the original image. For absent features (0), h_x greys out the corresponding area. Assigning the average color of surrounding pixels or similar would also be an option.

SHAP Optimization

$$\pi_x(x') = \frac{M-1}{\binom{M}{|z'|}|z'|(M-|z'|)}$$

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

$$L(f, g, \pi_x) = \sum_{z' \in Z} \left[f\left(h_x(z')\right) - g(z')\right]^2 \pi_x(z')$$

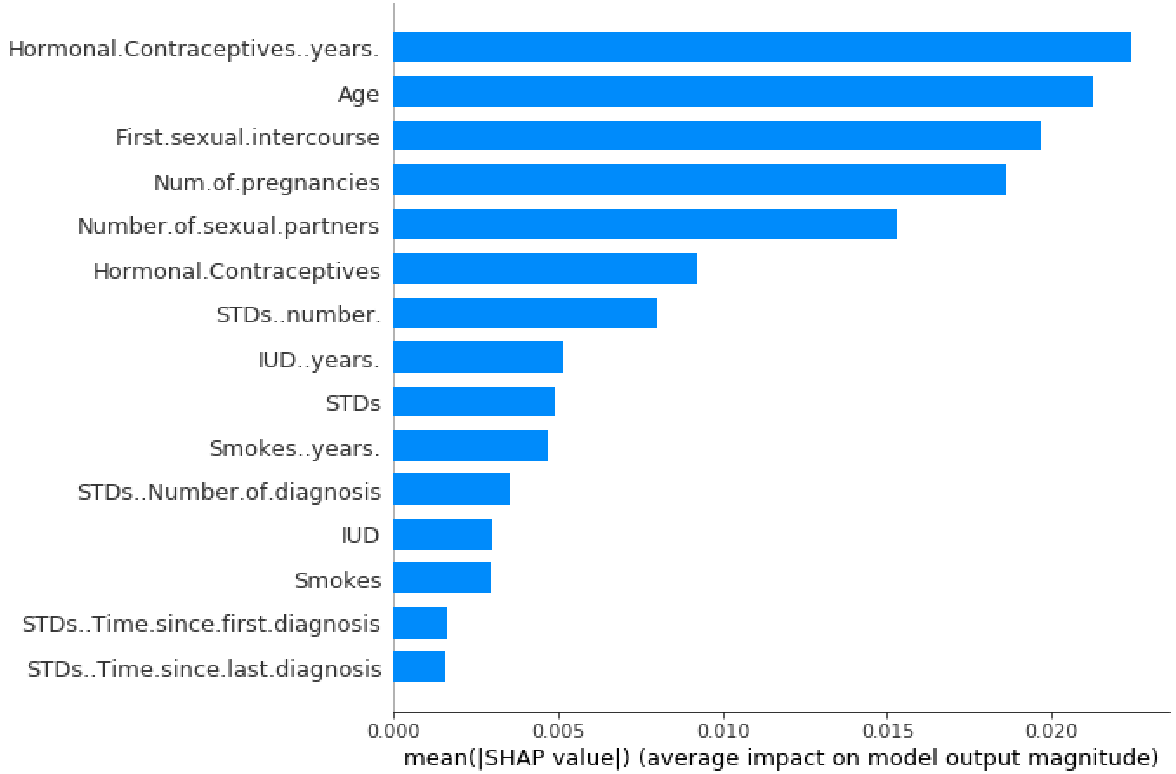
Kernel SHAP

- KenrelSHAP esimates for an instance x the contributions of each feature value to the prediction

  - Sample coalitions $z'_k \in \{0,1\}^M, \quad k \in \{1,\ldots,K\}$ (1 = feature present in coalition, 0 = feature absent).
  - Get prediction for each z'_k by first converting z'_k to the original feature space and then applying model f: $f(h_x(z'_k))$
  - Compute the weight for each z'_k with the SHAP kernel.
  - Fit weighted linear model.
  - Return Shapley values ϕ_k, the coefficients from the linear model.

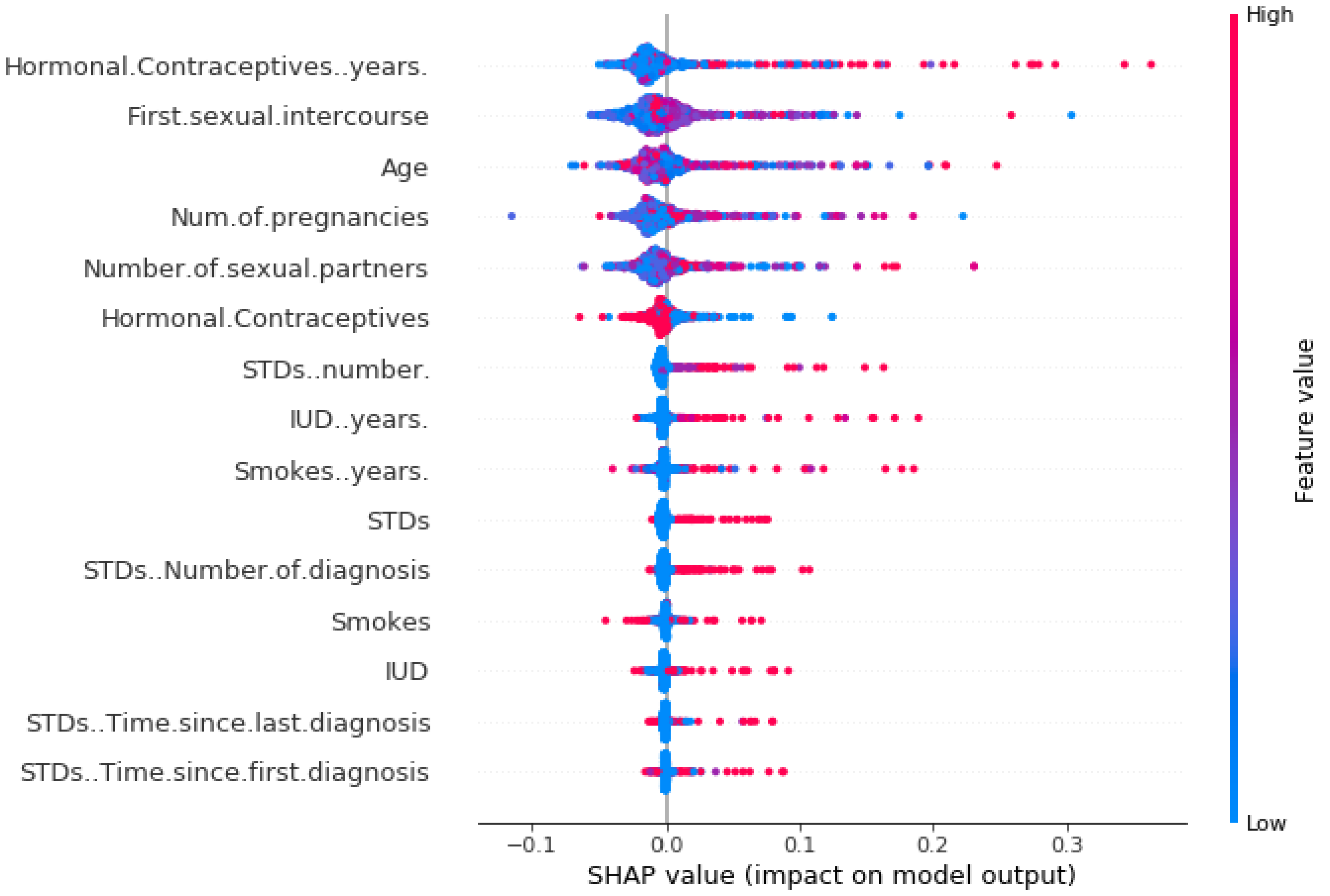
SHAP Feature Importance

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



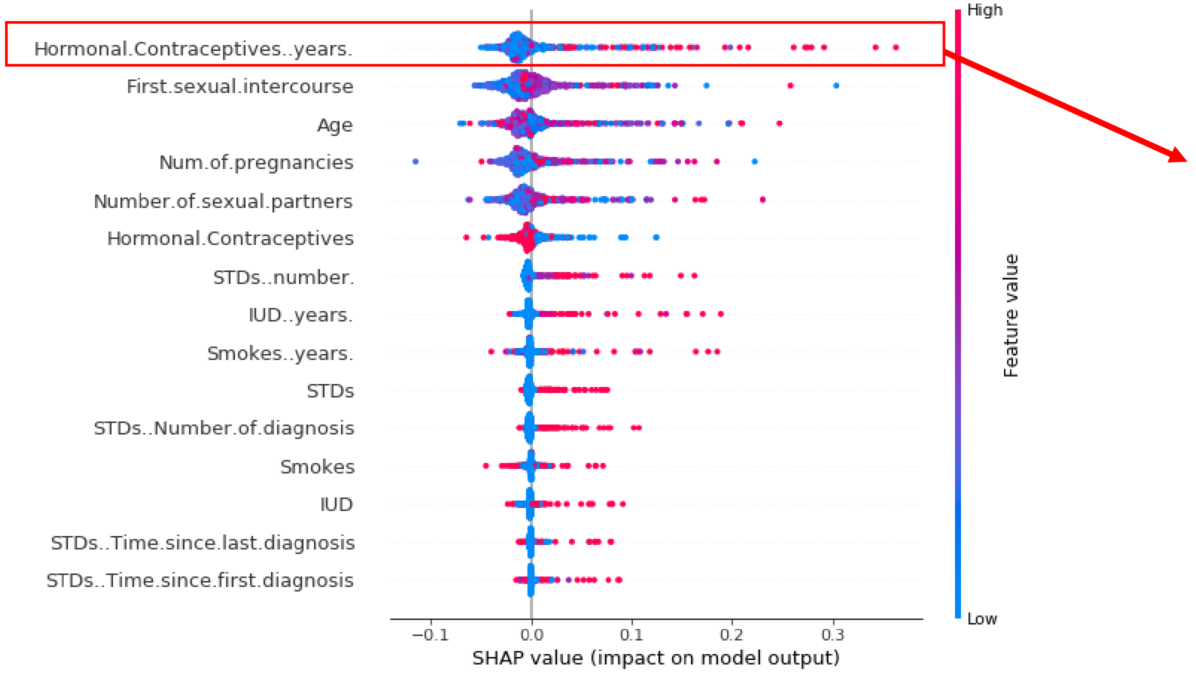
https://christophm.github.io/interpretable-ml-book/pdp.html

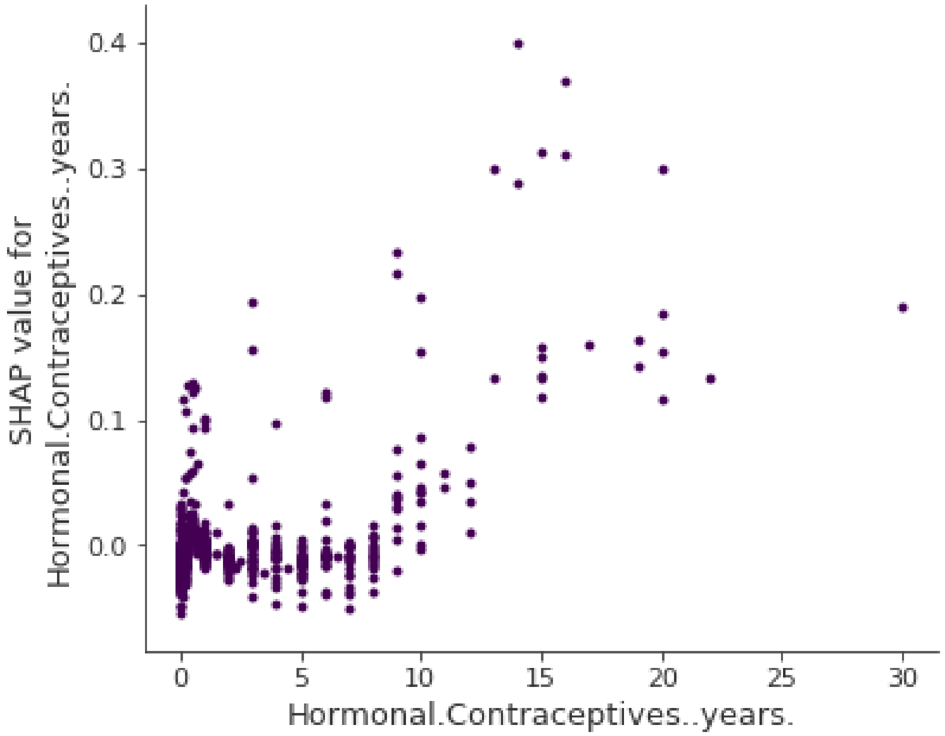
SHAP Summary Plot



https://christophm.github.io/interpretable-ml-book/pdp.html

SHAP Dependence Plot





https://christophm.github.io/interpretable-ml-book/pdp.html

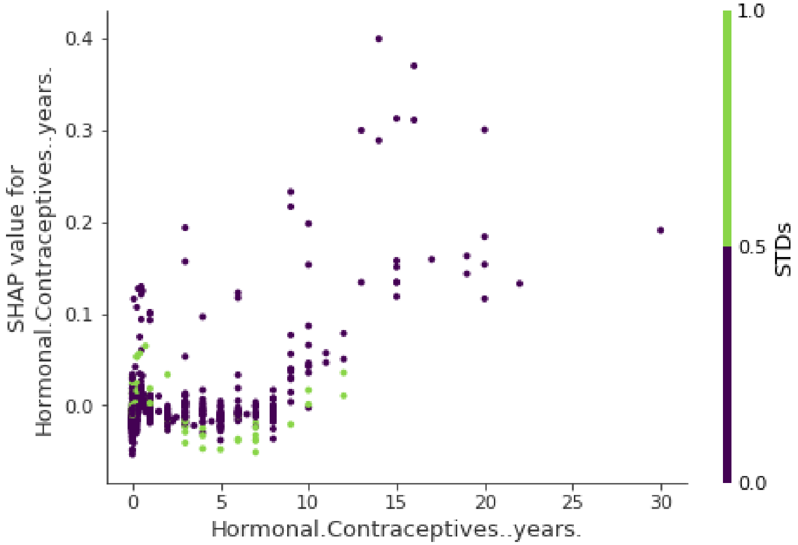
SHAP Interaction Values

- The Shapely interaction index from game theory is defined as :

$$\phi_{i,j} = \sum_{S \subseteq \backslash \{i,j\}} \frac{|S|!\,(M - |S| - 2)!}{2(M-1)!} \delta_{ij}(S)$$

when $i \neq j$

$$\delta_{ij}(S) = f_x(S \cup \{i,j\}) - f_x(S \cup \{i\}) - f_x(S \cup \{j\}) + f_x(S)$$



https://christophm.github.io/interpretable-ml-book/pdp.html

Pros and Cons for SHAP

Pros:

1. Solid theoretical foundation with fairly distributed prediction among features
2. Contrastive Explanations are allowed
3. Fast implementation for tree-based models
4. Global model interpretations

Cons:

1. KernelSHAP is slow
2. KernelSHAP ignores feature dependence
3. It is possible to create intentionally misleading interpretations.

https://christophm.github.io/interpretable-ml-book/pdp.html

Reference

[Ribeiro, 2016] Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. “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. ACM (2016).

[Lundberg, 2017] Lundberg, Scott M., and Su-In Lee. “A unified approach to interpreting model predictions.” Advances in Neural Information Processing Systems. 2017.

[Lundberg, 2018] Lundberg, Scott M., Gabriel G. Erion, and Su-In Lee. “Consistent individualized feature attribution for tree ensembles.” arXiv preprint arXiv:1802.03888 (2018).

Thank you!