SHAPR in dalex for Python

Jacek Rutkowski, Szymon Tworkowski, Jakub Walendowski

Motivation

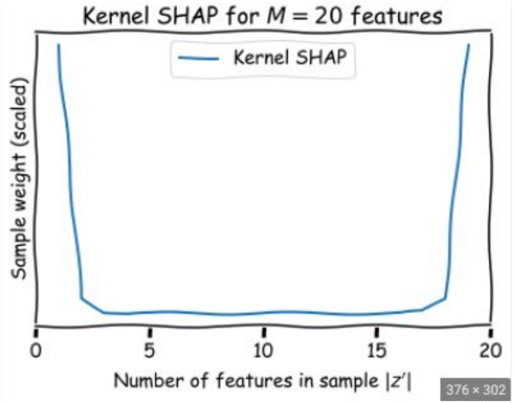
- Kernel SHAP is a very useful and efficient method, but it assumes feature independence.

- In practical cases, it is very rare that features are independent. So it can lead to poor explanations.

- SHAPR method [https://arxiv.org/pdf/1903.10464.pdf] proposes to alleviate this problem.

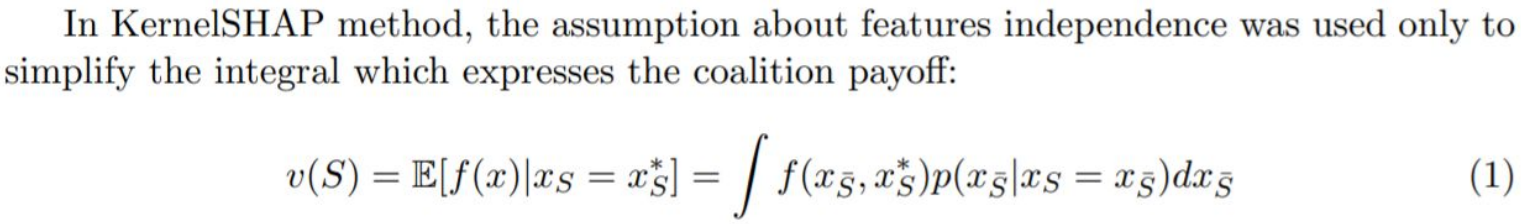
Kernel SHAP - recap

- Approximates Shapley values by linear regression in an interpretable feature space
- We need to consider all subsets of features M that gives O(2^M) computational complexity, or…
- Perform some clever sampling, which libraries like *shap* do
- For each subset (coalition), features that are not selected are sampled randomly from the dataset



SHAPR improvement for Kernel SHAP

- Instead of assuming independence of features, we try to compute conditional probabilities empirically:



In KernelSHAP method, the assumption about features independence was used only to simplify the integral which expresses the coalition payoff:

$$v(S) = \mathbb{E}[f(x)|x_S = x_S^*] = \int f(x_{\bar{S}}, x_S^*)p(x_{\bar{S}}|x_S = x_{\bar{S}})dx_{\bar{S}} \tag{1}$$

SHAPR vs. SHAP on synthetic benchmarks

To verify the SHAPR implementation, we benchmark it on 2 datasets:

- Synthetic data from XAI-Bench, for which groundtruth Shapley values can be computed
- OpenXAI benchmark consisting of real-world datasets

We compare SHAPR to baselines such as brute force Kernel SHAP (same setting as ours), official implementation from the *shap* library and random explanation

XAI-Bench Dataset

We follow https://arxiv.org/pdf/1903.10464.pdf and test multivariate Gaussian distribution with the following covariance matrix:

$$\mathbf{\Sigma}(\rho) = \begin{bmatrix} 1 & \rho & \rho \\ \rho & 1 & \rho \\ \rho & \rho & 1 \end{bmatrix}$$

We test rho = 0.7, simulating a setting where the features are dependent.

XAI-Bench Results

Both shap and brute kernel SHAP implementations achieve 0.84 correlation with groundtruth Shapley Values on the synthetic data.

SHAPR improves this correlation to **0.91**, indicating that it can handle dependent features better.

However, we found that SHAPR slightly underperforms shap on the OpenXAI benchmark (COMPAS dataset).

Limitations

- After we estimate the conditional probabilities, we need to compute *f* for every point in the training dataset, for each coalition, and take a weighted average of these probabilities.
- This is prohibitive, as usual practical datasets have 1000s of examples.

A simple heuristic: take only top-k weights and ignore the others.

After applying this heuristics, we measured that execution time is similar to standard shap. How well does it work?

OpenXAI Results

For all listed metrics, **larger score = better**

Method/Metric	RC	FA	RA	SA
Random	0.05	0.46	0.15	23.3
SHAP	0.62	0.55	0.33	0.3
SHAPR	0.52	0.51	0.27	0.3
SHAPR top5	0.53	0.52	0.25	0.31
SHAP top10	0.55	0.53	0.3	0.29

Summary

We confirmed SHAPR works well on synthetic dataset

The proposed heuristic to speed it up does not degrade performance on both the synthetic dataset and real world data

We couldn’t see any improvements on real world data, comparing to the *shap* baseline. In fact, the method is slightly worse which we attribute to the engineering.