Evaluating MASHAP as a faster alternative to LIME for model-agnostic machine learning interpretability

Andreas Messalas
Code4Thought PC
Patras, Greece
andreas@code4thought.eu

Christos Aridas
Code4Thought PC
Patras, Greece
chris@code4thought.eu

Yannis Kanellopoulos
Code4Thought PC
Patras, Greece
yannis@code4thought.eu

***Abstract*—Explaining machine learning models without any knowledge of their inner workings is an ambitious and often a necessary challenge to be solved. Local interpretable model-agnostic explanations (LIME) is undoubtedly one of the most well-known methods to deal with this issue, however its slow performance might render LIME unsuitable for industry level tasks. In this paper, we evaluate the Model-Agnostic SHAPley value explanations (MASHAP) method as a faster alternative for explaining black-box models and we compare it with LIME across a series of evaluation metrics. Our experiments[1] show valid reasons why one should choose MASHAP over LIME, since it delivers roughly the same consistency in a significantly faster way.**

***Index Terms*—machine learning, interpretability, transparency**

I. Introduction

Rendering Machine Learning (ML) models transparent is one of the biggest challenges Artificial Intelligence (AI) is currently facing. The intricate structure and perplexing complexity of many state-of-the-art ML models (e.g. deep neural networks) hinder vastly their level of interpretability [1], which is the ability to explain in understandable terms the decision process of the model.

A basic categorization of interpretability methods concerns the model to be explained: a *model-agnostic* method has only access to the data and the predictions of the black-box model, while the inner-working are withheld, in contrast to *model-specific* techniques, which are applicable only for a single type of algorithm, as each method is based on some specific model's internals [2].

The most well-known model-agnostic method is *LIME* [3], which stands for Local Interpretable Model-agnostic Explanations, and works by fitting an interpretable model locally around a prediction of interest. LIME is used by the majority of well-known open-source frameworks (Microsoft's *InterpretML* [4], IBM's *AI Explainability 360* [5], Oracle's *Skater* [6], *DALEX* [7]) for the part of the model-agnostic method.

Finally, the method in our previous work in [8], which we now name as *MASHAP* (i.e. Model-Agnostic SHAPley value explanations), creates a high-fidelity surrogate XGBoost model [9], which then is explained by the model-specific *Tree SHAP* method [10], [11].

In this paper, we evaluate and compare on a variety of evaluation metrics the MASHAP method with the most-well known model-agnostic counterpart, the LIME method. Our experiments have shown that MASHAP is much faster than LIME, while it achieves similar scores on the other comparison metrics, rendering it a considerable alternative for model-agnostic explanations.

[1]The code is available at https://github.com/amessalas/mashap-lime

II. LIME

LIME [3] is a model-agnostic explanation method that explains any ML model, by training a local (interpretable) surrogate model around individual predictions.

The method starts by generating a new artificial dataset of perturbed samples around the instance of interest and obtains their corresponding predictions using the black-box model as an oracle. The new data points are weighted based on their proximity to the original observation. Then, a feature selection step is implemented, which picks the features that best describe the model outcome from the new data. Finally, a simple model is trained and fitted to the new selected data and an explanation is derived through it.

The process above can be summarized in the following minimization problem, which formulates a LIME explanation $\xi(x)$ of an instance of interest x as

$$\xi(x) = \arg\min_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

where g is an explanation model, G is the class of potentially interpretable models (e.g. linear models, decision trees), $\Omega(g)$ is a measure the complexity of the explanation model (e.g. depth of tree), π_x is a proximity measure of an instance i to x, f is the probability that x belongs to a certain class and finally $\mathcal{L}$ is a measure of fidelity between f and g.

III. MASHAP

MASHAP [8] is another model-agnostic technique that can explain any model, which utilizes the Shapley values [12].

Shapley values represent the qualitative (positive or negative) and quantitative contribution of each feature to a certain prediction. They are derived from Shapley regression values [12], which have been studied extensively in cooperative game theory, ascertaining their strong and solid theoretical background. Moreover, it is proved that they are the only possible locally accurate and consistent feature contribution values [10]. While computing the Shapley values in the general case is a $\#P$ problem [13], the *Tree SHAP* method has a low-order polynomial complexity $\mathcal{O}(TLD^2)$ (where T is number of trees, L is the maximum number of leaves in any tree and D is the maximum depth of the trees) by taking advantage of the special structure of the tree-based models.

MASHAP initially builds a global surrogate model on the data we wish to explain. The goal of this step is to achieve the highest possible fidelity between the surrogate and the original model. This usually leads to over-fitted models, which however is not of great concern, since the surrogate model will only be used for extracting the explanations for the current input data and not for making any future predictions on new data. Afterwards, the surrogate model will be passed to the *Tree SHAP* method [10], [11], which will produce the explanations based on the Shapley values [12]. The *Tree SHAP* method is model-specific and accepts only tree-structured models. In our implementation we chose the *XGBoost* model [9], since it provides efficiency, over-fitting control and model re-train capability. If we have training and test sets, we can train the MASHAP explainer on the training set and then use XGBoost to partially retrain it only on the test set and get an updated version of the explainer (another option would be to merge the two sets).

IV. Evaluation metrics

The wide variety of the numerous explanation methods that have been proposed makes their taxonomy a challenging task. Based on the metrics presented in [11], [14] and [15] we created an suitable evaluation scheme to compare LIME and MASHAP on 6 metrics:

- **Runtime** [11]: the time to explain 100 predictions
- **Consistency guarantees** [11]: if a model changes so that a feature has a larger impact or stays the same, then its attribution value should not decrease. These metrics work by adding or removing features, measuring model's output or accuracy and then calculating the total Area Under the Curve (AUC). The features that increased or decreased the model's output are called *positive* or *negative* correspondingly, while features that increase model's accuracy are named *absolute*. There are three hiding modes: *mask*, where hidden values are masked with the mean value, *resample*, where they are replaced by values from a random training sample and *impute*, where they are replaced by imputed values that match the maximum likelihood estimate under the assumption that the inputs features follow a multivariate normal distribution. In total there are 18 consistency guarantees metrics, which we aggregate into 6 metrics, by grouping together the hiding modes (mask, resample and impute).
- **Identity** [14]: identical data points must have identical explanations
- **Expressive power** [15]: the presentation of the explanations (language/structure)
- **Translucency** [15]: the degree to which the explanation method uses the inner-workings of the model
- **Portability** [15]: the range of different models the explanation method can be applied to

V. Experiments and Results

The main limitation of the two methods is located on the fidelity of their produced (local or global) surrogate models. Producing a surrogate model that predicts the exact same outcome as the original model for a given dataset, does not necessarily directly imply that the two internal decision processes are the same. In other words, the reasons that led one model to make one prediction may not be the same as the reasons that led another model, even if the prediction is the same. This effect is known as the "Multiplicity of Good Models" or the "Rashomon effect" [16] and it is the main problem of every surrogate model method. In LIME this limitation is found locally, while in MASHAP it is found globally. In this section, we will examine inter alia in which case the "Rashomon effect" has the biggest impact.

For our experiments, we used 21 datasets from the OpenML repository [17], which are listed in Table I and whose dimensionality (i.e. number of features) ranges from 16 to 144 features. We also used five machine learning models from the scikit-learn library [18] that implement the following algorithms: *K-Nearest Neighbors*, *Decision Trees*, *Random Forests*, *Gradient Boosting* and *Multi-layer Perceptron (MLP)*. Our goal was to get a variety of predictions, from different models and from datasets with different dimensions.

The results were:

- **Runtime**: MASHAP was approximately 495 times faster than LIME. The runtime for each method was calculated by averaging the runtime of the five models for a given dataset. LIME is significantly slower because it requires a generator of perturbed data points around the instance of interest, feature selection and training of a linear model for every instance of the test set.
- **Consistency guarantees** [11]: each one of the six aggregated metrics was calculated on 315 measurements (1890 in total). We used *Student's t-test*, for paired samples, with a confidence level threshold α of 0.05 to compare the results of the six metrics, which are presented in Table II. MASHAP achieves significantly better performance than LIME at the Keep and Remove absolute metrics, while LIME outperforms MASHAP at the Keep negative metric. As for the other metrics there is no clear winner, since there is not statistical significance in the mean differences of the two methods. Based on these results, we can conclude that LIME's and MASHAP's performance in consistency guarantees is similar.

TABLE I
21 OPENML DATASETS USED IN THE EXPERIMENTS

Dataset (OpenML id)	# records	# features
adult	48842	14
bank-marketing	45211	16
climate-model-simulation-crashes	540	20
compas-two-years	5278	13
credit-approval	666	15
credit-g	1000	20
cylinder-bands	378	37
default-of-credit-card-clients	30000	23
elevators	16599	18
ionosphere	351	34
jasmine	2984	144
kc1	2109	21
kc3	458	39
kr-vs-kp	3196	36
mushroom	8124	22
nomao	34465	118
pc1	1109	21
ringnorm	7400	20
SPECTF	349	44
twonorm	7400	20
vote	435	16

TABLE II
CONSISTENCY METRICS RESULTS

Metric	Winner	p-value	Statistical Significance (α=0.05)
Keep positive	MASHAP	0.535367	No
Keep negative	LIME	0.000031	Yes
Keep absolute	MASHAP	0.000807	Yes
Remove positive	MASHAP	0.089962	No
Remove negative	LIME	0.247725	No
Remove absolute	MASHAP	0.004407	Yes

- **Translucency**: both MASHAP and LIME are model-agnostic methods, so they are oblivious of the inner-workings of the model they try to explain. However, LIME requires access to the predictive function of the black-box to use it as an oracle for the generated perturbed data points. MASHAP, on the other hand, only requires the data to be explained and its corresponding predictions from the model.
- **Portability**: both MASHAP and LIME can be applied to any kind of machine learning model that predicts an output based on some input.
- **Identity**: both MASHAP and LIME meet the requirement for this metric. MASHAP uses deterministic procedures, whereas LIME has a data sampling step, which is probabilistic, however it can be manipulated to provide the same output by passing a random state parameter.
- **Expressive power**: both MASHAP and LIME have the same expressive power, since both return feature summaries, which can lead to various presentations of the explanation (plots, text, etc.).

VI. CONCLUSIONS

We presented two model-agnostic methods that output local explanations through feature scores and that use surrogate models, which generally suffer from the "Rashomon effect" [16]. The consistency guarantees in our experiments showed that this effect statistically has similar impact on a local (i.e. case of LIME) and a global (i.e. case of MASHAP) level. The same applies also for the identity, expressive power and portability metrics. However, the runtime metric showed that MASHAP is much faster than LIME for batches of data. Moreover, MASHAP is a "pure" model-agnostic method (translucency metric), since it does not require any access to the original model, unlike LIME which uses it as an oracle. Considering all the above arguments, it is fair to say that MASHAP is a considerable alternative to LIME for model-agnostic interpretability, since it delivers roughly the same consistency as LIME, in a much faster way.

REFERENCES

[1] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017.

[2] C. Molnar, *Interpretable Machine Learning*, 2019, https://christophm.github.io/interpretable-ml-book/.

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, "Model-agnostic interpretability of machine learning," *ArXiv*, vol. abs/1606.05386, 2016.

[4] H. Nori, S. Jenkins, P. Koch, and R. Caruana, "Interpretml: A unified framework for machine learning interpretability," *arXiv preprint arXiv:1909.09223*, 2019.

[5] V. Arya, R. K. E. Bellamy, P.-Y. Chen, A. Dhurandhar, M. Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilović, S. Mourad, P. Pedemonte, R. Raghavendra, J. Richards, P. Sattigeri, K. Shanmugam, M. Singh, K. R. Varshney, D. Wei, and Y. Zhang, "One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques," 2019. [Online]. Available: https://arxiv.org/abs/1909.03012

[6] Oracle, "Skater," 2017. [Online]. Available: https://github.com/oracle/Skater

[7] P. Biecek, "Dalex: Explainers for complex predictive models in r," *Journal of Machine Learning Research*, vol. 19, no. 84, pp. 1–5, 2018.

[8] A. Messalas, Y. Kanellopoulos, and C. Makris, "Model-agnostic interpretability with shapley values," *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pp. 1–7, 2019.

[9] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, 2016, pp. 785–794.

[10] S. M. erg, G. G. Erion, and S. Lee, "Consistent individualized feature attribution for tree ensembles," *CoRR*, vol. abs/1802.03888, 2018.

[11] S. M. Lundberg, G. G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S. Lee, "Explainable AI for trees: From local explanations to global understanding," *CoRR*, vol. abs/1905.04610, 2019. [Online]. Available: http://arxiv.org/abs/1905.04610

[12] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games II*, H. W. Kuhn and A. W. Tucker, Eds. Princeton: Princeton University Press, 1953, pp. 307–317.

[13] X. Deng and C. H. Papadimitriou, "On the complexity of cooperative solution concepts," *Mathematics of Operations Research*, vol. 19, no. 2, pp. 257–266, 1994.

[14] M. Honegger, "Shedding light on black box machine learning algorithms: Development of an axiomatic framework to assess the quality of methods that explain individual predictions," *CoRR*, vol. abs/1808.05054, 2018.

[15] M. Robnik-Sikonja and M. Bohanec, *Perturbation-Based Explanations of Prediction Models*, 06 2018, pp. 159–175.

[16] L. Breiman, "Statistical modeling: The two cultures," *Statistical Science*, 2001.

[17] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "Openml: Networked science in machine learning," *SIGKDD Explorations*, vol. 15, no. 2, pp. 49–60, 2013.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.