Version of Record: https://www.sciencedirect.com/science/article/pii/S0306437922001405
Manuscript_353d884f0a149c0a93e15f57a0d6b82a

A quantitative approach for the comparison of additive local explanation methods

Emmanuel Doumard[a,c,e], Julien Aligon[b,c], Elodie Escriva[b,c,1], Jean-Baptiste Excoffier[d], Paul Monsarrat[a,e,f,g], Chantal Soulé-Dupuy[b,c]

[a] *Université de Toulouse - Paul Sabatier*
[b] *Université de Toulouse - Capitole*
[c] *Institut de Recherche en Informatique de Toulouse IRIT (CNRS/UMR 5505)*
[d] *Kaduceo company*
[e] *RESTORE Research Center*
[f] *Artificial and Natural Intelligence Toulouse Institute ANITI*
[g] *Oral Medicine Department*

Abstract

Local additive explanation methods are increasingly used to understand the predictions of complex Machine Learning (ML) models. The most used additive methods, *SHAP* and *LIME*, suffer from limitations that are rarely measured in the literature. This paper aims to measure these limitations on a wide range (304) of OpenML datasets using six quantitative metrics, and also evaluate emergent coalitional-based methods to tackle the weaknesses of other methods. We illustrate and validate results on a specific medical dataset, *SA-Heart*. Our findings reveal that *LIME* and *SHAP*'s approximations are particularly efficient in high dimension and generate intelligible global explanations, but they suffer from a lack of precision regarding local explanations and possibly unwanted behavior when changing the method's parameters. Coalitional-based methods are computationally expensive in high dimension, but offer higher quality local explanations. Finally, we present a roadmap summarizing our work by pointing out the most appropriate method depending on dataset dimensionality and user's objectives.

Email addresses: emmanuel.doumard@irit.fr (Emmanuel Doumard), julien.aligon@irit.fr (Julien Aligon), elodie.escriva@kaduceo.com (Elodie Escriva), jeanbaptiste.excoffier@kaduceo.com (Jean-Baptiste Excoffier), paul.monsarrat@univ-tlse3.fr (Paul Monsarrat), chantal.soule-dupuy@irit.fr (Chantal Soulé-Dupuy)

1. Introduction

Machine Learning (ML) represents a real revolution in various domains, such as finance, insurance, healthcare, biomedical. However, machine learning models give a prediction without necessarily being accompanied by an understandable explanation. These models, often referred as "black-boxes", raise the challenging question of how humans can understand the determinants of the prediction. Explainability is also more than a technological problem, it raises ethical, societal and legal issues. In healthcare, this may involve the professional being able to explain to the patient how the algorithm works and the criteria for the decision process. The results of ML models must therefore be expressed in a way that can be understood by domain-experts, like medical practitioners [1, 2]. Since *SHAP* [3], machine learning experts show a very clear interest for the additive methods as a huge number of works using these methods are published each year. The local additive methods include *LIME* [4], *SHAP* [3] and more recently the coalitional-based methods [5] which use subsets of features (coalitions) to approximate explanations. They are called "local" because they produce an explanation for each data instance (*e.g.* each patient) and "additive" because the sum of the influence of each feature for a given instance approximates the prediction of the model. The user-friendly representation of explanations, based on the influences of features, allows domain and non-domain experts to better understand models' predictions [6]. Existing explanation methods are model-specific or model-agnostic depending on whether they can be applied to some or all types of machine learning models, with local or global explanations to understand either an individual prediction or the behaviour of the model as a whole. While these methods have been evaluated in a number of contexts, no *in-depth* evaluation is available for a rational choice of one technique over another.

The objective of our previous work [7] was to study the advantages and disadvantages of using each additive method, considering the effects of both the ML models used and the type of dataset on the influences of features (both at the instance and feature level). In this paper, we complete our work with six different metrics, computed for each model and dataset, providing insight on computational time, feature importance, robustness, readability

and clusterability of explanations.

The paper is organised as follows. Section 2 describes the four additive methods to be compared along with existing work on classification and comparison of local explanation methods with associated metrics. Section 3 presents the methodology of our experiments, including the datasets and ML models used, and the six metrics retained to compare local explanation methods. Section 4 presents the experiments where we study the explanation characteristics and the impact of the predictive model on explanation profiles. Section 5 highlights the behavior of explanation methods based on a practical medical use case (*SA-Heart*) with a focus on the impact of hyperparameters of studied methods on the generated influences. Conclusive lessons-learned are then detailed in Section 6, while future axis of work are identified in Section 7.

2. Related Works

We first present the additive methods we use through this paper (Section 2.1) as well as an overview of works that aim to compare and evaluate them (Section 2.2). Finally, in Section 2.3, we discuss the current limits of the metrics used to compare explanations.

2.1. Additive methods

Additive methods are described as explanation models that produce for a single instance a vector of weights to represent the contribution of each feature to the prediction. The sum of these contributions (also called *influences*) approximates the prediction of the original model. In this section, we explore several existing methods that fit this definition. We focus on post-hoc methods that deliver their explanations for a given model already trained. Methods used in this study are all agnostic, meaning that they can be applied to any kind of machine learning model, except for the *TreeSHAP* method [8] that is designed specifically for tree-based models.

To illustrate how local additive methods work, we give an example of these explanations for the well-known Iris dataset (4 attributes and 150 instances). As the Iris dataset is an easy problem, it can be modeled by a small decision tree, described in Figure 1 and explanations can be compared to this classification.

In this example, Figure 2 shows the influence of each attribute for the class "Versicolor" for a Versicolor iris. The petal width is the attribute with

the greatest influence on the prediction "Versicolor", followed by the petal length. When analyzing the decision tree, the same attributes seem the most important to distinguish each iris class. On the other hand, sepal width and length do not have a huge impact on the prediction as their influences are close to 0. As these attributes are not relevant for differentiating classes based on the decision tree, it seems consistent that they have little influence on the prediction. Based on the Figure 2, users can understand that, for this instance, petal measures are the most important to determine the class of the iris while sepal measures are almost unnecessary with low influences on the prediction.

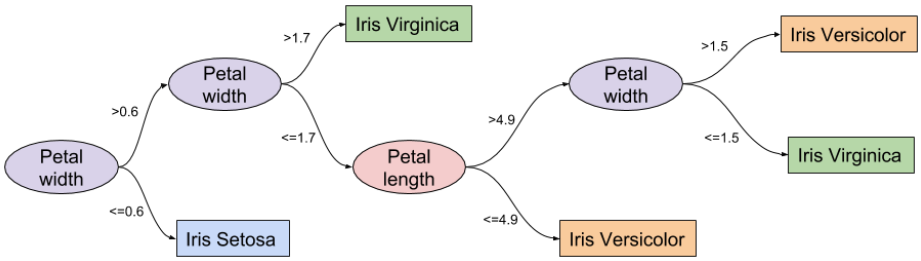


Figure 1: Decision tree trained for the iris classification [9].

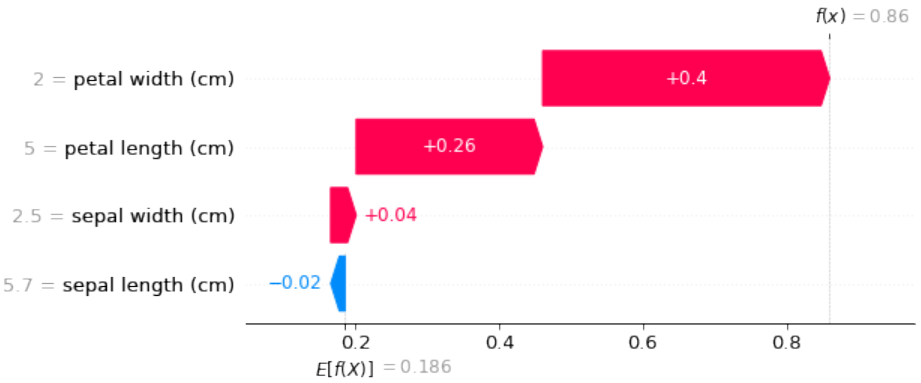


Figure 2: Example of local additive explanations for one instance from the iris dataset for the class "Versicolor".

2.1.1. LIME

LIME is a well-known local explanation method described in [4]. *LIME* uses explainable models to locally approximate a complex black-box model

and, for each instance, explain the influence of each feature on the prediction. For each instance to be explained, *LIME* generates new data in a close neighborhood and computes the predictions of these new instances with the black-box model. A linear regression model, an interpretable model, is trained with the new dataset. This local model is then used to explain the prediction of the instance of interest in the form of a weight vector associating each feature with its influence on the prediction. A well-known limitation of *LIME* is the restrictive hypotheses on which *LIME* is based, such as local linearity and feature independence [10, 11]. Defining the locality around an instance of interest can also be a challenge, as the fitness of the surrogate model has a significant impact on the accuracy of the explanations [12] as well as their stability [13].

The full implementation of *LIME* is available on GitHub[1].

2.1.2. Shapley Values (complete method)

To explain individual predictions, a method based on Shapley values is described in [14, 15, 16]. Shapley values 'fairly' weight groups of features according to their relative importance to a defined gain [17]. In machine learning, the gain can be linked to the prediction made by the model. Influences of each feature are computed based on its impact on the prediction for each coalition of features. The explanation method based on Shapley values is called the *complete method*. All coalitions are evaluated with and without each feature and the change on the prediction is used to compute the influence of the feature. The *complete method* can be used as a baseline to compare other methods as it is an exhaustive method close to the original intuition behind feature influence [5]. This method is however very expensive to compute, with an exponential complexity in relation to the number of features in the dataset.

Several more recent methods, including *SHAP* [3] and coalitional methods [5], are based on Shapley values with the aim to solve limitations of the *complete method*.

2.1.3. SHAP

SHAP (SHapley Additive exPlanations) [3] method worked on improving computation time and explanation precision, especially for tree-based models

[1] `https://github.com/marcotcr/lime`

[8]. It combines *LIME* [4] and Shapley values [16], along with other methods from the literature [18, 19, 20, 21], in a unique framework to produce local explanations. The main idea is to create perturbations to simulate the absence of a feature and to use a linear local model to approximate the change in the prediction, as in *LIME*. This avoids retraining the complex model without the feature of interest. Local explanations can be aggregated to explain the global behaviour of the model. Global and local explanations are then consistent with each other as they have the same foundation. *SHAP* includes an agnostic explainer, *KernelSHAP*, as well as model-specific explainers, such as *TreeSHAP*, *LinearSHAP* or *DeepSHAP* for tree-based models, linear models and deep models respectively. While commonly used in Machine Learning context [22], *SHAP* still suffers from lack of precision [10, 23] mostly due to their restrictive hypothesis (local linearity and feature independance) as with *LIME*. Moreover, computation time is still high for other models than tree-based models [24].

The full implementation of *SHAP* is available on GitHub[2].

2.1.4. Coalitional-based method

Another agnostic explainer based on Shapley values, the *coalitional method*, was introduced to take into account the interdependence of features and solve some restrictions of SHAP. It uses grouping methods such as *Principal Component Analysis* (PCA), *Spearman correlation factor* (Spearman) and *Variance Inflation Factor* (VIF) to select amongst all the possible groups of features those that would be the most interesting for explanations [5]. These groups are then used as coalitions to compute Shapley values as in the *complete method*. The influence of each feature is defined as its impact on the prediction only on these groups of features, approximating the *complete method* and reducing the computational time. Grouping methods are defined with a parameter that changes the number and size of feature groups in order to prioritise a lower computational time or a higher accuracy. As for *SHAP*, local explanations can be aggregated into global explanations to study global and local behavior of the model.

The full implementation of *Coalitional-based method* is available on GitHub[3].

[2] `https://github.com/slundberg/shap`

[3] `https://github.com/kaduceo/coalitional_explanation_methods`

2.2. Describing and comparing the additive methods

Few works [25, 26] exist in the literature to classify and categorize machine learning explanation methods. In [6], a complete description of explanation approaches from literature is given. The author explain their advantages and disadvantages, giving an overview of their limits in general. For example, even if the *LIME* and *SHAP* approaches are model-agnostic and human-friendly, they suffer from a lack of consideration of feature correlation and possible instability of the explanations. Another paper tackling the limits of the additive methods (*LIME* and *SHAP*) is presented in [10]. Biased classifiers can fool explanation methods, whose problem is even more accentuated on *LIME*.

Comparative studies between local explanation methods are also available, such as [27, 28, 5, 29]. Most works focus only on SHAP and LIME when comparing local methods. In [29], the authors compare *LIME*, *SHAP* and Scoped rules using the ranking of features by importance. However, they use a single metric with a single prediction model, on a single dataset, which limits the generalisability of results. In [5], a new additive method was proposed based on Shapley values and taking into account feature correlation, described in section 2.1. This method was compared with *LIME* and *SHAP* considering computation time and accuracy score, and show that their proposal is competitive with the literature. In [28], *LIME* and *SHAP* are used in a context of feature selection and compared to a Mean Decrease Accuracy (MDA) approach. They show that these explanation methods can also be used as feature selection methods with better results than the literature. In [27], the authors compare six local model-agnostic explanation techniques using custom quantitative measures evaluated on two tabular and two text datasets. From these experiments, no single method stands out for all metrics and all datasets. According to the metrics considered, each method has its own strengths and weaknesses, the choice being dependant on both the user's goal and dataset content.

However, none of these previous works clearly indicates in which situation a method should be preferred to another one, nor what are the strengths and weaknesses of each method. Most previous works focus on a specific case (dataset) and metrics, which makes generalization difficult. Consequently, our aim is to give the key factors to make an informed decision among the existing additive methods. To improve generalization, we evaluate metrics on several different prediction models with a large number of widely different datasets.

2.3. Metrics for evaluating explanation methods

In the machine learning explanation field, one challenge is to define what is a good explanation and how to show mathematically their relevance. [30] indicates that evaluating explanations methods is very subjective and no consensus yet exists to propose relevant metrics. Nevertheless, the authors summarize criteria for a good human-friendly explanation such as contrastiveness, social adaptation, focus on the abnormal, truthfulness or consistency with prior beliefs. They present their work as general guidelines to objectively define good explanations. [31] also defined properties for individual explanations to help characterize good explanations, like accuracy, fidelity, representativity, understandability or consistency. Explanations that comply with these properties can be good explanations as they can be seen as true to the model or the data, trustworthy and easy to understand for the end-users. However, all these characteristics are mainly subjective as they are not defined mathematically and it is not obvious how to measure them. The definition of a good explanation can also differ based on the end-user, the application domain, the objectives for the use of explanations, making it difficult to objectively assess the quality of the explanations [30].

Other papers define more precise metrics to evaluate explanations and apply them on some use-cases. In [27], the authors define similarity, bias detection, execution time, and trust as quantitative measures to evaluate explanations. These metrics are generic as the authors aim to compare a wide range of explainability techniques. They are evaluated on several tabular and text datasets. However, only an intuitive description of the metrics is given and no mathematical implementation is provided, making it difficult to re-use them.

Through this paper, we focus on local additive explanation methods, which produce, for each instance to be explained, one vector of attribute influences. This allows for quantitative metrics to evaluate and compare explanations based on their influence vectors, and are sometimes referred to as attribution-based metrics. In particular, in [32] the authors define monotonicity and effective complexity to evaluate explanation qualities. Monotonicity is particularly interesting as it evaluates the relationship between the values of an explanation and its expectations. Effective complexity is related to conciseness, evaluating the minimum number of features necessary to the explanation. Robustness is another frequently mentioned metric over the literature, defined as the capacity of the explanation to be similar when inputs are similar. Several mathematical formulations of this metric exist,

based on how authors determine what similarity means and how to compute it [33].

These multiple implementations produce metrics that are efficient to evaluate each explanation method specifically. However, it can be confusing to compare measures that have not been calculated in the same way or to find inconsistent definition of the same metrics over the literature. One challenge then lies in the ability to find metrics that are applicable to all the methods to be compared. The lack of unified metrics and the differences between explanations methods mean that only similar methods can be easily compared thus increasing the complexity of selecting metrics and evaluating and comparing explanations [33, 34, 35].

Another approach is to compare the evaluated methods to a baseline and measure the error between the two. This technique compares multiple explanation methods at once, based on the same metric as long as the methods produce similar outputs [5, 36]. The error is computed as the distance between the baseline and each method, allowing the use of any existing distance that can be applied on all the methods to be evaluated. Although this approach solves the problem of finding a metric relevant to all explanations, it raises the problem of defining a trustable and consensual baseline.

In this paper, we propose metrics based on desirable properties mentioned earlier. As we are comparing methods that all produce the same kind of output - a vector of influences for each instance of a data set - we also provide the exact formula used to compute each metric. Therefore, a major strength of our metrics is that they are systematically applicable to any method that produce a vector of influence for one instance, and any machine learning model as long as the method is model-agnostic. The main objective of this paper is not to qualify each explanation method in absolute, but to compare their relative performances.

3. Methodology

In this section, we describe our methodology for comparing the explanation methods presented in the previous section. The goal is to identify the general behavior of each method and how this behavior eventually differs across the predictive models (learned from data) and the dimensionality of the data (number of features). Section 3.1 presents the experimental protocol we use, and we define in section 3.2 the six metrics we use to compare the explanation methods.

3.1. Experimental protocol

All experiments are run on an Intel Xeon Gold 6230 processor with 125 GB of RAM using Python 3.9.7. All runs are performed on a single core of CPU for optimization and reproducibility. To compare explanation methods, we apply them to a wide range of 304 datasets available on OpenML[4]. Due to computational constraints of explanation methods, we only considered datasets with at most 13 features, and at most 10 000 instances. We also only considered classification tasks to use comparable predictive models and metrics. We describe the amount and size of datasets per number of features in Table 1.

As an explanation method needs a model to be applied to, we choose four widely used types of ML models for classification: Logistic Regression (LR), Support Vector Machines (SVM), Random Forests (RF) and Gradient Boosted Machines (GBM). For the first three, we use the implementation of Python library *scikit-learn* version 1.0.1. For GBM, we use the Python library *XGBoost* version 1.5. We use default values for models hyperparameters. For explanations methods, we use Python libraries *shap* 0.40 and *lime* 0.2.0.1.

In Section 3.2, we present six metrics that will be used for this study to compare the explanations.

3.2. Metrics of interest

Through this paper, to evaluate explanation methods performances and compare them over a high number of datasets, we use six different metrics that only consider the influence values given by the method. In all the following definitions, let X be a given dataset with n instances and d the number of features, and f an explanation method that can be applied to each instance of the dataset given a machine learning model (that we omit for conciseness).

The first metric is the mean computation time per instance, which is the amount of time taken by a given method to compute the local influences of a whole dataset, divided by the number of instances in the dataset.

[4] `www.openml.org`

Number of features	Number of datasets	Number of instances		
		Min	Max	Mean
1	5	130	9100	3079
2	21	52	5456	901
3	43	60	9989	1729
4	23	96	8641	1016
5	35	62	7129	941
6	27	51	9517	949
7	33	54	4052	499
8	32	52	8192	1473
9	23	52	1473	484
10	37	57	5473	712
11	8	66	4898	942
12	12	123	8192	1175
13	5	178	506	293
Total	304	51	9989	1035

Table 1: Datasets description

The second one is a quantification of the average deviation of the influence given by a method from the *Complete* method (see Section 2.1). It is very similar to the error metric used in [9].

Definition 1. *Error with regards to the Complete method*

Let $f_k(x)$ be the influence of a feature k produced by an explanation method f for a given instance x, and a given machine learning model, and $f_k^C(x)$ the influence given by the Complete *method for the same model, same feature and same instance. We define the mean error of the explanation method as:*

$$err(f, X) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{p} \sum_{k=1}^{d} \left| f_k(X_i) - f_k^C(X_i) \right| \tag{1}$$

The third metric is inspired from the principle of effective complexity defined in [32]. However, it benefits from the absence of any parameter. It evaluates the conciseness of an explanation given the distribution of feature importance. Feature importance (mean absolute value of influence assigned to instances for a given feature) is ranked in decreasing order, then cumulative sum is calculated. For example, in a dataset with 2 features, if a method gives 80% of the importance to the most important feature (and so 20% to the second), it would have a cumulative importance proportion vector of $[0, 0.8, 1]$. We can then define the normalised Area Under Curve (AUC) as:

Definition 2. *Area under the cumulative feature importance curve*

Let $C \in [0; 1]^{d+1}$ be the cumulative importance proportion vector given by an explanation method over a dataset, with C_i the total importance proportion taken by the i most important features. We define the area under the cumulative feature importance curve as:

$$AUC(X) = \frac{1}{d} \sum_{i=0}^{d-1} \frac{C_i + C_{i+1}}{2} \tag{2}$$

This metric shows whether an explanation method favours the attribution of great importance to a few features or, on the contrary, a more homogeneous distribution among a larger number of features. As this cumulative sum is sorted by decreasing value, this value is bound between 0.5 and 1. A value of 0.5 means that the explanation method gives the same importance to all

features, while a value of 1 means that the explanation method gives non-zero influences only to a single feature, explaining the model's predictions with a single feature.

The fourth metric is a measure of robustness of the method. A method is robust if similar instances lead to similar explanations. Formalized in [33], we use the discrete version of the local Lipschitz estimation.

Definition 3. *Robustness (local Lipschitz estimation)*
Let $\mathcal{N}_\epsilon = \{x_j \in X | \|x_i - x_j\| \leq \epsilon\}$ *the* ϵ*-neighborhood of the instance* x_i*.*

$$\tilde{L}_X(x_i) = \max_{x_j \in \mathcal{N}_\epsilon(x_i) \leq \epsilon} \frac{\|f(x_i) - f(x_j)\|_2}{\|x_i - x_j\|} \tag{3}$$

A high value of $\tilde{L}_X(x_i)$ means that the explanation method is not robust for the instance x_i over the dataset X, and a low value means that the explanation is robust for the instance x_i over the dataset X. We average this value over all instances of a dataset to get the value of the metric for a method for a dataset.

The fifth metric is a measure of readability of the global explanation. It is inspired from the monotonicity metric defined in [32], but rather than looking at the correlation between the absolute values of the attributions and the expectations (that we cannot compute), we look at the correlation between the data values and the influences for an attribute. Even if the explanations are calculated for each instance, we want these explanations to make sense when comparing one to another. To evaluate that, we look at the relationship between the value of a feature, and the value of the explanation for this feature, for all instances using the Spearman correlation r.

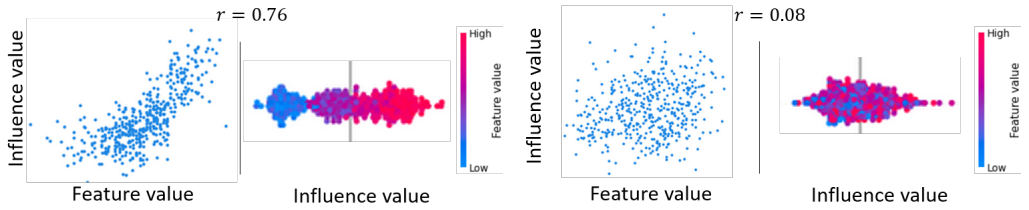
$$r = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

Definition 4. *Readability*
Let $X_i \in \mathbb{R}^n$ *be the dataset feature* i*,* $f(X_i) \in \mathbb{R}^n$ *the explanation of each instance for such feature* i *and* $r(X, Y)$ *the Spearman correlation coefficient of two vectors of equal size. We define the readability of an explanation method over a dataset* X *as:*

$$d_i = X_i - f(X_i)$$

$$\mathcal{R}(X) = \frac{1}{d} \sum_{i=1}^{d} |r(X_i, f(X_i))| \tag{4}$$

In Figure 3, we show a visual example of what we consider readable or unreadable according to our definition.



(a) (b)

Figure 3: (a) Example of a readable explanation. Each dot corresponds to an instance. On the right (compact representation), the color represents the value of the feature. (b) Example of an unreadable explanation

The sixth and last metric measures the pairwise feature interaction as captured by the explanations. In order to do that, for each pair of features within a global explanation, we use a clustering method to create a partition of the explanation of all instances for the pair of features, and then evaluate the quality of the clustering created this way. We average this value over all pairs of features and name this metric (2-dimensional)-clusterability:

Definition 5. *Clusterability*

Let $f(X_i) \in \mathbb{R}^n$ the explanation of each instance for a feature i, K be a clustering function, and S an evaluation function for a clustering. We define clusterability as:

$$Cl(X) = \frac{2}{d * (d-1)} \sum_{\substack{i,j \in [1,\dots,d] \\ i \neq j}} S(K(f(X_i), f(X_j))) \tag{5}$$

A high clusterability score means that the explanation method draws relationships between pairs of features for their joined contribution to the predictions. For our experiments, we use K-Means as the clustering method and the Silhouette score for the clustering quality measure.

Through this paper, we will refer to these six metrics as, in order, *Computation time*, *Error*, *AUC*, *Robustness*, *Readability* and *Clusterability*

4. Results

In this section, following the methodology previously described, we present the results in two ways. Section 4.1 aims to compare the four additive meth-

ods introduced in Section 2.1. In particular, we use two distinct coalitional-based methods: the *Complete* method, which serves as reference for an influence deviation measurement (Definition 1), and the *Spearman* method with a threshold of 25% of all groups of features. Regarding *SHAP*, we use the model-agnostic *KernelSHAP* on all datasets. As this method is very slow to execute if we use the whole dataset as background samples for permutations, we choose to follow *SHAP*'s recommendation[5] by doing a *K-Means* clustering on the input dataset, and then taking the centroids as background samples. We choose $K = 10$ clusters for each dataset. In addition, for the two tree-based predictive models XGBoost and Random Forests, we use the model-specific explainer *TreeSHAP* by two implementations. The first one determines *SHAP* values with background samples, similarly to *KernelSHAP* but optimised for tree-based methods. We use the whole dataset as background samples for this method. The second one approximates *SHAP* values by considering the trees structures, and does not need background samples in input, so we name it *TreeSHAPapprox*. Last, we consider *LIME*, which requires a number of perturbed samples to be created for explaining each instance. We choose to set this number to 100 samples for all datasets. With similar methodology, Section 4.2 identifies the impact of the predictive model on specific explanation methods.

Supplementary data referenced through the rest of the paper are available on Github[6].

4.1. Additive method comparison

Computation time

We show in Figure 4 the evolution of the computation time of each method for each predictive model, averaged over datasets that share the same number of features.

LIME, having a linear complexity with the number of features, is computationally expensive compared to other methods in low dimension (few features), but is less expensive than coalitional-based methods and *KernelSHAP* in higher dimensions. *LIME* also seems to have very low inter-dataset time variability, resulting in smaller error bars on the graph.

[5] *KernelSHAP* documentation includes recommendation to use K-Means algorithm to speed up computation time `https://shap-lrjball.readthedocs.io/en/latest/generated/shap.KernelExplainer.html`

[6] `https://github.com/EmmanuelDoumard/local_explanation_comparative_study`

Coalitional-based methods show an exponential complexity with the number of features, having high execution time in high dimension, but have a similar execution time with other methods in low dimension.

Spearman method execution time seems naturally correlated to the *Complete* method execution time, taking a fraction of the time (roughly 25%) of the *Complete* method.

KernelSHAP, despite a limitation on the amount of background samples, has a high execution time in high dimension, comparable to coalitional-based methods for non-tree based methods.

For tree-based methods, *KernelSHAP* is slower in low dimension, but faster in high dimension than coalitional-based methods.

Last, tree-based explainers seem to have constant execution time per instance no matter the number of features, and the approximate tree path dependent version of *TreeSHAP* has the lowest execution time per instance.

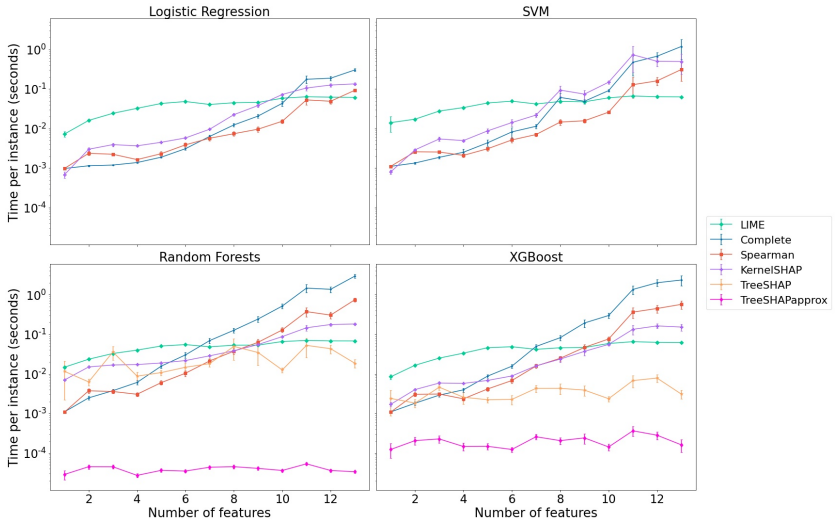


Figure 4: Execution time of each method per instance, averaged by number of features, for each model

Error

Regarding the error, Figure 5 shows the average absolute difference in influence between each method and the *Complete* method (reference). First, we can see that overall, the more features there are in a dataset, the closest (measured by the second metric) the influences are to the *Complete* method.

This is probably due to the fact that usually, the more features there are, the less influence amplitude each individual feature has in the prediction. We also note that no matter the model, methods are ranked in the same way. In low dimension (less than 6 features), *KernelSHAP* is the closest to the *Complete* method, followed by *Spearman*, while *LIME* is the farthest. In higher dimensions, *Spearman* becomes more precise than *KernelSHAP*. *TreeSHAP* (both the approximate and the data dependent version) is more precise than *KernelSHAP*, but still less precise than *Spearman* in high dimensions. Note that the approximate version of *TreeSHAP* is not showed on the graph for XGBoost because its implementation forces its *SHAP* values to be in log odds instead of probabilities, making it impossible to compare to other methods.

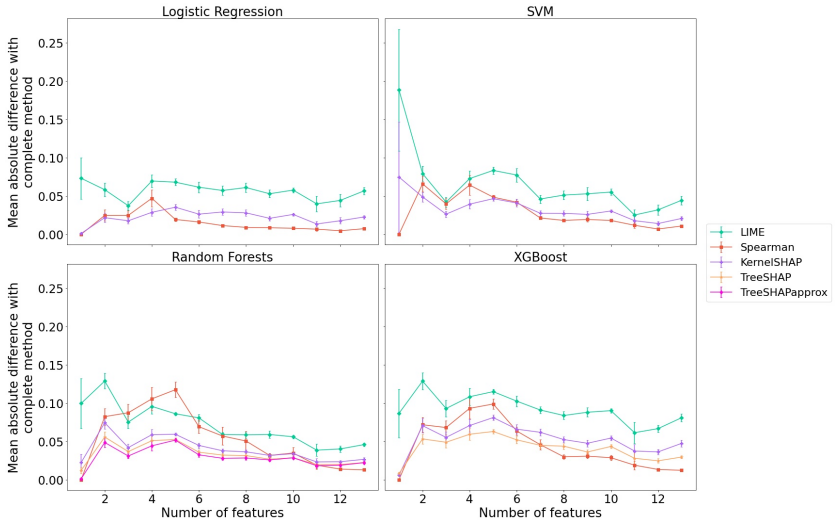
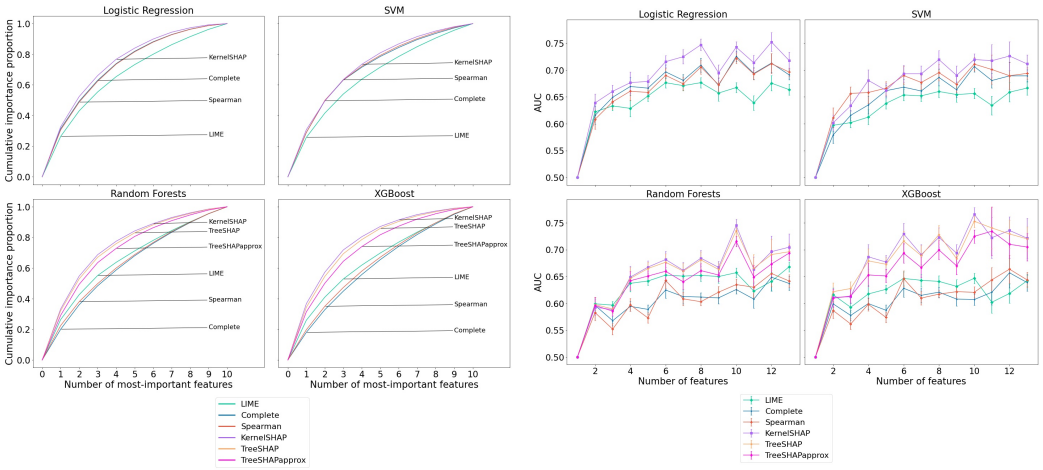


Figure 5: Mean absolute difference of each method with the *Complete*, averaged by number of features, for each model

AUC

We show in Figure 6a the graphical representation of an example of the cumulative feature importance proportion. The figure shows the averaging of the cumulative importance proportion of the most-important features for the 37 datasets having 10 features. This way, for each predictive model and for each method, we obtain a curve from which we compute the third metric: the AUC of the curve.

We see on the figure that some methods present steeper curves than others. For example, with Logistic Regression and SVM, *LIME* gives less proportion of the total importance to the few first most-important features, compared to coalitional-based and *SHAP* methods. For tree-based models, we see that *SHAP*, no matter the method, gives much more importance to the first few most-important features than the other methods.



(a) (b)

Figure 6: (a) Most-important features cumulative importance proportion by method, for each model. Only influences computed on datasets with 10 features are shown. (b) AUC of each method, averaged by number of features, for each model

According to the method for computing AUC illustrated in Figure 6a, we represent the average values of AUC for datasets from 2 to 13 features for each ML model and explanation method in Figure 6b. For all models, we can see that *SHAP* methods tend to produce influences with a higher AUC compared to other methods. This means that *SHAP* methods tend to assign most of the feature importance to fewer most-important features, while other methods tend to distribute the feature importance more uniformly over all features. The two coalitional-based methods seem to generate similar AUCs for the features importance. Finally, *LIME* tends to produce influences with lower AUCs for non-tree-based methods, while it produces AUCs closer to the coalitional-based methods for tree-based methods.

Robustness

Regarding robustness, we show in Figure 7 the local Lipschitz estimates for each model, grouped by method. We used formula 3 with $\epsilon = 0.3$. We show in supplementary data that different values of ϵ did not change the relative order of results. Overall, the explanation method doesn't impact the robustness so much, except for LIME with the Logistic Regression and SVM models, for which the method is far less robust. We can also see that the *Spearman* method is slightly less robust than the *Complete* and *SHAP* methods.

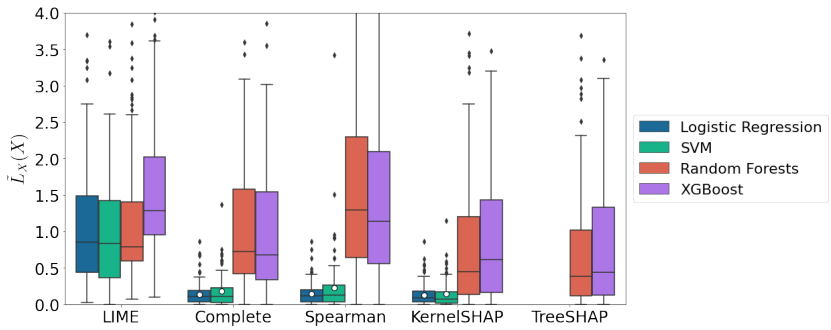


Figure 7: Local Lipschitz estimate for each model, grouped by method. Each box represents the results aggregated for all datasets. The white dot represents the mean value. Due to far outliers, we cropped the plot at $\tilde{L}_X(X) = 4$

Readability

Figure 8, similarly, represents the readability for each model, grouped by method. The explanation method does not impact so much readability. The *Complete* and *Spearman* methods have a slightly lower readability than the other ones. It means that the link between a feature and its explanations tends to be less obvious with these methods than with the others. This is possibly due to the coalitional nature of these methods: by focusing on coalitions, these methods are often able to capture complex interactions between multiple features, meaning that the marginal contribution of a feature is too complex to be explained only by the feature value.

Clusterability

Finally, we show in Figure 9 the two-dimensional clusterability of the methods applied to each model. We can see that LIME has significantly lower

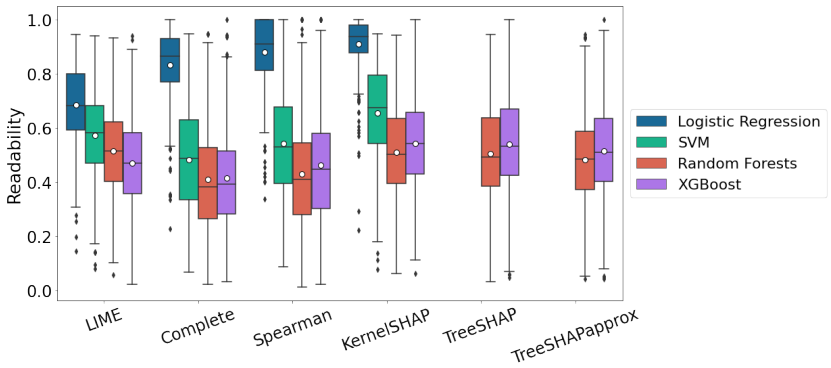


Figure 8: Readability for each model, grouped by method. Each box represents the results aggregated for all datasets. The white dot represents the mean value.

clusterability than the other methods, which have themselves similar clusterability. It means that LIME tends to capture fewer interactions between pairs of features by groups of instances. It may be due to the discretization imposed by LIME on each feature independently of the others.

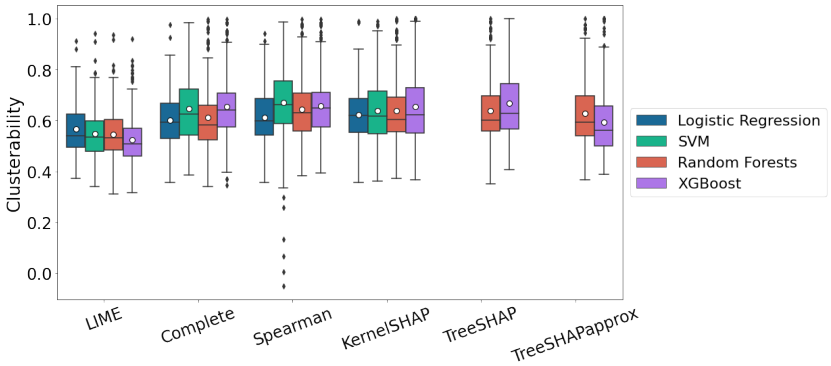


Figure 9: Clusterability for each model, grouped by method. Each box represents the results aggregated for all datasets. The white dot represents the mean value.

4.2. Machine Learning model explanations comparison

Computation time

We show in Figure 10 the computational time per instance needed to compute the explanations of each predictive model, for each explanation method.

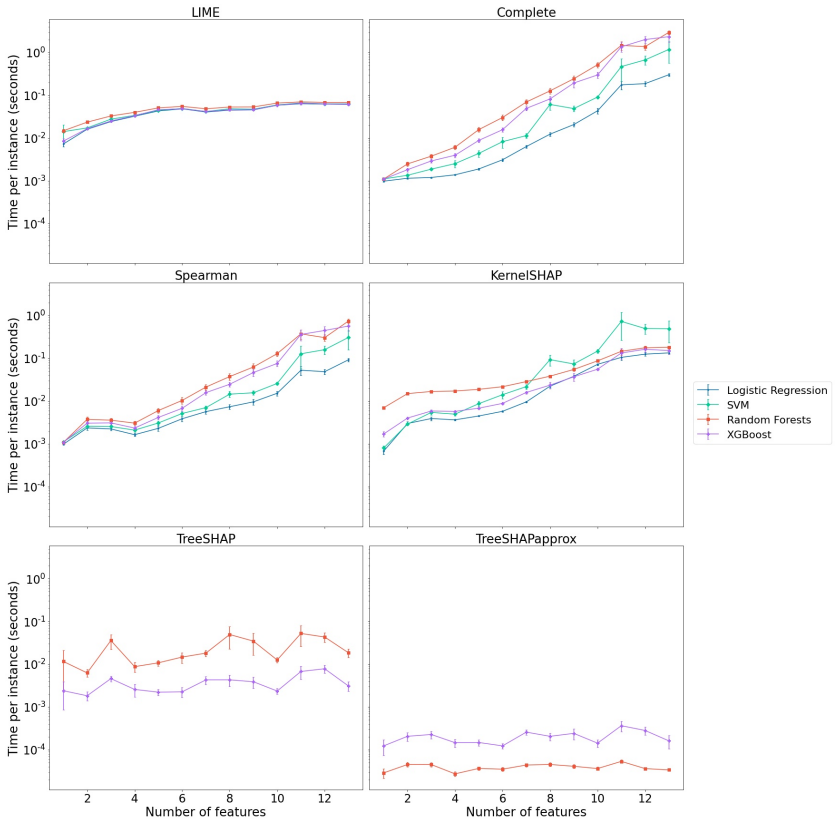


Figure 10: Execution time of each model per instance, averaged by number of features, for each method

We can see that *LIME*'s execution time has almost no inter-model variability: the computation time per instance is the same no matter the model. For the other methods, the ranking of the method's computational performances according to the model is roughly the same, from slowest to fastest: Random Forests, XGBoost, SVM and Logistic Regression. SVM has overall higher variability, presenting steeper curves and higher error bars. SVM even presents outlying results when applied to *KernelSHAP* in higher dimensions. Overall, we do not observe specific behavior of method's computation time in regards to the model used, except for *TreeSHAPapprox* where Random Forests are faster to compute. This may be related to the fact that *TreeSHAPapprox* only considers tree structures, as Random Forests tree structures are simpler than XGBoost's. In general, the faster a model is to train and predict values and the simpler it is, the faster the explanations are to

compute, no matter the method.

Error

We present in Figure 11 the error for each method for each model. The figure does not present the results for *TreeSHAPapprox* because the only relevant model for this method is Random Forests, there is no other model to compare the results with.

For the three model-agnostic methods (*LIME*, *KernelSHAP* and *Spearman*), the Logistic Regression and SVM models generate the most precise explanations compared to the *Complete* method on the same models. We can see that the explanations based on Logistic Regression are usually more precise than SVM's, especially in low dimensions. XGBoost explanations are less precise than Random Forest's, except for the *Spearman* method (similar results are observed). Overall, it seems that the simpler the model, the more precise it is in regards to the *Complete* method.

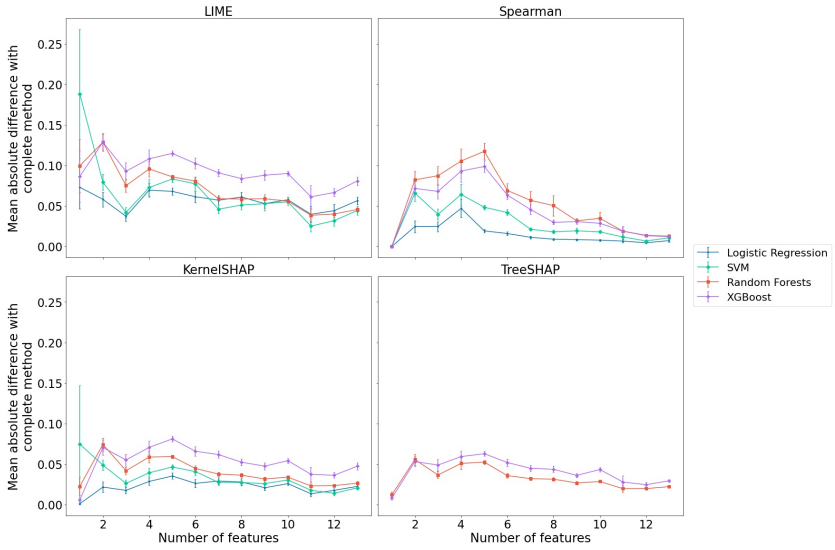


Figure 11: Mean absolute difference of each method with the *Complete*, averaged by number of features, for each model

AUC

Regarding the AUC, we present all the results in Figure 12. We observe that for *LIME* and *KernelSHAP*, there is no significant difference between the AUC of the model's explanations. However, for the coalitional-based

methods, we can see a clear separation between tree-based methods and non tree-based methods: the latter have higher AUC than the others. This means that, when using coalitional-based methods, one should be aware that different models may yield different importance distributions over the features. For the tree-specific methods, we can see that XGBoost generates explanations with slightly higher AUCs than Random Forests on average.

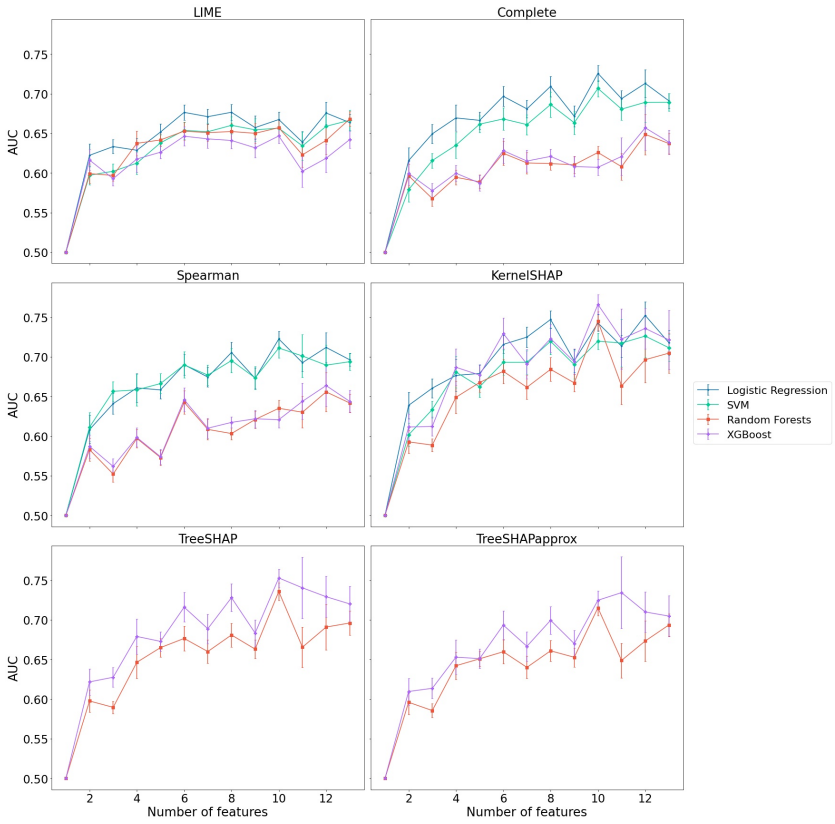


Figure 12: AUC of each model, averaged by number of features, for each method

Regarding robustness, readability, and clusterability, we use the same graphs presented in the previous part to analyze the impact of the model on the explanations regarding these three metrics.

Robustness

We look again at Figure 7 to compare the robustness of the methods applied to each model. We confirm that, except for LIME, the Logistic

Regression and SVM models produce much more robust explanations than Random Forests and XGBoost models. This is probably tied to the complexity of the models. On a one hand, a more complex model is usually harder to explain even for model-agnostic explanation methods, and on the other hand, a more complex model leads to highly non-linear functions, meaning that instances that are close to each other may have different predictions and therefore, different explanations.

Readability

For readability, Figure 8 shows that the explanations made on the Logistic Regression model are much more readable than the ones on the other models. Explanations made on the SVM model fall between the Logistic Regression and the tree-based models in term of readability for most explanation methods. This is probably due to the fact that simpler models tends to draw relationships between individual features and the output without necessarily considering the interaction between features, producing explanations that can be read feature by feature.

Clusterability

Finally, we look at the clusterability of the explanation applied to the ML models by looking at Figure 9. We can see that all models have similar clusterability between them. This may indicate that the model is not important in determining particular subpopulations of explanations by pairs of features, or that it depends more on the considered dataset than on the model.

5. Use-case study and parameters exploration

In this section, we focus on a specific example that a user could face while analyzing their data and building or using explainability tools. In Section 5.1, we show visual representations of the explanations obtained with the different methods on a specific dataset. In Section 5.2, we use the same dataset to present some important parameters for the different methods and visually represent their impact on the explanations.

5.1. Example on a medical dataset

Amongst the OpenML datasets previously studied, we choose a medical dataset, SA-Heart, to compare the explanations given by the different additive methods on an example. This way, we aim to both illustrate and validate

the conclusions of the previous sections regarding explanation methods characteristics. We also aim to highlight practical differences that we can see on the influences of different methods for the same model and dataset.

SA-Heart is a dataset extracted from a larger database of South-Africans detailed in a 1983 study [37]. The extracted dataset is a retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. The dataset is composed of 462 individuals for 10 features. The main objective is to predict the binary target feature '**chd**', a coronary heart disease, according to 9 explanatory factors: **tobacco** (cumulative consumption tobacco), **age** (at the onset), **ldl** (low density lipoprotein cholesterol), **adiposity** (estimation of the body fat percentage), **obesity** (through the body mass index), **family** (family history of heart disease, present or absent), **alcohol** (current alcohol consumption), **sbp** (systolic blood pressure) and **type-A** (Type-A behavior scale). After model training, the different explanatory profiles obtained between the different methods of explanation are compared. By considering a reflection on the end-user side, the health care practitioners, explanatory profiles should be used 1) at the population level (global explanations), for example to highlight high-risk patient profiles, develop new prevention programs, develop new physio-pathological hypotheses but also 2) at the instance level (local explanations), for personalized medicine.

For conciseness in this paper, we limit the analysis to a single machine learning model. We choose Random Forests, as every explanation method that we consider is applicable to it. We present the results with SVM, Logistic Regression and XGBoost models in supplementary data.

In Table 2, we show the values of each metric on the SA-Heart dataset. To enforce the robustness of the results, we calculated the explanations 10 times for each method and averaged the metrics. *TreeSHAP* looks promising, giving the best score in AUC, Robustness and Clusterability, while maintaining correct performances in Computation Time, Error and Robustness. Confirming trends seen in the previous section, *Spearman* is the most precise method w.r.t the *Complete* method, the approximate version of *TreeSHAP* is the fastest, and *LIME* produce the most readable explanations.

To compare the explanations of the different additive methods, we look at global explanations given by each method. We use *SHAP*-like representations to visualize global explanations by aggregating local explanations on the same representation. This way, we build different figures. The first one, in Figure 13, represents a global explanation of the predictive model, given by

	LIME	Complete	Spearman	KernelSHAP	TreeSHAP	TreeSHAPapprox
Time per instance	0.062	0.141	0.036	0.061	0.011	**<0.001**
Error	0.046	0.000	**0.026**	0.034	0.029	0.033
AUC	0.604	0.560	0.550	0.623	**0.625**	0.614
Readability	**0.686**	0.499	0.427	0.679	0.652	0.621
Robustness	0.116	0.099	0.146	0.086	**0.080**	0.095
Clusterability	0.460	0.485	0.506	0.521	**0.522**	0.520

Table 2: Metrics applied to explanations of Random Forests on SA-Heart

each explanation method, by plotting the explanation profile of each feature on a separated line. For each method, the features are sorted in decreasing feature importance, the top one being the most contributing feature on average, while the bottom one being the least contributing feature on average. For each feature, each dot represents an individual from the dataset, its color representing the value of the associated feature. Its position on the x-axis represents the contribution of the feature to the prediction of this individual, and overlapping dots are jittered on the y-axis.

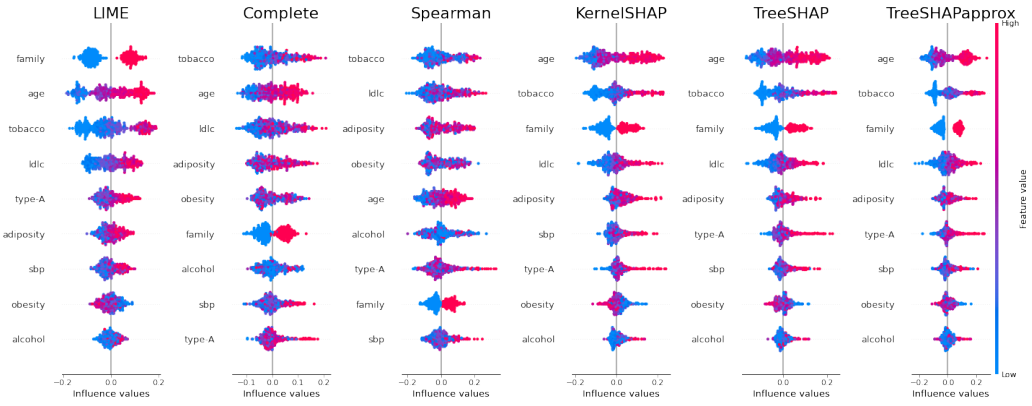


Figure 13: Summary plots of each method on the SA-Heart dataset

We can see that most of the features have similar ranking among the different methods: tobacco and age are the two most important features except for the *Spearman* method which ranks age 5$^{\text{th}}$. On the opposite side, alcohol, spb, and type-A are always in the 4 least important features. These features have also similar explanation profiles. Conversely, some other features exhibit more marked difference depending on the methods. The most important difference is observed on the binary feature family history of heart

disease. This feature is assigned fairly low importance by the coalitional-based method, relatively high importance (3rd most important feature) by *SHAP* methods, and very high importance by *LIME* (most important feature). Obesity and adiposity have also different influences depending on the method: obesity is ranked second least contributing by *LIME* and *SHAP*, but more important by the coalitional-based methods. It is important to note that obesity and adiposity are highly correlated (Spearman's correlation r=0.72). We hypothesize that it may be the reason for such differences. Overall, the three *SHAP* methods give similar explanations and have almost identical ranking of the features. From a global perspective, we can also see that *SHAP* and *LIME* present a more homogeneous "gradient" of colors for the explanations, where coalitional-based methods present mixed up colors in the explanations. This means that *LIME* and *SHAP*'s explanations are more locally monotonic, in the sense that the influence value of a feature for an individual is more locally correlated to the value of the feature for *LIME* and *SHAP* than it is for coalitional-based methods. This also illustrates well the values of readability seen in Table 2.

The second visualization that we present are Partial Dependence Plots (PDP). PDPs focus on the relationship between a feature and the influence of this feature on the model's prediction by plotting each pair of feature value and influence value on a 2-dimensional axis. We compare the PDPs of several important features in Figure 14.

Looking at the PDPs for the **age** feature, we show that *LIME* seems to form clusters of points around specific cut-off age values. To a lesser extent, this phenomenon can also be seen on the other *SHAP* methods. Conversely, coalitional-based methods have similar PDPs, and do not seem to find such cut-offs. However, it seems to be a special behavior of the explanation at specific ages. For example, subjects around 50 years have a marked lower contribution of this feature to the prediction of the presence of coronary heart disease than people even slightly younger or older. This may hint at an overfitting of the machine learning model that would not have been captured by the other explanation methods. The explanation of the tobacco feature also largely differs among explanation methods. Where all the methods agree on attributing a low value to non-smoking individuals, the evolution of the contribution varies with the quantity of tobacco. Once again, *LIME* and *SHAP* explanations seem to find a cut-off value for tobacco consumption, of around 7 and 9 respectively, while coalitional-based methods capture a non-monotonic, more complex relationship.

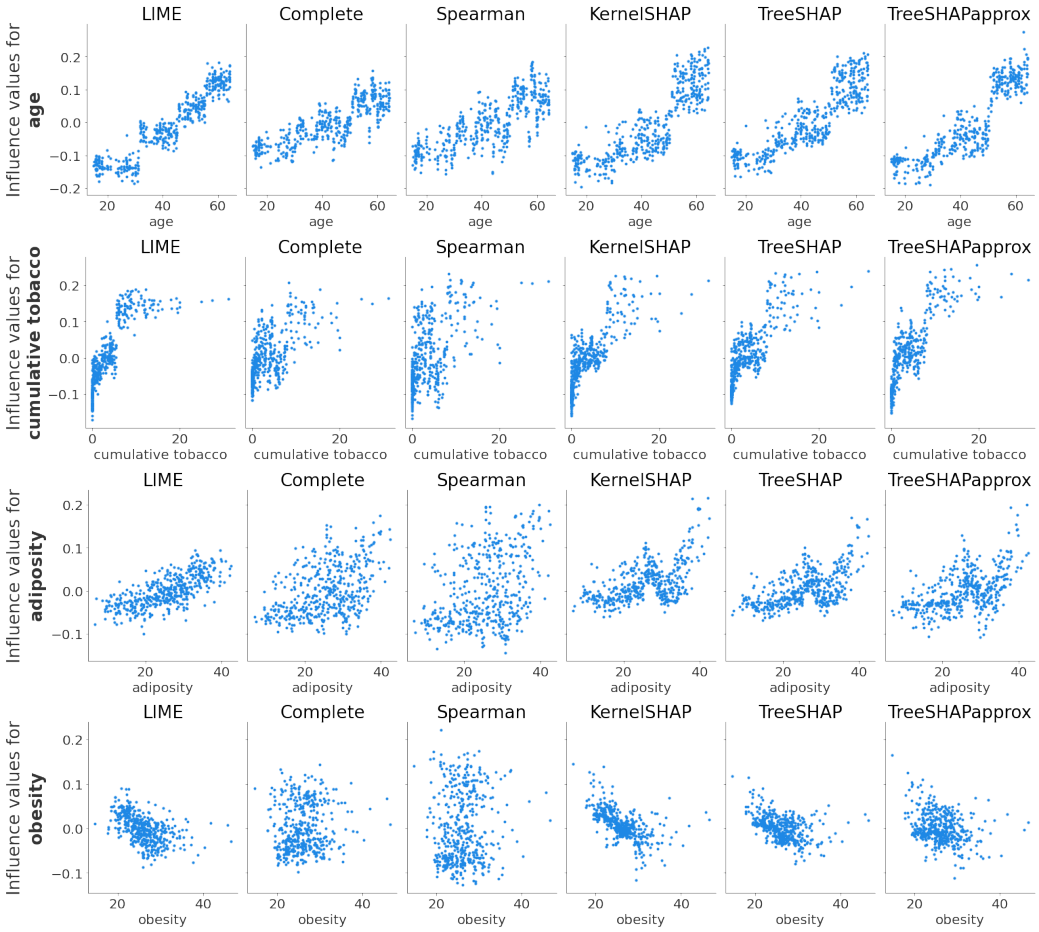


Figure 14: Partial dependence plots of age, tobacco, adiposity and obesity for each method

We also look at adiposity PDPs. Once again, the three *SHAP* explanations are close to each other. Interestingly, they capture a non-monotonic relationship between the feature and the outcome, giving people around 30% of adiposity a higher influence for this feature (in absolute value) than people close to this value. This relationship seems to be captured in a lesser extent by coalitional-based methods, but not captured at all by *LIME*. We also note that the *Complete* and *Spearman* influences are more scattered, which means that more variance exists amongst subjects of the same adiposity for these methods than for the others.

Lastly, looking at obesity PDPs, *LIME* and *SHAP* methods find a negative relationship between obesity and the chd prediction. This seems counter

intuitive, as obesity is a strong known comorbidity factor of heart diseases. As previously mentioned, obesity and adiposity are strongly correlated (r=0.72), and it may be the reason for such observation. Furthermore, we have mentioned in section 2.1 that *SHAP* works under the hypothesis that features are independent, but with such correlation, it is very unlikely that obesity and adiposity are independent. To better understand the relationship between these two features, as found by the methods, we plot in Figure 15 the influence values of adiposity and obesity given by each method.

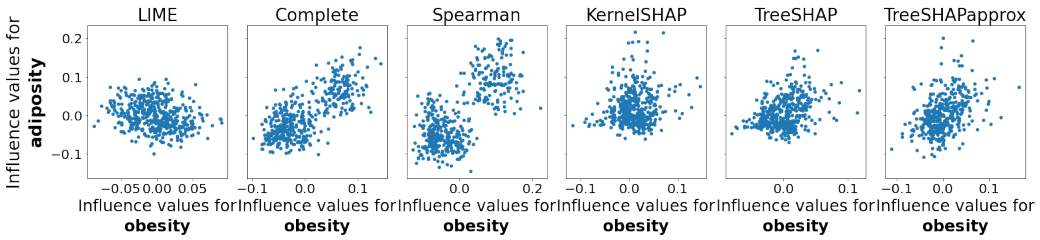


Figure 15: Influence value of adiposity against the influence value of obesity

The *Complete* and *Spearman* methods seem to find a positive correlation between the influences of the two features: when an individual is assigned a high influence value for obesity, a high influence value for adiposity is usually assigned, and conversely. We can even distinguish two clusters of individuals: one for individuals that have a high influence value for both features, and one for individuals that have a low influence value for both features. Such pattern is not found by *LIME* or *SHAP*, thus confirming the lack of ability of these methods to consider dependent features. This shows the limits of clusterability as a global metric to evaluate explanations. As seen in Table 2, on this dataset, the three *SHAP* methods have an overall higher clusterability than coalitional-based methods. However, when we consider pairs of features individually, we see that coalitional-based methods can capture clusters that *SHAP* fails to capture.

On a more global scale, we see that *LIME* and *SHAP* produce explanations that are easier to read at a first glance compared to *Complete* and *Spearman* explanations. However, *LIME* and *SHAP* seem to capture different cut-offs and relationships, and it is hard to confirm such values without further biological knowledge. Coalitional-based methods seem to produce explanations that are harder to read on a global scale, but more precise at an individual level and able to take into account the dependencies between

features. PDPs for all features are available in supplementrary data.

5.2. Hyperparameter exploration for the explanation methods

Most explanation methods have several parameters that can change the way the explanations are generated, and so their values. Previously, we showed results for a single set of parameters for each method. In this section, we present new results on the SA-Heart dataset by taking different values of several parameters. For conciseness, we present the results for only a single model, Random Forests, although we observe similar results on the other models as well.

5.2.1. LIME

The first parameter we investigate is one of *LIME*'s most important parameters: the number of samples drawn from the distribution to generate the local linear model to explain an instance. Its default value is 5000, but as the computation time scales linearly with this number, we limited this number of samples to 100 in our previous experiments. In Figure 16, we visually show the effect of different values of this parameter on the explanations.

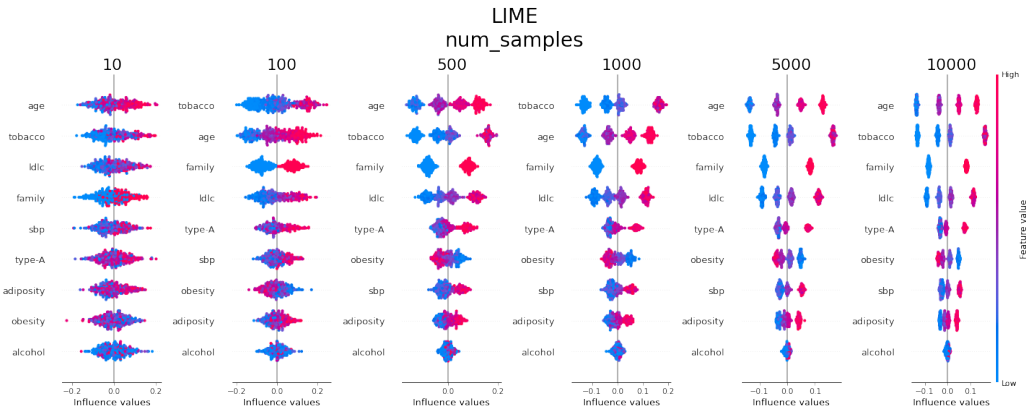


Figure 16: Summary plot of the explanations given by LIME on the SA-Heart dataset with different values for the number of samples drawn to create the local model for each explanation.

We can immediately see that the number of samples impacts the global explanations for the dataset. However, looking at the relative importance of the features, knowing they are sorted in descending importance from top to bottom, we can see that this parameter does not change the feature importance so much. As the number of samples increases, we can see for each

feature that LIME's explanations are grouped by features values around specific influence values. This creates vertical stripes that get thinner when the number of samples increases for each explanation. To have a better visualisation of this phenomenon, we look at the partial dependence plot of the **age** feature in Figure 17. *LIME* tends to discretize age values, with influence values more and more grouped and homogeneous as the number of samples increases. This goes to an extreme case when taking 10000 samples, with the same influence values for an entire age group. As we see that the age category of an individual defines almost entirely the influence value given by LIME for this feature. This can be an incorrect explanation, as this would mean that the model does not consider any interaction between the age and other features to make a prediction, while we know that Random Forests use tree depth and node successions to take into account the relationship between features. This would also mean that the model has not enough granularity to consider the features as *continuums* and instead considers only categories, which again is certainly incorrect regarding Random Forests. However, when looking at the relationship between the number of samples and the local Lipschitz estimate in Figure 18, the robustness increases with the number of samples per explanation. This underlines the limits of robustness and, in a broader extent, the limits of objective metrics to evaluate the explanations. Despite being systematically measurable on all the explanations, they must be taken as a whole to qualify and compare explanations. Human and expert reading are always necessary to validate the quality of the explanations.

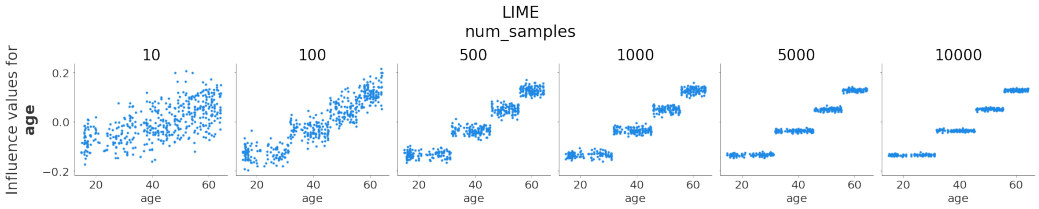


Figure 17: Partial dependence plot of the explanations given by LIME for the feature **age** on the SA-Heart dataset with different values for the number of samples drawn to create the local model for each explanation.

Another important parameter for LIME is the kernel width. The kernel is used by LIME to weight the samples drawn to create the local linear model considering their distance to the instance for which we want to explain the prediction. The farthest the drawn sample is from the instance, the less weight it has in the local linear model. This enforces the notion of locality

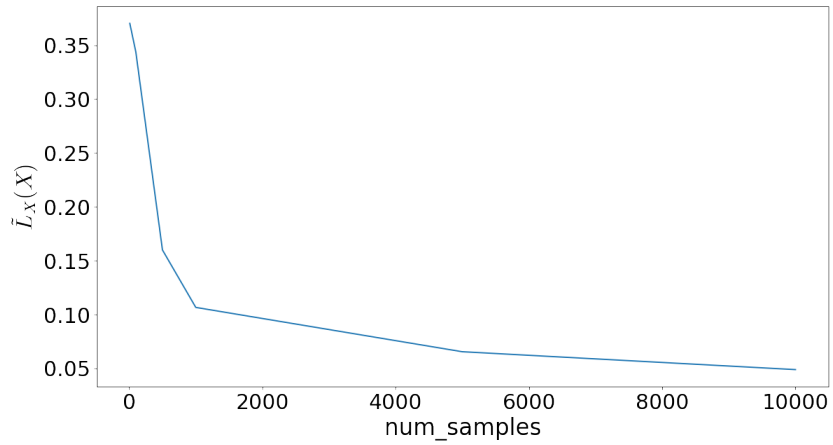


Figure 18: Local Lipschitz estimate of LIME explanations on the SA-Heart dataset according to the number of samples drawn for each explanation.

for the linear model, and the higher the kernel width, the less local the linear model is. In [38], the authors insist on the trade-off between stability (the equality of the local model's coefficient through repeated trials) and adherence (the R^2 performance of the local model). The article shows that such trade-off is mainly determined by the value of the kernel width. The base value is $0.75 \times \sqrt{d}$ with d the number of features (2.25 for the SA-Heart dataset)[7]. We show in Figure 19 a partial dependence plot of the age feature for different values of kernel width, but we observe very similar results for each feature as shown in supplementary figures.

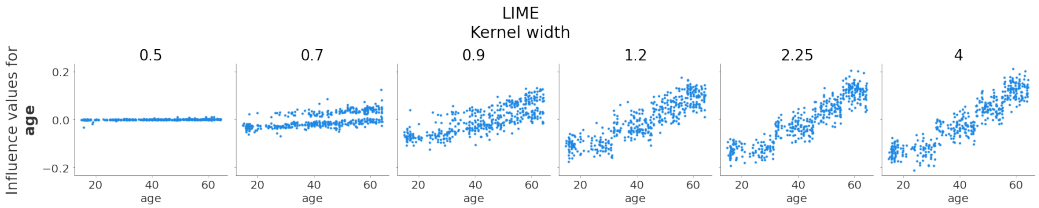


Figure 19: Partial dependence plot of the explanations given by LIME for the feature **age** on the SA-Heart dataset with different values for the kernel width.

We can see that the main impact of the kernel width is the amplitude of the explanations: lower values of kernel width results in flattened values

[7]LIME documentation

of influence that mix together in an unreadable fashion, while higher values of kernel width leads to the usual "boxes" that LIME creates for the explanations. The default value (2.25 for SA-Heart) seems to be on the higher end. This could mean that the default value of kernel width makes the linear model not local enough, giving a high weight to samples that are far from the instance we want to explain. A more appropriate kernel width value, for this dataset and model, may be closer to a value of 1.

5.2.2. Spearman

For the *Spearman* method, we look at its single parameter: the proportion of subsets of features (or coalitions) taken into account by the method to compute the contributions. This parameter is called *rate* and it goes from 0 excluded (we need at least a coalition) to 1 included. A rate of 1 gives exactly the same algorithm than the *Complete* method. We show in Figure 20 the partial dependence plot of the age feature for different values of rate, and again we observe very similar behaviors on other dependence plots.

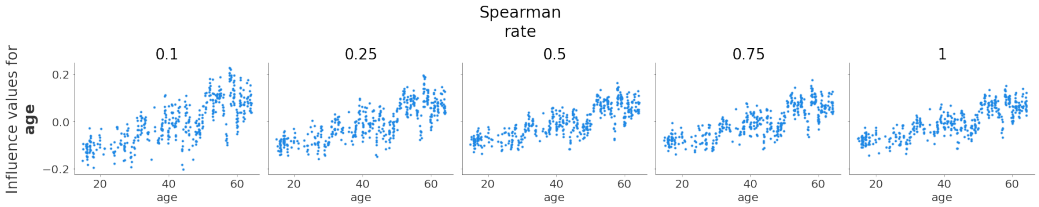


Figure 20: Partial dependence plot of the explanations given by the Spearman method for the feature **age** on the SA-Heart dataset with different values for the rate.

We can see that a higher rate produce less scattered explanations. However, after 0.5, the change is barely visible. We can conclude that the rate effectively controls the degree of approximation, and on this dataset and model, the *Spearman* method with a rate of 0.25 is a good approximation of the *Complete* method, and it is a very good approximation for rates of 0.5 and more.

5.2.3. SHAP

Next, we look at the *KernelSHAP* method. When using it, a so-called "background" dataset must be used to provide relevant samples to train the

explainer. However, this can heavily slow down the process as the method draws samples around each of the background samples. As advised by the documentation, if the method takes too much time to compute, we can use a clustering method (namely KMeans) to extract the few most relevant samples in the train dataset to represent the data distribution. We then refer to this number of "most relevant samples" as "Number of background samples". We observe in Figure 21 the age feature dependence plot for different values of number of background samples.

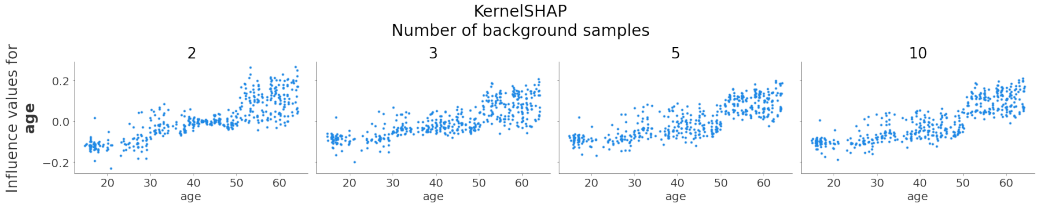


Figure 21: Partial dependence plot of the explanations given by the KernelSHAP method for the feature **age** on the SA-Heart dataset with different values for the number of background samples.

We can see that even with 2 background samples, the explanations are already close to the one with 10 background samples, and that there is almost no difference between 3, 5 and 10 background samples. While this obviously depends on the dataset number of samples and distribution, we can still make the hypothesis that in most cases, we can drastically reduce the number of background samples with the KMeans algorithm in order to reduce the amount of time taken by the method to compute the explanations.

Finally, we look at the number of samples drawn from the distribution (based on the background samples previously mentioned) by creating perturbations. This parameter is called nsamples and its default value is $2d+2048$, with d the number of features in the dataset (2066 for the SA-Heart dataset)[8]. We show in Figure 22 the age feature dependence plot for different values of nsamples.

As we increase the number of samples, we can see that we quickly reach a plateau around 300 nsamples. Under this value, we can still see the shape of the explanation, but we can also see many samples that are given an influence of 0, which is incorrect. Nevertheless, we can firmly say that the

[8]KernelSHAP documentation

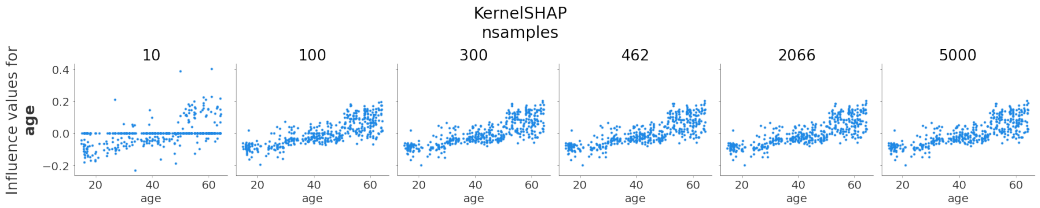


Figure 22: Partial dependence plot of the explanations given by the KernelSHAP method for the feature **age** on the SA-Heart dataset with different values for the number of drawn samples.

default value (2066) is too many samples for this dataset and model, and that we can reduce this number a lot to compute the explanations.

Overall, we find that the impact of the parameters really depends on the method, and each parameter has a different impact. *Spearman* has a main parameter that can control the trade-off between the degree of approximation and the computation time of the method. *KernelSHAP* and *LIME* have several parameters, allowing the user to control the robustness, computation time and the locality of the explanations to some extent, but they require good knowledge of the explanation method.

6. Lessons-learned for the use of additive local methods

Method name		**Advantages**		**Drawbacks**	
Coalitional based	Complete	Consider feature interdependence	Exact shapley values	Slow in high dimension	
	Spearman		Parameter α to control the level of approximation	Global explanations can be hard to read	Less robust on tree-based models
LIME		Fast in high dimension Various parameters to control robustness and locality trade-offs		Slow in low dimension Low quality explanations Tends to miss non linear and non monotonic influences Not robust with simple models Can miss relationship between pairs of features	
SHAP	KernelSHAP	Easy to interpret global explanations	Various parameters	Approximations may be inprecise	Slow in high dimension
	TreeSHAP		Very fast in low and high dimensions		Tree-based models specific
	TreeSHAPapprox				

Table 3: Summary table of advantages and drawbacks of each method (v2)

Table 3 summarizes advantages and drawbacks of each method studied in this paper. Overall, we highlight the fact that coalitional-based methods should be better at producing precise local explanations while *SHAP* should be better at producing coherent and easily interpretable global explanations. It is also confirmed by the fact that *SHAP* tends to assign more

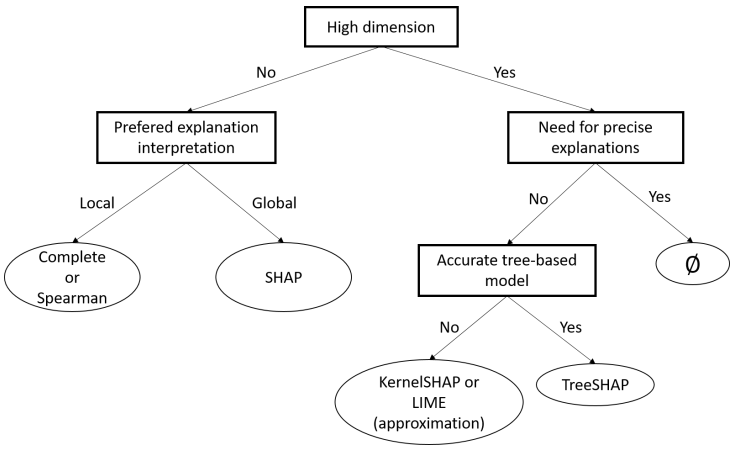


Figure 23: Roadmap for the most appropriate use of methods

importance to few features than other methods, producing global explanations that more concise, but potentially hiding other feature contributions and inter-dependences. *Spearman*'s explanations are overall slightly less robust than the other methods, and coalitional-based methods are slightly less readable. *LIME* has several drawbacks, one of the most distinguishable being its tendency to miss the interactions between features as well as the complex influences. Regarding method parameters, each method offers different number and types of parameters. *Spearman*'s α allows users to easily control the trade-off between computation time and degree of approximation. *LIME*'s parameters, numerous and complex, allows a fine-tuning of the method, but requires an extensive knowledge of *LIME*'s behavior regarding these parameters and the model and dataset considered. *KernelSHAP*'s parameters are similar to *LIME*'s, but they seem to induce less important changes in the resulting explanations, allowing them to be used to reduce computation time without much degrading explanation quality.

We use all the results presented in this paper to show a simplified roadmap in the form of a decision tree in Figure 23 with the intent to help readers finding the most suitable explanation method according to their datasets and objectives.

On this figure, high dimension represents the number of features present in the studied dataset. Indeed, there is no "hard" cut-off to define when it goes from low to high dimension, but with our experiments, we can consider this cut-off somewhere between 11 and 15 features, depending on the

dataset complexity and the user computational time and material available. "Accurate tree-based model" represents the ability of training a satisfactory (defined by the user's objectives) tree-based model on the dataset. The model can then be explained thanks to the optimization done in *TreeSHAP*. If the desired model is not tree-based, we advise the user to look at *KernelSHAP* and *LIME*'s parameters to reduce the number of background samples and perturbation samples respectively, until the explanations are computed in a reasonable time. However, we warn about the potential loss of precision and robustness induced by such method approximations.

Finally, we show that *SHAP* and *LIME* can make important approximations in some cases, and that coalitional-based methods cannot be executed in reasonable time in high dimension. This leaves an empty space for high dimension precise explanations that is not yet addressed to our knowledge.

7. Conclusion and perspectives

In this paper we performed a practical analysis of several local explainability methods for tabular data. Our findings indicate that there is not a single method that is the most appropriate for every usage. Therefore, this thorough analysis allowed to identify strengths and limitations of each method along with practical recommendations on which method is most suitable for the use case of the user.

We have seen that the choice of the predictive machine learning model does not impact the general behavior of the explanation methods much. However, except with *LIME*, simpler predictive models tends to produce more readable and robust explanations, but tree-based models allow for the use of *TreeSHAP* which is more efficient.

Regarding explanation methods, the *Complete* is of course the most accurate but suffers from very long computational time. Nevertheless, *Coalitional*-based methods allow an acceptable computation time while maintaining a strong precision of explanations. On the contrary, *LIME* and *SHAP* methods offer a more intelligible global view of feature effects.

The greatest problem arises when high dimension (*i.e.,* high number of features) is involved, as it is often the case in statistics and machine learning. In this case, the exponential complexity of *Coalitional-based* methods make them too long to compute. Indeed, the worst case scenario is the need for high precision local explanations in high dimension since there is a clear lack of methods addressing this problem in the current literature. However,

it is still possible to have local explanations with limited quality in high dimension, with the level of quality mostly depending on the time available for the user to generate such explanations. It is thus a very interesting future axis of work to benchmark the performances, in terms of precision of local explanations, of every local explainability method in a high dimension context under the constraint of a time limit. This would add value to our recommendations by filling out the 'high-precision in high-dimension' gap identified in our study. It would also be interesting to look into other machine learning models, especially deep neural networks which are more and more used. The high complexity of this type of models hints at a different behavior for the explanation methods, but also an increase in computation time.

Acknowledgments

We gratefully acknowledge the Programme d'Investissements d'Avenir, and the Agence Nationale pour la Recherche for the support of the national infrastructure "ECELLFrance: Development of mesenchymal stem cell-based therapies" (PIA-ANR-11-INBS-005) and the PhD grant EUR CARe N°ANR-18-EURE-0003. We also thank the French National Association for Research and Technology (ANRT) and Kaduceo company for providing us with a PhD grant (no. 2020/0964).

References

[1] W. K. Diprose, N. Buist, N. Hua, Q. Thurier, G. Shand, R. Robinson, Physician understanding, explainability, and trust in a hypothetical machine learning risk calculator, Journal of the American Medical Informatics Association : JAMIA 27 (4) (2020) 592–600. doi:10.1093/jamia/ocz229.
URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7647292/

[2] J. Amann, A. Blasimme, E. Vayena, D. Frey, V. I. Madai, Explainability for artificial intelligence in healthcare: a multidisciplinary perspective, BMC Medical Informatics and Decision Making 20 (2020) 310. doi:10.1186/s12911-020-01332-6.
URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7706019/

[3] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 4765–4774.

[4] M. Ribeiro, S. Singh, C. Guestrin, "why should i trust you?": Explaining the predictions of any classifier, 2016, pp. 97–101. doi:10.18653/v1/N16-3020.

[5] G. Ferrettini, E. Escriva, J. Aligon, J.-B. Excoffier, C. Soulé-Dupuy, Coalitional strategies for efficient individual prediction explanation, Information Systems Frontiers (2021). doi:10.1007/s10796-021-10141-9.
URL https://doi.org/10.1007/s10796-021-10141-9

[6] C. Molnar, A guide for making black box models explainable, 2018.
URL https://christophm.github.io/interpretable-ml-book/v

[7] E. Doumard, J. Aligon, E. Escriva, J. Excoffier, P. Monsarrat, C. Soulé-Dupuy, A comparative study of additive local explanation methods based on feature influences, in: K. Stefanidis, L. Golab (Eds.), Proceedings of the 24th International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data (DOLAP) co-located with the 25th International Conference on Extending Database Technology and the 25th International Conference on Database Theory (EDBT/ICDT 2022), Edinburgh, UK, March 29, 2022, Vol. 3130 of CEUR Workshop Proceedings, CEUR-WS.org, 2022, pp. 31–40.
URL http://ceur-ws.org/Vol-3130/paper4.pdf

[8] S. M. Lundberg, G. G. Erion, S.-I. Lee, Consistent individualized feature attribution for tree ensembles, arXiv preprint arXiv:1802.03888 (2018).

[9] G. Ferrettini, J. Aligon, C. Soulé-Dupuy, Explaining single predictions: A faster method, in: A. Chatzigeorgiou, R. Dondi, H. Herodotou, C. Kapoutsis, Y. Manolopoulos, G. A. Papadopoulos, F. Sikora (Eds.), SOFSEM 2020: Theory and Practice of Computer Science, Springer International Publishing, Cham, 2020, pp. 313–324.

[10] D. Slack, S. Hilgard, E. Jia, S. Singh, H. Lakkaraju, Fooling lime and shap: Adversarial attacks on post hoc explanation methods, Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (2020).

[11] D. Garreau, U. von Luxburg, Explaining the explainer: A first theoretical analysis of lime, in: Proceedings of the 23rd International Conference

on Artificial Intelligence and Statistics (AISTATS), Vol. 108 of Proceedings of Machine Learning Research, PMLR, 2020, pp. 1287–1296.
URL http://proceedings.mlr.press/v108/garreau20a.html

[12] T. Laugel, X. Renard, M.-J. Lesot, C. Marsala, M. Detyniecki, Defining Locality for Surrogates in Post-hoc Interpretablity, Workshop on Human Interpretability for Machine Learning (WHI) - International Conference on Machine Learning (ICML) (2018).
URL https://hal.sorbonne-universite.fr/hal-01905924

[13] R. ElShawi, Y. Sherif, M. Al-Mallah, S. Sakr, Interpretability in healthcare: A comparative study of local machine learning interpretability techniques, Computational Intelligence (2020).

[14] E. Štrumbelj, I. Kononenko, Towards a model independent method for explaining classification for individual instances, in: International Conference on Data Warehousing and Knowledge Discovery, Springer, 2008, pp. 273–282.

[15] E. Strumbelj, I. Kononenko, An Efficient Explanation of Individual Classifications Using Game Theory, J. Mach. Learn. Res. 11 (2010) 1–18, publisher: JMLR.org.

[16] E. Štrumbelj, I. Kononenko, Explaining prediction models and individual predictions with feature contributions, Knowledge and Information Systems 41 (3) (2014) 647–665. doi:10.1007/s10115-013-0679-x.
URL https://doi.org/10.1007/s10115-013-0679-x

[17] L. S. Shapley, 17. A value for n-person games, Princeton University Press, 2016.

[18] S. Lipovetsky, M. Conklin, Analysis of regression in game theory approach, Applied Stochastic Models in Business and Industry 17 (4) (2001) 319–330. doi:10.1002/asmb.446.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/asmb.446

[19] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLOS ONE 10 (7), publisher: Public Library of Science (2015). doi:10.1371/journal.pone.0130140.

URL https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0130140

[20] A. Datta, S. Sen, Y. Zick, Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems, in: 2016 IEEE Symposium on Security and Privacy (SP), 2016, pp. 598–617. doi:10.1109/SP.2016.42.

[21] A. Shrikumar, P. Greenside, A. Kundaje, Learning important features through propagating activation differences, in: International Conference on Machine Learning, PMLR, 2017, pp. 3145–3153.
URL http://proceedings.mlr.press/v70/shrikumar17a.html

[22] P. Linardatos, V. Papastefanopoulos, S. Kotsiantis, Explainable ai: A review of machine learning interpretability methods, Entropy 23 (1) (2021) 18.

[23] I. E. Kumar, S. Venkatasubramanian, C. Scheidegger, S. Friedler, Problems with shapley-value-based explanations as feature importance measures, in: International Conference on Machine Learning, PMLR, 2020, pp. 5491–5500.

[24] G. Van den Broeck, A. Lykov, M. Schleich, D. Suciu, On the tractability of shap explanations, in: Proceedings of AAAI, 2021.

[25] N. Burkart, M. F. Huber, A survey on the explainability of supervised machine learning, J. Artif. Int. Res. 70 (2021) 245–317. doi:10.1613/jair.1.12228.
URL https://doi.org/10.1613/jair.1.12228

[26] G. Vilone, L. Longo, Notions of explainability and evaluation approaches for explainable artificial intelligence, Information Fusion 76 (2021) 89–106. doi:https://doi.org/10.1016/j.inffus.2021.05.009.
URL https://www.sciencedirect.com/science/article/pii/S1566253521001093

[27] R. El Shawi, Y. Sherif, M. Al-Mallah, S. Sakr, Interpretability in healthcare a comparative study of local machine learning interpretability techniques, in: 2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS), 2019, pp. 275–280. doi:10.1109/CBMS.2019.00065.

[28] X. Man, E. P. Chan, The best way to select features? comparing mda, lime, and shap, The Journal of Financial Data Science 3 (1) (2021) 127–139. arXiv:https://jfds.pm-research.com/content/3/1/127.full.pdf, doi:10.3905/jfds.2020.1.047.
URL https://jfds.pm-research.com/content/3/1/127

[29] J. Duell, X. Fan, B. Burnett, G. Aarts, S.-M. Zhou, A comparison of explanations given by explainable artificial intelligence methods on analysing electronic health records, in: 2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI), IEEE, 2021, pp. 1–4.

[30] T. Miller, Explanation in artificial intelligence: Insights from the social sciences, Artificial Intelligence 267 (2019) 1–38. doi:https://doi.org/10.1016/j.artint.2018.07.007.
URL https://www.sciencedirect.com/science/article/pii/S0004370218305988

[31] M. Robnik-Šikonja, M. Bohanec, Perturbation-Based Explanations of Prediction Models, Springer International Publishing, Cham, 2018, pp. 159–175. doi:10.1007/978-3-319-90403-0_9.
URL https://doi.org/10.1007/978-3-319-90403-0_9

[32] A.-p. Nguyen, M. R. Martínez, On quantitative aspects of model interpretability, arXiv:2007.07584 [cs, stat]ArXiv: 2007.07584 (Jul. 2020).
URL http://arxiv.org/abs/2007.07584

[33] D. Alvarez-Melis, T. S. Jaakkola, On the Robustness of Interpretability Methods, Workshop on Human Interpretability for Machine Learning (WHI) - International Conference on Machine Learning (ICML)ArXiv: 1806.08049 (Jun. 2018).
URL http://arxiv.org/abs/1806.08049

[34] M. Nauta, J. Trienes, S. Pathak, E. Nguyen, M. Peters, Y. Schmitt, J. Schlötterer, M. van Keulen, C. Seifert, From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai (2022). doi:10.48550/ARXIV.2201.08164.
URL https://arxiv.org/abs/2201.08164

[35] C.-K. Yeh, C.-Y. Hsieh, A. Suggala, D. I. Inouye, P. K. Ravikumar, On the (in)fidelity and sensitivity of explanations, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 32, Curran Associates, Inc., 2019.
URL https://proceedings.neurips.cc/paper/2019/file/a7471fdc77b3435276507cc8f2dc2569-Paper.pdf

[36] Z. Carmichael, W. J. Scheirer, A framework for evaluating post hoc feature-additive explainers (2021). doi:10.48550/ARXIV.2106.08376.
URL https://arxiv.org/abs/2106.08376

[37] Rossouw, du Plessis, Benade, Jordaan, Kotze, Jooste, Ferreira, Coronary risk factor screening in three rural communities-the coris baseline study, South African medical journal 64 (12) (1983) 430–436.

[38] G. Visani, E. Bagli, F. Chesani, OptiLIME: Optimized LIME Explanations for Diagnostic Computer Algorithms, arXiv:2006.05714 [cs, stat]ArXiv: 2006.05714 (Feb. 2022).
URL http://arxiv.org/abs/2006.05714