SHAPR in dalex for Python - second checkpoint

Jakub Walendowski, Jacek Rutkowski, Szymon Tworkowski

Introduction

SHAPR is an extension to KernelSHAP, designed to account for the dependence of features. The goal of our project is to add a SHAPR extension to the DALEX library and benchmark the newly added method.

What was done?

We have created a fork of DALEX library with a proof-of-concept implementation of the SHAPR method - the repository is available here: https://github.com/jakubpw/DALEX. Existing unit tests have been modified so that we could verify that the method runs.

The xai-bench repository has been forked and modified: https://github.com/syzymon/xai-bench so that the benchmark can be run by anyone on google colab - notebook attached to the checkpoint. We ran the benchmark and showed better performance of SHAPR method compared to standard SHAP.

Experiments

We use **xai-bench** to automatically measure the performance of different explanations on synthetic benchmarks described in https://arxiv.org/pdf/2106.12543.pdf. Below we show an evaluation of 6 different explanation methods, for 2 models - decision tree and MLP, on the "gaussianPiecewiseConstant" dataset from the paper (with default hyperparameters). The results of the experiments indicate that SHAPR method achieves much higher correlation to ground-truth Shapley values comparing to standard SHAP. The detailed results are visible in tables in the Appendix.

Metrics

In order to test SHAPR we used common metrics such as ground-truth Shapley values, faithfulness, infidelity, monotonicity and ROAR (remove-and-retrain). In more detail:

- Ground-truth Shapley values method consists in comparing obtained estimations of Shapley values with the true Shapley values
- Faithfulness, infidelity and monotonicity measure the relation between how each feature changes the model output with the obtained attribution weights
- ROAR method attempts to check the validity of attribution weights by retraining the model after removing those features which obtained the biggest weights

Each of these methods is easy to apply for synthetic datasets constructed in https://arxiv.org/pdf/2106.12543.pdf, yielding a suitable way to measure the performance of explanation methods.

Appendix

Results

Decision Tree model explanations:

	shapley_corr	infidelity	roar_faithfulness	roar_monotonicity	faithfulness	monotonicity
random	0.13	0.33	0.02	0.45	0.02	0.45
shap	0.87	0.13	0.28	0.47	0.8	0.45
shapr	**0.95**	0.13	0.35	0.49	0.83	0.44
brutekernelshap	0.88	0.14	0.29	0.45	0.82	0.43
maple	0.52	0.2	-0.34	0.49	0.4	0.35
lime	0.77	0.15	0.4	0.47	0.71	0.42

MLP model explanations:

	shapley_corr	infidelity	roar_faithfulness	roar_monotonicity	faithfulness	monotonicity
random	0.11	0.19	-0.01	0.47	0.05	0.42
shap	0.8	0.03	0.49	0.52	0.83	0.44
shapr	**0.91**	0.02	0.49	0.53	0.87	0.44
brutekernelshap	0.82	0.03	0.5	0.54	0.82	0.45
maple	0.78	0.03	0.33	0.53	0.79	0.43
lime	0.79	0.04	0.5	0.51	0.8	0.47

Difficulties

SHAPR implementation from https://github.com/abacusai/xai-bench is pretty slow. It takes 30min to evaluate it on one benchmark task, assuming 1000 training samples, whereas standard SHAP takes only a few seconds and LIME takes up to 5 minutes.

What will be done next?

- Integrate dalex with xai-bench to benchmark SHAPR implemented in dalex, and also default shap from dalex
- Prepare an example usage notebook for SHAPR in dalex (Titanic dataset)
- Unit tests + documentation