A k-additive Choquet integral-based approach to approximate the SHAP values for local interpretability in machine learning

Guilherme Dean Pelegrina, Leonardo Tomazeli Duarte, Michel Grabisch

A k-additive Choquet integral-based approach to approximate the SHAP values for local interpretability in machine learning

Guilherme Dean Pelegrina[a,b,*], Leonardo Tomazeli Duarte[a], Michel Grabisch[b,c]

[a] *School of Applied Sciences - University of Campinas, Limeira, Brazil*
[b] *Centre d'Économie de la Sorbonne - Université Paris I Panthéon-Sorbonne, Paris, France*
[c] *Paris School of Economics - Université Paris I Panthéon-Sorbonne, Paris, France*

Abstract

Besides accuracy, recent studies on machine learning models have been addressing the question on how the obtained results can be interpreted. Indeed, while complex machine learning models are able to provide very good results in terms of accuracy even in challenging applications, it is difficult to interpret them. Aiming at providing some interpretability for such models, one of the most famous methods, called SHAP, borrows the Shapley value concept from game theory in order to locally explain the predicted outcome of an instance of interest. As the SHAP values calculation needs previous computations on all possible coalitions of attributes, its computational cost can be very high. Therefore, a SHAP-based method called Kernel SHAP adopts a strategy that approximates such values with less computational effort. However, we see two weaknesses in Kernel SHAP: its formulation is difficult to understand and it does not consider further game theory assumptions that could reduce the computational cost. Therefore, in this paper, we propose a novel approach that addresses such weaknesses. Firstly, we provide a straightforward formulation of a SHAP-based method for local interpretability by using the Choquet integral, which leads to both Shapley values and Shapley interaction indices. Thereafter, we propose to adopt the concept of k-additive games from game theory, which contributes to reduce the computational effort when estimating the SHAP values. The obtained results attest that our proposal needs less computations on coalitions of attributes to approximate the SHAP values.

Keywords: Local interpretability; Choquet integral; Machine learning; Shapley values

*Corresponding author
Email addresses: guidean@unicamp.br (Guilherme Dean Pelegrina), leonardo.duarte@fca.unicamp.br (Leonardo Tomazeli Duarte), michel.grabisch@univ-paris1.fr (Michel Grabisch)

1. Introduction

In the last decade, Machine Learning (ML) models have been used to deal with problems that directly affect people's life, such as consumer credit scoring (Kruppa et al., 2013), cybersecurity (Xin et al., 2018), disease detection (Ahsan & Siddique, 2022) and patient care evaluation (Ben-Israel et al., 2020). Aiming at dealing with such problems, complex ML models have been proposed to achieve good solutions in terms of accuracy. Examples include random forests (Fawagreh et al., 2014; Biau & Scornet, 2016), deep neural networks (LeCun et al., 2015; Goodfellow et al., 2016) and gradient boosting algorithms (Bentéjac et al., 2021). Despite the good performance towards accuracy, they act as black box models, as the obtained results (predictions and/or classifications) are difficult to be interpreted. Therefore, there is an inherent trade-off between adopting an accurate model, whose structure is frequently complex, or an interpretable model, such as linear/logistic regression (Molnar, 2021).

Interpretability plays an important role in machine learning-based automatic decisions and has been discussed in several recent works in the ML community (Lipton, 2018; Gilpin et al., 2018; Carvalho et al., 2019; Molnar, 2021; Setzu et al., 2021). As stated by Miller (2019), interpretability can be defined as "*the degree to which an observer can understand the cause of a decision*". Therefore, we can argue that interpretability is as important as accuracy in automatic decisions as it can show if the model can or cannot be trusted. For example, assume a situation in which a person asks for a credit to his/her bank manager. Moreover, suppose that, after an internal analysis based on a machine learning model, the bank system classifies that person as a possible default and, as a consequence, he/she would not receive the credit. He/she will naturally ask to the bank manager why such a classification was achieved. If the machine is a black box, the manager would not be able to explain such a classification and, therefore, the client may not trust the algorithm. In this situation, a local interpretation would be suitable to understand how each characteristic (e.g., salary, presence or absence of previous default, etc...) contributes towards the default credit classification.

There are practically two main types of interpretability in machine learning: global and local ones (see (Molnar, 2021) for a further discussion on them). The aim of global interpretability methods consists in explaining the trained model as a whole. In other words, one attempts to explaint the average behavior of a trained machine by taking all samples. An example of such a method is the partial dependence plot (Molnar, 2021), whose goal is to provide the marginal effects that each feature has in the predicted outcome. On the other hand, methods for local interpretability attempts to explain, for a specific instance of interest (e.g., a person asking for a credit), how each

attribute's value contributes to achieve the associated prediction or classification. In this paper, we deal with local interpretability. Moreover, we consider a model-agnostic approach, i.e., a method that can be applied to interpret the prediction or classification of any machine learning model.

Among the model-agnostic methods proposed in the literature, two are of interest in this paper: LIME (Local Interpretable Model-agnostic Explanations) (Ribeiro et al., 2016) and SHAP (SHapley Additive exPlanations) (Lundberg & Lee, 2017). In summary, the idea in LIME to explain the prediction of a specific instance of interest consists in, locally, adjusting an interpretable function (e.g., a linear model) based on a set of perturbed samples in the neighborhood of such an instance. When adjusting this linear function, one considers an exponential kernel that ensures that closer are the perturbed samples from the instance of interest, greater are their importance in the learning procedure. Although this function may not be complex enough to explain the model as a whole, it can locally provide a good understanding of the contribution of each attribute towards the model prediction. The other approach, called SHAP, brings concepts from game theory to provide local interpretability. The idea is to explain a prediction by means of the Shapley value (Shapley, 1953) associated with each attribute value. An interesting aspect in such an approach, which leads to the SHAP values[1], is that it satisfies desired properties in interpretability, such as local accuracy, missingness and consistency (Lundberg & Lee, 2017). For that reason, the classical SHAP and its extended versions have been largely used in the literature (Lundberg et al., 2020; Chen et al., 2021; Aas et al., 2021).

Although the Shapley value (as well as the SHAP value in SHAP method) appears as an interesting solution for model-agnostic machine learning interpretability, there is a drawback in its calculation. As it lies on the marginal contribution of each attribute by taking into account all possible coalitions of attributes, the number of evaluations increases exponentially with the number of attributes. Precisely, if we have m attributes, we need 2^m evaluations to calculate the Shapley values. This makes the calculation impracticable in situations where m is large. In order to soften this inconvenience, one may adopt some approaches that approximate the Shapley values, such as the Shapley sampling values strategy (Štrumbelj & Kononenko, 2010, 2014) or the Kernel SHAP (Lundberg & Lee, 2017). We address the latter in this paper.

The Kernel SHAP was also proposed in the original SHAP paper (Lundberg & Lee, 2017). It

[1]In this paper, as the SHAP values are referred to as the Shapley values obtained by means of the SHAP formulation, we will frequently adopt SHAP values or Shapley values interchangeably in the context of machine learning interpretability.

provides a link between LIME and the use of SHAP values for local machine learning interpretability. Although the authors provide this link by assuming an additive function as the interpretable model and a specific kernel, the formulation is not straightforward and there is a lack of details in the proof. With respect to the SHAP values calculation, in order to reduce the computational effort, the authors adopted a clever strategy that selects the evaluations that are most promising to approximate such values. However, this strategy does not reduce the number of evaluations needed for an exact SHAP values calculation. Indeed, although one is only interested in estimating the SHAP values, implicitly, there are 2^m unknown parameters associated with the set of evaluations. Moreover, the authors do not assume further game theory concepts that could reduce such a number and speed up the convergence.

Aiming at providing a straightforward formulation of a Kernel SHAP-based method and speeding up the SHAP values approximation, in this paper, we propose to adopt game theory-based concepts frequently used in multicriteria decision making: the Choquet integral (Choquet, 1954; Grabisch, 1996; Grabisch & Labreuche, 2010) and k-additive games (Grabisch, 1997b). Instead of assuming an additive function as the interpretable model, as a first contribution of this paper, we show that the use of the non-additive function called Choquet integral also leads to the same desired properties for local interpretability. Indeed, we can directly associate the Choquet integral parameters to the Shapley indices, which include both Shapley values and Shapley interaction indices. While the Shapley values indicate the marginal contribution of each attribute, individually, the Shapley interaction indices provide the understanding about how they interact between them (positively or negatively). This is of interest in ML interpretability as it indicates if the simultaneous presence of two characteristics has a higher (or lower) contribution than both of them separately. It is worth mentioning that Lundberg et al. (2020) also discuss how the SHAP method could be adapted to find the Shapley interaction indices. However, in our proposal, they are obtained automatically.

Besides the aforementioned formulation, we can also assume some degree of additivity about the Choquet integral which contributes to reduce its number of parameters. In this context, as a second contribution, we propose to adopt a k-additive Choquet integral. The use of k-additive models (such as 2-additive or 3-additive ones) significantly reduces the number of parameters and has proved flexible enough to achieve good results in terms of generalization (Grabisch et al., 2002, 2006; Pelegrina et al., 2020). For instance, the number of parameters in the 2-additive and the 3-additive models are $m(m+1)/2$ and $m(m^2+5)/6$, respectively. Therefore, by reducing the number of parameters in our proposal, we avoid over-parametrization, which can be the case in Kernel

SHAP as, implicitly, there are 2^m unknown parameters. As attested by numerical experiments, our proposal requires a lower number of evaluations (and, consequently, a lower computational time) to approximate the SHAP values.

The rest of this paper is organized as follows. Section 2 contains the theoretical aspects of Shapley values and the adopted Choquet integral. We also provide a description of LIME and SHAP as model-agnostic methods for local interpretability. In Section 3, we present our Choquet integral-based formulation that leads to the Shapley values and interaction indices, and how the concept of k-additive games can be used to reduce the computational effort when estimating the SHAP values. Thereafter, in Section 4, we conduct some numerical experiments in order to attest our proposal. Finally, in Section 5, we present our concluding remarks and discuss future perspectives.

2. Background

In this section, we present some theoretical aspects that will be used in this work. We start by some concepts frequently used in game theory and multicriteria decision making. Thereafter, we discuss both LIME and SHAP as well as the SHAP values approximation strategy used in Kernel SHAP.

It is worth recalling that, in this paper, we deal with local interpretability. Therefore, we consider a classical machine learning scenario where a model $f(\cdot)$ (e.g., a black box model) has been trained based on a set of n_{tr} training data $(\mathbf{X}, \mathbf{y})$, where $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_{n_{tr}}]$ and $\mathbf{y} = [y_1, \ldots, y_{n_{tr}}]$ represent the inputs and the outputs (e.g., predicted values or a classes), respectively. Our aim consists in explaining the predicted outcome $f(\mathbf{x}^*)$ of the instance of interest $\mathbf{x}^* = [x_1^*, \ldots, x_m^*]$, where m is the number of attributes. Therefore, how $f(\cdot)$ was trained is not important in this paper. We only consider that we are able to use the model $f(\cdot)$ in order to predict the outcome of any instance.

2.1. Shapley values and k-additive games

In cooperative game theory, a coalitional game is defined by a set $M = \{1, 2, \ldots, m\}$ of m players and a function $v : \mathcal{P}(M) \to \mathbb{R}$, where $\mathcal{P}(M)$ is the power set of M, that maps subsets of players to real numbers. For a coalition of players A, $v(A)$ represents the payoff that this coalition can obtain by cooperation. By definition, one assumes $v(\emptyset) = 0$, i.e., there is no payoff when there is no coalition.

The Shapley value (or Shapley power index) is a well-known solution concept in cooperative game theory (Shapley, 1953). In summary, the Shapley value of a player j indicates its (positive or

negative) marginal contribution in the game payoff when taking into account all possible coalitions of players in M. It is defined as follows:

$$\phi_j = \sum_{A \subseteq M \setminus \{j\}} \frac{(m - |A| - 1)!\,|A|!}{m!} \left[\upsilon(A \cup \{j\}) - \upsilon(A)\right], \tag{1}$$

where $|A|$ represents the cardinality of subset A. An interesting property of the Shapley value (called efficiency, which will be further discussed in this paper) is that $\sum_{j=1}^{m} \phi_j = \upsilon(M) - \upsilon(\emptyset)$. For this reason, the Shapley value is a convenient way of sharing the payoff of the grand coalition between the players.

Similarly as in Equation (1), one may also measure the marginal effect of a coalition $\{j, j'\}$ in the payoffs. In this case, one obtains the Shapley interaction index (Murofushi & Soneda, 1993; Grabisch, 1997a), which is defined by

$$I_{j,j'} = \sum_{A \subseteq M \setminus \{j,j'\}} \frac{(m - |A| - 2)!\,|A|!}{(m-1)!} \left[\upsilon(A \cup \{j, j'\}) - \upsilon(A \cup \{j\}) - \upsilon(A \cup \{j'\}) + \upsilon(A)\right] \tag{2}$$

and can be interpreted as the interaction degree of coalition $\{j, j'\}$ by taking into account all possible coalitions of players in M. The sign of $I_{j,j'}$, together with the sign[2] of $\zeta = \upsilon(M) - \upsilon(\emptyset)$, indicate the type of interaction between players j, j':

- If $\zeta I_{j,j'} < 0$, there is a negative interaction (also called redundant effect) between players j, j'.
- If $\zeta I_{j,j'} > 0$, there is a positive interaction (also called complementary effect) between players j, j'.
- If $I_{j,j'} = 0$, there is no interaction between players j, j'.

Besides ϕ_j and $I_{j,j'}$, one may also define the interaction index for any $A \subseteq M$. In this case, the (generalized) interaction index is defined by (Grabisch, 1997a)

$$I(A) = \sum_{D \subseteq M \setminus A} \frac{(m - |D| - |A|)!\,|D|!}{(m - |A| + 1)!} \left(\sum_{D' \subseteq A} (-1)^{|A| - |D'|}\, \upsilon(D \cup D') \right). \tag{3}$$

However, one does not have a clear interpretation as for ϕ_j and $I_{j,j'}$.

[2]Note that the use of ζ in machine learning local interpretability is important to properly interpret the interaction indices. For instance, if $\upsilon(M) > \upsilon(\emptyset)$, $I_{j,j'} > 0$ indicates a positive interaction between players j, j'. However, if $\upsilon(M) < \upsilon(\emptyset)$, there exists a positive interaction between players j, j' if $I_{j,j'} < 0$.

It is important to remark that, given the interaction indices $I(A)$, one may recover the payoffs $\upsilon(A)$ through the linear transformation

$$\upsilon(A) = \sum_{D \subseteq M} \gamma_{|A \cap D|}^{|D|} I(D), \tag{4}$$

where $\gamma_{|A \cap D|}^{|D|}$ is defined by

$$\gamma_r^{r'} = \sum_{l=0}^{r} \binom{r}{l} \eta_{r'-l}, \tag{5}$$

with

$$\eta_r = -\sum_{r'=0}^{r-1} \frac{\eta_{r'}}{r - r' + 1} \binom{r}{r'} \tag{6}$$

being the Bernoulli numbers and $\eta_0 = 1$. As the relation between the game and the interaction indices is linear, it is common to represent the aforementioned transformations using matrix notation. Assume, for instance, that the vectors $\upsilon = [\upsilon(\emptyset), \upsilon(\{1\}), \ldots, \upsilon(\{m\}), \upsilon(\{1,2\}), \ldots, \upsilon(\{m-1,m\}), \ldots, \upsilon(\{1,\ldots,m\})]$ and $\mathbf{I} = [I(\emptyset), \phi_1, \ldots, \phi_m, I_{1,2}, \ldots, I_{m-1,m}, \ldots, I(\{1,\ldots,m\})]$ are represented in a cardinal-lexicographic order (i.e., the elements are sorted according to their cardinality and, for each cardinality, based on the lexicographic order). The transformation from the interaction indices to υ can be represented by $\upsilon = \mathbf{TI}$, where $\mathbf{T} \in \mathbb{R}^{2^m \times 2^m}$ is the transformation matrix. For example, in a game with 3 players, we have

$$\mathbf{T} = \begin{bmatrix} 1 & -1/2 & -1/2 & -1/2 & 1/6 & 1/6 & 1/6 & 0 \\ 1 & 1/2 & -1/2 & -1/2 & -1/3 & -1/3 & 1/6 & 1/6 \\ 1 & -1/2 & 1/2 & -1/2 & -1/3 & 1/6 & -1/3 & 1/6 \\ 1 & -1/2 & -1/2 & 1/2 & 1/6 & -1/3 & -1/3 & 1/6 \\ 1 & 1/2 & 1/2 & -1/2 & 1/6 & -1/3 & -1/3 & -1/6 \\ 1 & 1/2 & -1/2 & 1/2 & -1/3 & 1/6 & -1/3 & -1/6 \\ 1 & -1/2 & 1/2 & 1/2 & -1/3 & -1/3 & 1/6 & -1/6 \\ 1 & 1/2 & 1/2 & 1/2 & 1/6 & 1/6 & 1/6 & 0 \end{bmatrix}.$$

Another concept in game theory directly associated with the interaction indices is the k-additive games. We say that a game is k-additive if $I(A) = 0$ for all A such that $|A| > k$. As it will be further detailed in the next section, an advantage of such games is that one reduces the number of parameters to be defined. In the example with 3 players, for instance, if one assumes a 2-additive game, the last column of $\mathbf{T}$ can be removed since $I(1,2,3) = 0$.

2.2. The Choquet integral

The (discrete) Choquet integral (Choquet, 1954) is a non-additive (more precisely, a piecewise linear) aggregation function that models interactions among attributes. It is defined on a set of parameters associated with all possible coalitions of attributes. It has been largely used in multicriteria decision making problems (Grabisch, 1996; Grabisch & Labreuche, 2010) and, in such situations, the parameters associated with the Choquet integral are called capacity coefficients. A capacity is a set function $\mu : 2^M \rightarrow \mathbb{R}_+$ satisfying the axioms of normalization ($\mu(\emptyset) = 0$ and $\mu(M) = 1$) and monotonicity (if $A \subseteq D \subseteq M$, $\mu(A) \leq \mu(D) \leq \mu(M)$). However, the Choquet integral is not restricted to capacities (Grabisch, 2016). Indeed, it can be defined by means of a game $v : 2^M \rightarrow \mathbb{R}$ satisfying $v(\emptyset) = 0$. The Choquet integral definition based on a game v is given as follows:

$$f_{CI}(\mathbf{x}) = \sum_{j=1}^{m} (x_{(j)} - x_{(j-1)}) v(\{(j), \ldots, (m)\}), \tag{7}$$

where $\cdot_{(j)}$ indicates a permutation of the indices j such that $0 \leq x_{(1)} \leq x_{(j)} \leq \ldots \leq x_{(m)} \leq 1$ (with $x_{(0)} = 0$).

As the Choquet integral is defined by means of a game, one may define it in terms of Shapley values and interaction indices. Therefore, one has a clear interpretation about the marginal contribution of each feature in the aggregation procedure as well as the interaction degree between them. For instance, if two attributes have a positive (resp. negative) interaction, the payoff of such a coalition is (resp. is not) greater than the sum of its individual payoffs. Moreover, one may also consider the case of a k-additive game and, therefore, a k-additive Choquet integral (Grabisch, 1997b). For example, if one assumes a 2-additive game, (7) can be formulated as follows:

$$f_{CI}(\mathbf{x}) = \sum_{j} x_j \left(\phi_j - \frac{1}{2} \sum_{j'} \left| I_{j,j'} \right| \right) + \sum_{I_{j,j'} < 0} (x_j \vee x_{j'}) \left| I_{j,j'} \right| + \sum_{I_{j,j'} > 0} (x_j \wedge x_{j'}) I_{j,j'}, \tag{8}$$

where $\vee$ and $\wedge$ represent the maximum and the minimum operators, respectively. Note that, when learning the Choquet integral parameters, if one assumes a 2-additive model, one reduces the number of parameters from 2^m to $m(m+1)/2$. Therefore, 2-additive and, more generally, k-additive models emerge as a strategy that reduces the computational complexity in optimization tasks and provides a more interpretable model (since one has less parameters to interpret). Moreover, it is also known from multicriteria decision making applications (Grabisch et al., 2002, 2006; Pelegrina et al., 2020) that, even if one adopts a 2-additive model, the Choquet integral is still being flexible enough to

model inter-attributes relations and can achieve a high level of generalization. Surely, greater the k, greater the flexibility to model inter-attributes relations.

It is important to remark that, if one assumes a 1-additive game, the Choquet integral boils down to the weighted arithmetic mean.

2.3. Model-agnostic methods for local interpretability

We describe in this section two famous model-agnostic methods for local interpretability: LIME and SHAP. At first, we briefly present the idea behind tabular LIME (i.e., LIME for tabular data). Then, we further discuss the SHAP method, specially the Kernel SHAP strategy. It is worth mentioning that, differently from (Ribeiro et al., 2016; Lundberg & Lee, 2017), we here adopt a notation based on set theory in order to clearly define the elements used in the considered approaches.

2.3.1. LIME

The main idea of LIME (Ribeiro et al., 2016) for local explanations is to locally approximate a (generally) complex function $f(\cdot)$ (frequently obtained by a black box model) by an interpretable model $g(\cdot)$. For this purpose, in order to explain the outcome $f(\mathbf{x}^*)$ of an instance $\mathbf{x}^*$, one firstly generates a set of q perturbed samples $\mathbf{z}_l$, $l = 1, \ldots, q$, in the neighborhood of $\mathbf{x}^*$. For each sample $\mathbf{z}_l$, one also defines a binary vector $\mathbf{z}'_l$ such that $z'_{l,j} = 1$ if $z_{l,j}$ is close enough[3] to x^*_j, or $z'_{l,j} = 0$ otherwise. Once all samples have been generated, LIME deals with the following optimization problem:

$$\min_{g \in G} \mathcal{L}(f, g, \pi_{\mathbf{x}^*}) + \Omega(g), \tag{9}$$

where $\mathcal{L}(f, g, \pi_{\mathbf{x}^*})$ is the loss function, $\pi_{\mathbf{x}^*}$ is a proximity measure between the instance to be explained and the perturbed samples and $\Omega(g)$ is a measure of complexity of the interpretable model $g(\cdot)$. In tabular LIME, the authors used the exponential kernel for the proximity measure, which leads to the expression

$$\pi_{\mathbf{x}^*}(\mathbf{z}'_l) = \exp\left(\frac{-\|\mathbb{1} - \mathbf{z}'_l\|^2}{\alpha^2}\right), \tag{10}$$

where $\|\cdot\|$ is the Euclidean norm, $\mathbb{1}$ is a vector of 1's and α is a positive bandwidth parameter (as default, the authors assumed $\alpha = \sqrt{0.75m}$). By assuming a weighted least squared function for

[3] In order to define how close $z_{l,j}$ is from x^*_j, for each attribute, LIME equally splits the training data (by taking the quantiles of the training data) into predefined bins. If $z_{l,j}$ is on the same bin as x^*_j, $z'_{l,j} = 1$, or $z'_{l,j} = 0$ otherwise. For further details about this procedure, the interested reader may refer to (Garreau & von Luxburg, 2020).

$\mathcal{L}(f, g, \pi_{\mathbf{x}^*})$, a linear function $g(\mathbf{z}') = \beta_0 + \beta^T \mathbf{z}'$ (where $\beta = (\beta_1, \ldots, \beta_m)$) and letting $\Omega(\beta) = \lambda \|\beta\|^2$ represent a regularization term with $\lambda > 0$, LIME can be formulated as follows:

$$\min_{\beta_0, \beta_1, \ldots, \beta_m} \sum_{l=1}^{q} \pi_{\mathbf{x}^*}(\mathbf{z}'_l) \left(f(\mathbf{z}_l) - \left(\beta_0 + \beta^T \mathbf{z}'_l \right) \right)^2 + \lambda \|\beta\|^2. \tag{11}$$

After solving (11), one can visualize the obtained parameters β and, therefore, interpret the (positive or negative) contribution of each attribute in the predicted outcome in the vicinity of $\mathbf{x}^*$.

2.3.2. SHAP

Differently from LIME, the purpose of SHAP is to use the Shapley values in order to locally explain a prediction. The idea is to associate to each attribute its marginal contribution in the predicted outcome. In this section, we present a summary of the idea behind SHAP. Moreover, we discuss the Kernel SHAP, which is a kernel-based approach for approximating the SHAP values by using the LIME formulation. For further details, the interested reader may refer to (Lundberg & Lee, 2017; Lundberg et al., 2018, 2020; Aas et al., 2021).

The idea that brings Shapley values into interpretability methods in machine learning consists in associating players and payoffs in game theory to attributes and values of a subset of attributes in the model prediction, respectively. Before presenting the idea behind SHAP, let us define the characteristic vector of A. Recall that $M = \{1, \ldots, m\}$ represents the set of m attributes. For any $A \subseteq M$, $\mathbf{1}_A \in \{0, 1\}^m$ denotes the characteristic vector of A, i.e., a binary vector such that the j-th coordinate is 1, if $j \in A$, and 0, otherwise. For example, for $M = \{1, 2, 3\}$, $\mathbf{1}_{\{2,3\}} = [0, 1, 1]$ means a coalition of attributes $\{2, 3\}$.

Based on the aforementioned definition, in order to explain the predicted outcome $f(\mathbf{x}^*)$ of an instance $\mathbf{x}^*$, the authors decompose $f(\mathbf{x}^*)$ by assuming the additive feature attribution function given by

$$f(\mathbf{x}^*) = g(\mathbf{1}_M) = \phi_0 + \sum_{j \in M} \phi_j. \tag{12}$$

Moreover, they argue that the only possible explanation model $g(\cdot)$ that follows Equation (12) and satisfies the local accuracy, missingness and consistency properties (see Appendix A for the definitions) consists in defining $\phi_0 = \mathbb{E}[f(\mathbf{x})]$, i.e., the (overall) expected prediction when one does not know any attribute value from $\mathbf{x}^*$, and the (exact) SHAP values ϕ_j, $j = 1, \ldots, m$, given by

$$\phi_j(f, \mathbf{x}^*) = \sum_{A \subseteq M \setminus \{j\}} \frac{(m - |A| - 1)!\, |A|!}{m!} \left[\hat{f}_{\mathbf{x}^*}(A \cup \{j\}) - \hat{f}_{\mathbf{x}^*}(A) \right], \tag{13}$$

where $\hat{f}_{\mathbf{x}^*}(A)$ is the expected model prediction given the knowledge on the attributes values of $\mathbf{x}^*$ that are present in coalition A, that is:

$$\hat{f}_{\mathbf{x}^*}(A) = \mathbb{E}\left[f\left(\mathbf{x}\right)|x_j = x_j^* \; \forall \; j \in A\right]. \tag{14}$$

Note in Equation (14) that one has missing values for all attributes $j' \in \overline{A}$, where $\overline{A}$ is the complement set of A (if $A = M$, then $\hat{f}_{\mathbf{x}^*}(M) = \mathbb{E}\left[f\left(\mathbf{x}^*\right)\right] = f\left(\mathbf{x}^*\right)$ and there are no missing values). In this case, in order to calculate the expected prediction, one randomly samples these missing values from the training data. In this paper, as well as in the Kernel SHAP method, we assume independence among attributes. Therefore, the expected prediction can be calculated as follows:

$$\hat{f}_{\mathbf{x}^*}(A) = \frac{1}{q}\sum_{l=1}^{q} f\left(\mathbf{x}_A^*, \mathbf{x}_{l,\overline{A}}\right), \tag{15}$$

where $\mathbf{x}_{l,\overline{A}}$, $l = 1, \ldots, q$, are samples from the training data. Note that, in comparison with the game theory formulation presented in Equation (1), $\hat{f}_{\mathbf{x}^*}(A)$ represents the payoff $\upsilon(A)$. Moreover, when all attributes are missing, i.e., $A = \emptyset$, one has $\hat{f}_{\mathbf{x}^*}(\emptyset) = \mathbb{E}\left[f\left(\mathbf{x}\right)\right] = \phi_0$.

Among the properties satisfied by the SHAP values, the local accuracy plays an important role in local interpretability and differentiate SHAP from the original LIME formulation (as presented in Section 2.3.1). It states that one can decompose the predicted outcome $f(\mathbf{x}^*)$ by the sum of the SHAP values and the overall expected prediction ϕ_0, i.e., $f(\mathbf{x}^*) = \phi_0 + \sum_{j=1}^{m}\phi_j$. Therefore, one may interpret the SHAP values as the contribution of each attribute when one moves from the overall expected prediction when all attributes are missing to the actual outcome $f(\mathbf{x}^*)$.

2.3.3. Kernel SHAP

An important remark in the exact SHAP values calculation is that one needs to sample all 2^m possible coalitions of attributes and calculate its expected model prediction. Therefore, this procedure may be computationally heavy for a large number of attributes. In order to overcome this inconvenience, the authors proposed a SHAP value-based formulation called Kernel SHAP (Lundberg & Lee, 2017). Kernel SHAP emerges as the formulation of LIME method that leads to the SHAP values. For instance, the authors claimed that if one assumes

- $\Omega(g) = 0$,
- $\pi(A) = \frac{(m-1)}{\binom{m}{|A|}|A|(m-|A|)}$,
- $\mathcal{L}(f, g, \pi) = \sum_{A\in\mathcal{M}}\pi(A)\left(\hat{f}_{\mathbf{x}^*}(A) - g(\mathbf{1}_A)\right)^2$, where $g(\mathbf{1}_A) = \phi_0 + \sum_{j\in A}\phi_j$ and $\mathcal{M} \subseteq \mathcal{P}(M)$ (recall that $\mathcal{P}(M)$ is the power set of M),

the solution of the weighted least square problem

$$\min_{\phi_0,\phi_1,\ldots,\phi_m} \sum_{A\in\mathcal{M}} \frac{(m-1)}{\binom{m}{|A|}\,|A|\,(m-|A|)} \left(\hat{f}_{\mathbf{x}^*}(A) - \left(\phi_0 + \sum_{j\in A}\phi_j\right)\right)^2 \tag{16}$$

leads to the SHAP values. Note that, differently from the LIME formulation, $\pi(A)$ in Kernel SHAP only depends on coalition A. Moreover, $\pi(A)$ tends to infinity when $A = M$. Therefore, in the optimal solution, $\hat{f}_{\mathbf{x}^*}(M) = f(\mathbf{x}^*) = g(\mathbf{1}_M) = \phi_0 + \sum_{j=1}^{m}\phi_j$. This ensures that $f(\mathbf{x}^*)$ is explained by the sum of the SHAP values and the overall expected prediction $\mathbb{E}[f(\mathbf{x})]$. Similarly, when $A = \emptyset$, the associated $\pi(\emptyset)$ also tends to infinity. This ensures that $\hat{f}_{\mathbf{x}^*}(\emptyset) = \mathbb{E}[f(\mathbf{x})] = g(\mathbf{1}_\emptyset) = \phi_0$. In practice, we replace these infinite values by a big constant (e.g., 10^6).

As (16) is a weighted least square problem, one may easily represent it (as well as its solution) by means of matrices and vectors (we borrow such a formulation from (Aas et al., 2021)). Suppose that $n_{\mathcal{M}}$ represents the number of elements in $\mathcal{M}$ (i.e., the number of coalitions considered in the optimization problem (16)). Let us also define $\phi = [\phi_0, \phi_1, \ldots, \phi_m]$ and $\mathbf{Z} \in \{0,1\}^{n_{\mathcal{M}}\times(m+1)}$ as the matrix such that the first column is 1 for every row and the remaining $m+1$ columns are composed, in each row, by all $\mathbf{1}_A$, $A \in \mathcal{M}$. Moreover, assume that $\mathbf{f} \in \mathbb{R}^{n_{\mathcal{M}}\times 1}$ and $\mathbf{W} \in \mathbb{R}^{n_{\mathcal{M}}\times n_{\mathcal{M}}}$ are the vector of evaluations $\hat{f}_{\mathbf{x}^*}(A)$ and the diagonal matrix whose elements are given by $\pi(A)$, respectively, associated with all $A \in \mathcal{M}$. For example, in a problem with 3 attributes ($M = \{1,2,3\}$) and using $\emptyset$, $\{1\}$, $\{2\}$, $\{1,3\}$ and M as the coalitions of attributes, we have the following:

$$\phi = \begin{bmatrix}\phi_0\\ \phi_1\\ \phi_2\\ \phi_3\end{bmatrix},\ \mathbf{Z} = \begin{bmatrix}1&0&0&0\\ 1&1&0&0\\ 1&0&1&0\\ 1&1&0&1\\ 1&1&1&1\end{bmatrix},\ \mathbf{f} = \begin{bmatrix}\hat{f}_{\mathbf{x}^*}(\emptyset)\\ \hat{f}_{\mathbf{x}^*}(\{1\})\\ \hat{f}_{\mathbf{x}^*}(\{2\})\\ \hat{f}_{\mathbf{x}^*}(\{1,3\})\\ \hat{f}_{\mathbf{x}^*}(M)\end{bmatrix} \text{ and } \mathbf{W} = \begin{bmatrix}10^6&0&0&0&0\\ 0&\pi(\{1\})&0&0&0\\ 0&0&\pi(\{2\})&0&0\\ 0&0&0&\pi(\{1,3\})&0\\ 0&0&0&0&10^6\end{bmatrix}.$$

Based on the vector/matrix notation, one may represent the optimization problem (16) as

$$\min_{\phi} (\mathbf{f} - \mathbf{Z}\phi)^T \mathbf{W} (\mathbf{f} - \mathbf{Z}\phi), \tag{17}$$

whose solution is given by

$$\phi = \left(\mathbf{Z}^T\mathbf{W}\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{W}\mathbf{f}. \tag{18}$$

Remark that $\mathbf{S} = \left(\mathbf{Z}^T\mathbf{W}\mathbf{Z}\right)^{-1}\mathbf{Z}^T\mathbf{W}$ can be calculated independently of the instance of interest $\mathbf{x}^*$. Therefore, an interesting aspect in Kernel SHAP is that, even if one would like to explain the outcome of several instances of interest, one only needs to calculate $\mathbf{S}$ once. The only element that

varies in Equation (18) is the vector of evaluations $\mathbf{f}$, which is dependent on the instance of interest under analysis.

Another remark in Kernel SHAP is that, if $\mathcal{M} = \mathcal{P}(M)$, Equation (18) leads to the exact SHAP values (as in Equation (13)). Therefore, in this exact calculation, one needs the expected predictions $\hat{f}_{\mathbf{x}^*}(A)$ for all possible 2^m coalitions A, which can be infeasible for a large number of attributes. However, the clever strategy used in Kernel SHAP aims at selecting the most promising expected predictions to approximate the SHAP values. For instance, if one considers the weighting kernel $\pi(A)$, one may note that the majority of A has a low contribution in the SHAP value calculation. Therefore, the aim in Kernel SHAP consists in defining a subset $\mathcal{M}$ from $\mathcal{P}(M)$ such that the elements $A \in \mathcal{M}$ are sampled[4] from a probability distribution following the weighting kernel $\pi(A)$. Greater is the weight associated with A, greater is the chance that A is sampled from $\mathcal{P}(M)$.

It is important to note that, although the Kernel SHAP approximates the SHAP values with less coalitions, the relation between the game and the generalized interaction indices (which include the Shapley values) involves 2^m parameters. Therefore, even if one is interested in estimating m parameters (the SHAP values), implicitly, we are dealing with the whole power set of M. We address such an issue in our proposed approach.

3. A more general model for local interpretability based on Shapley values

As highlighted in Section 1, we have two main contributions in this paper: to provide a straightforward formulation of the Kernel SHAP method based on the Choquet integral, and to adopt the concept of k-additive games in order to reduce the number of evaluations needed to approximate the SHAP values. Both contributions are presented in the sequel.

3.1. The Choquet integral as an interpretable model for Kernel SHAP formulation

We here show that we need not consider an additive function as the interpretable model in order to explain a prediction based on the Shapley values. Indeed, if we adopt the non-additive function called Choquet integral, we also achieve such values. Recall the Choquet integral function defined in Equation (7). The idea is to define the local interpretable model $g(\cdot)$ as

$$g(\mathbf{1}_A) = \phi_0 + f_{CI}(\mathbf{1}_A), \tag{19}$$

[4]In order to avoid double selecting the same A, in the experiments conducted in this paper, we adopted a sampling procedure without replacement. Therefore, after sampling a coalition, we update the probability distribution by removing the associated kernel weight and normalizing the probabilities.

where ϕ_0 is the intercept parameter. In order to simplify the notation, let us also define $\bar{f}_{\mathbf{x}^*}(A) = \hat{f}_{\mathbf{x}^*}(A) - \phi_0$. In this case and based on the LIME formulation for local interpretability, one obtains the following loss function:

$$\mathcal{L}(f, g, \pi) = \sum_{A \in \mathcal{M}} \pi'(A) \left(\bar{f}_{\mathbf{x}^*}(A) - f_{CI}(\mathbf{1}_A)\right)^2, \tag{20}$$

where the weights $\pi'(A)$ have the same values for all A (e.g., 1) except for the empty set and the grand coalition M, whose associated weights are big numbers (e.g., 10^6). We clarify these choices soon.

An interesting aspect on the Choquet integral and that can be easily checked from Equation (7) is that, when we only have binary data (which is our case since $\mathbf{1}_A$ is a binary vector), $f_{CI}(\mathbf{1}_A) = \upsilon(A)$. Therefore, we may redefine the loss function presented in (20) as

$$\mathcal{L}(f, g, \pi) = \sum_{A \in \mathcal{M}} \pi'(A) \left(\bar{f}_{\mathbf{x}^*}(A) - \upsilon(A)\right)^2. \tag{21}$$

Remark that, for $A = \emptyset$, we minimize $\bar{f}_{\mathbf{x}^*}(\emptyset) - \upsilon(\emptyset) = \hat{f}_{\mathbf{x}^*}(\emptyset) - \phi_0 - \upsilon(\emptyset) = \upsilon(\emptyset)$, since $\hat{f}_{\mathbf{x}^*}(\emptyset) = \phi_0$ by definition. In order to ensure that $\upsilon(\emptyset) = 0$ (according to the definition of a game), we assume a big number for $\pi'(\emptyset)$ when solving the optimization problem. Similarly, when $A = M$, we minimize the difference between $\hat{f}_{\mathbf{x}^*}(M)$ and $\phi_0 + \upsilon(M)$. In this case, as $\upsilon(M) = \upsilon(\emptyset) + \sum_{j=1}^m \phi_j$, the big weight $\pi'(M)$ ensures that $\phi_0 + \sum_{j=1}^m \phi_j = \hat{f}_{\mathbf{x}^*}(M) = f(\mathbf{x}^*)$ (the local accuracy property).

Furthermore, if one considers the linear transformation presented in Equation (4), the loss function can be directly defined in terms of the generalized Shapley interaction indices. In this case, we have the following optimization problem:

$$\min_{\mathbf{I}} \sum_{A \in \mathcal{M}} \pi'(A) \left(\bar{f}_{\mathbf{x}^*}(A) - \sum_{D \subseteq M} \gamma_{|A \cap D|}^{|D|} I(D)\right)^2, \tag{22}$$

where γ is defined as in Equation (5). As in the Kernel SHAP, our proposal also leads to the exact SHAP values if $\mathcal{M} = \mathcal{P}(M)$. We prove it in the sequel.

Theorem 1. *If $\mathcal{M} = \mathcal{P}(M)$, the solution of* (22) *leads to the exact SHAP values as calculated in Equation* (13)*.*

Proof. Assume $\mathcal{M} = \mathcal{P}(M)$. In this scenario, the optimization problem (22) has a unique solution such that $\sum_{D \subseteq M} \gamma_{|A \cap D|}^{|D|} I(D) = \upsilon(A) = \bar{f}_{\mathbf{x}^*}(A)$. From the obtained game and the linear transformation presented in Equation (1), we have that $\phi_j = \sum_{A \subseteq M \setminus \{j\}} \frac{(m-|A|-1)!|A|!}{m!} [\upsilon(A \cup \{j\}) - \upsilon(A)]$. It remains to show that $\phi_j \equiv \phi_j(f, \mathbf{x}^*)$.

Recall that we defined $\bar{f}_{\mathbf{x}^*}(A) = \hat{f}_{\mathbf{x}^*}(A) - \phi_0$ and, then, $\upsilon(A) = \hat{f}_{\mathbf{x}^*}(A) - \phi_0$ in the optimal solution. Therefore, we have the following:

$$\begin{aligned}
\phi_j &= \sum_{A \subseteq M \setminus \{j\}} \frac{(m - |A| - 1)!\,|A|!}{m!} \left[\hat{f}_{\mathbf{x}^*}(A \cup \{j\}) - \phi_0 - \hat{f}_{\mathbf{x}^*}(A) + \phi_0\right] \\
&= \sum_{A \subseteq M \setminus \{j\}} \frac{(m - |A| - 1)!\,|A|!}{m!} \left[\hat{f}_{\mathbf{x}^*}(A \cup \{j\}) - \hat{f}_{\mathbf{x}^*}(A)\right] \qquad (23) \\
&= \phi_j(f, \mathbf{x}^*),
\end{aligned}$$

which proves that our proposal also converges to the exact SHAP values when $\mathcal{M} = \mathcal{P}(M)$. □

Similarly as in the Kernel SHAP, we may here also rewrite the optimization problem in vector/matrix notation. For this purpose, let us represent $\bar{\mathbf{f}} \in \mathbb{R}^{n_\mathcal{M} \times 1}$ as the vector $\mathbf{f}$ (as defined in Section 2.3.3) discounted by ϕ_0 and $\bar{\mathbf{W}} \in \mathbb{R}^{n_\mathcal{M} \times n_\mathcal{M}}$ as the diagonal matrix whose elements are 1's except for the elements associated with the empty set and the grand coalition M, whose weights are a big number (e.g., 10^6). Moreover, we define $\upsilon_\mathcal{M}$ as the vector of payoffs for all coalitions A such that $A \in \mathcal{M}$. In addition, we consider $\mathbf{T}_\mathcal{M}$ as the transformation matrix whose rows are composed by the rows of $\mathbf{T}$ (as defined in Section 2.1) associated with all coalitions A such that $A \in \mathcal{M}$. For example, in the same problem when $M = \{1, 2, 3\}$ and using $\emptyset$, $\{1\}$, $\{2\}$, $\{1, 3\}$ and M as the coalitions of attributes, we have the following:

$$\upsilon_\mathcal{M} = \begin{bmatrix} \upsilon(\emptyset) \\ \upsilon(\{1\}) \\ \upsilon(\{2\}) \\ \upsilon(\{1,3\}) \\ \upsilon(M) \end{bmatrix} = \begin{bmatrix} 1 & -1/2 & -1/2 & -1/2 & 1/6 & 1/6 & 1/6 & 0 \\ 1 & 1/2 & -1/2 & -1/2 & -1/3 & -1/3 & 1/6 & 1/6 \\ 1 & -1/2 & 1/2 & -1/2 & -1/3 & 1/6 & -1/3 & 1/6 \\ 1 & 1/2 & -1/2 & 1/2 & -1/3 & 1/6 & -1/3 & -1/6 \\ 1 & 1/2 & 1/2 & 1/2 & 1/6 & 1/6 & 1/6 & 0 \end{bmatrix} \begin{bmatrix} I(\emptyset) \\ \phi_1 \\ \phi_2 \\ \phi_3 \\ I_{1,2} \\ I_{1,3} \\ I_{2,3} \\ I(\{1,2,3\}) \end{bmatrix} = \mathbf{T}_\mathcal{M}\mathbf{I},$$

$$\bar{\mathbf{f}} = \begin{bmatrix} \hat{f}_{\mathbf{x}^*}(\emptyset) - \phi_0 \\ \hat{f}_{\mathbf{x}^*}(\{1\}) - \phi_0 \\ \hat{f}_{\mathbf{x}^*}(\{2\}) - \phi_0 \\ \hat{f}_{\mathbf{x}^*}(\{1,3\}) - \phi_0 \\ \hat{f}_{\mathbf{x}^*}(M) - \phi_0 \end{bmatrix} \text{ and } \bar{\mathbf{W}} = \begin{bmatrix} 10^6 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 10^6 \end{bmatrix}.$$

The vector/matrix notation leads to the following optimization problem:

$$\min_{\mathbf{I}} \left(\bar{\mathbf{f}} - \mathbf{T}_{\mathcal{M}}\mathbf{I}\right)^T \bar{\mathbf{W}} \left(\bar{\mathbf{f}} - \mathbf{T}_{\mathcal{M}}\mathbf{I}\right), \tag{24}$$

whose solution is given by

$$\mathbf{I} = \left(\mathbf{T}_{\mathcal{M}}^T \bar{\mathbf{W}} \mathbf{T}_{\mathcal{M}}\right)^{-1} \mathbf{T}_{\mathcal{M}}^T \bar{\mathbf{W}} \bar{\mathbf{f}}. \tag{25}$$

It is important to note that, differently from the Kernel SHAP formulation discussed in Section 2.3.3, in our proposal we obtain all Shapley interaction indices (which, obviously, include the SHAP values). Therefore, this can also be infeasible for a large number of attributes as the number of parameters is given by 2^m. However, one may exploit some degree of additivity about the Choquet integral which contributes to reduce its number of parameters. We discuss this aspect in the next section.

3.2. k-additive games for local interpretability

As a second contribution, we propose to adopt the concept of k-additive games in the Choquet integral-based formulation for local interpretability. Called here k_{ADD}-SHAP, our proposal consists in dealing with the following weighted least square problem:

$$\min_{\mathbf{I}_k} \sum_{A \in \mathcal{M}} \pi'(A) \left(\bar{f}_{\mathbf{x}^*}(A) - \sum_{\substack{D \subseteq M, \\ |D| \leq k}} \gamma_{|A \cap D|}^{|D|} I(D) \right)^2, \tag{26}$$

where $\mathbf{I}_k = [I(\emptyset), \phi_1, \ldots, \phi_m, I_{1,2}, \ldots, I(\{m-k, \ldots, m\}]$ is the vector of Shapley interaction indices, in a cardinal-lexicographic order, for all $I(D)$ such that $|D| \leq k$. By using the vector/matrix notation, we may rewrite (26) as follows:

$$\min_{\mathbf{I}_k} \left(\bar{\mathbf{f}} - \mathbf{T}_{\mathcal{M},k}\mathbf{I}_k\right)^T \bar{\mathbf{W}} \left(\bar{\mathbf{f}} - \mathbf{T}_{\mathcal{M},k}\mathbf{I}_k\right), \tag{27}$$

whose solution is given by

$$\mathbf{I}_k = \left(\mathbf{T}_{\mathcal{M},k}^T \bar{\mathbf{W}} \mathbf{T}_{\mathcal{M},k}\right)^{-1} \mathbf{T}_{\mathcal{M},k}^T \bar{\mathbf{W}} \bar{\mathbf{f}}, \tag{28}$$

where $\mathbf{T}_{\mathcal{M},k}$ is equal to $\mathbf{T}_{\mathcal{M}}$ up to the columns associated with all $I(D')$ such that $|D'| \leq k$ ($I(D') = 0$ for all coalitions D' such that $|D'| > k$). Therefore, in this formulation and for $k \geq 2$, one eliminates unnecessary parameters ($I(A)$ when $|A|$ is large) in terms of local interpretability.

Note that, as in (26) (or (27)) we restrict the feasible domain to the Shapley indices whose cardinalities are at most k, we can not guarantee to achieve the exact SHAP values even if $\mathcal{M} = \mathcal{P}(M)$.

In other words, Theorem 1 is not valid for the proposed k_{ADD}-SHAP. However, as already mentioned in Section 2.2, an advantage of such a model is that one both drastically reduces the number of parameters to be determined while still having a flexible model to generalize the relation between inputs and outputs. Therefore, in order to approximate the exact SHAP values, we avoid over-parametrization and we may need less evaluations when adopting (27) compared to (17). It is important to note that, even if in the Kernel SHAP formulation one only searches for the Shapley values (i.e., m parameters), implicitly, such parameters as well as the remaining (generalized) interaction indices are associated with all 2^m expected predicted evaluations defined by all $A \subseteq M$ (recall the linear transformation presented in Equations (3) and (4)). This may bring an over-parametrization when the dataset has several attributes. Moreover, as the number of unknown parameters differs from our proposal and the Kernel SHAP, for the same subset of coalitions, both approaches will probably lead to different results. Also, we expect that our proposal converges faster than the Kernel SHAP as one eliminates parameters that are not important for the purpose of interpretability.

With respect to how to select the subset $\mathcal{M}$ of evaluations, we consider the same strategy as in Kernel SHAP. We sample the elements $A \in \mathcal{M}$ according to the probability distribution defined by $p_A = \frac{\pi(A)}{\sum_{A \subseteq M} \pi(A)}$. As we adopt in this paper a sampling procedure without replacement, after sampling a coalition, we update the probability distribution and normalize it. Moreover, as $p_\emptyset$ and p_M are much greater than the other probabilities, it is very likely that both the empty set and the grand coalition M are sampled to compose the subset $\mathcal{M}$.

Equivalently as in the Kernel SHAP formulation, $\mathbf{S}_{\mathcal{M},k} = \left(\mathbf{T}_{\mathcal{M},k}^T \bar{\mathbf{W}} \mathbf{T}_{\mathcal{M},k}\right)^{-1} \mathbf{T}_{\mathcal{M},k}^T \bar{\mathbf{W}}$ (or $\mathbf{S}_{\mathcal{M}} = \left(\mathbf{T}_{\mathcal{M}}^T \bar{\mathbf{W}} \mathbf{T}_{\mathcal{M}}\right)^{-1} \mathbf{T}_{\mathcal{M}}^T \bar{\mathbf{W}}$) can also be calculated independently of the instance of interest $\mathbf{x}^*$. Therefore, in order to explain the outcome of several instances of interest, one only needs to calculate $\mathbf{S}_{\mathcal{M},k}$ (or $\mathbf{S}_{\mathcal{M}}$) once.

4. Numerical experiments

In this section, we present some numerical experiments in order to check the validity and interest of our model[5]. The experiments are based on four datasets frequently used in the literature: Diabetes (Efron et al., 2004), Red Wine Quality (Cortez et al., 2009), Law School Admission Council Wightman (1998) and Pima Indians Diabetes (Smith et al., 1988). In the sequel, we provide a

[5]All codes can be accessed in `https://github.com/GuilhermePelegrina/k_addSHAP`.

brief description of them:

- Diabetes dataset: This dataset contains $m = 10$ attributes (age, sex, body mass index, average blood pressure and six blood serum measurements) that describe $n = 442$ diabetes patients. All collected data are centralized (with zero mean) and with standard deviation equals to 0.0476. For each patient, one also has as the predicted value a measure of the diabetes progression. The mean and the standard deviation for the diabetes progression measure are 152.13 and 77.00, respectively. In our experiments, we split the dataset into training (80%, i.e., $n_{tr} = 353$ samples) and test (20%, i.e., $n_{te} = 89$ samples).

- Red Wine Quality dataset: In this dataset, one has $m = 11$ attributes describing $n = 1599$ red wines. Both mean and standard deviation (std) of attributes are described in Table 1. For each wine, one also has a score (between 0 and 10) indicating its quality. In our experiments, we use this data for the purpose of classification and, therefore, we assume that a good (resp. a bad) wine has a score greater than 5 (resp. at most 5). In total, one has 855 good wine (class value 1) and 744 bad wine (class value 0). Moreover, we split the dataset into training (80%, i.e., $n_{tr} = 1279$ samples) and test (20%, i.e., $n_{te} = 320$ samples).

- Law School Admission Council (LSAC) dataset: This dataset contains $n = 23726$ candidates described by $m = 11$ features. Both mean and standard deviation (std) of attributes are described in Table 2. The purpose in this dataset is to predict whether a candidate would pass the bar exam. Therefore, it is a classification task.

- Pima Indians Diabetes: Differently from the previous Diabetes dataset, the goal in Pima Indians Diabetes is to diagnostically classify a person as having (or not) diabetes. This dataset contains $n = 768$ patients described by $m = 8$ features, summarized in Table 3.

Besides different datasets, we also evaluate our proposal by assuming two training models: Neural Network and Random Forest[6]. Recall that the purpose of this paper is to address inter-

[6]We borrowed these methods from the Scikit-learn library (Pedregosa et al., 2011) in Python and adopted the following parameters:

- Neural Network: $max_{iter} = 10^6$ for both MLPRegressor and MLPClassifier.
- Random Forest: $n\_estimators = 1000$, $max\_depth = None$ and $min\_samples\_split = 2$ for both RandomForestRegressor and RandomForestClassifier.

Table 1: Summary of the Wine dataset.

Attributes	**Mean (± std)**	**Attributes**	**Mean (± std)**	**Attributes**	**Mean (± std)**
fixed acidity	8.320 (±1.740)	chlorides	0.087 (±0.047)	pH	3.311 (±0.154)
volatile acidity	0.528 (±0.179)	free sulfur dioxide	15.875 (±10.457)	sulphates	0.658 (±0.169)
citric acid	0.271 (±0.195)	total sulfur dioxide	46.468 (±32.885)	alcohol	10.423 (±1.065)
residual sugar	2.539 (±1.409)	density	0.997 (±0.002)		

pretability in any trained machine learning model. We do not work on improving the model itself. So we attempt to explain the contributions of attributes regardless how the model is accurate.

4.1. Experiment varying the number of expected prediction evaluations until the exact SHAP values convergence

In the first experiment, we verify the convergence of the proposed k_{ADD}-SHAP and the Kernel SHAP to the exact SHAP values. For each dataset and test sample (recall that we use the training data to calculate the expected predictions given the coalitions in $\mathcal{M}$), we vary the number of expected prediction evaluations, apply both k_{ADD}-SHAP and Kernel SHAP and calculate the squared error when estimating the exact SHAP values. Let us represent, for a given test sample i', the SHAP values obtained by the Equation (13) (the exact SHAP values), the k_{ADD}-SHAP and the Kernel SHAP as $\phi^{exact,i'}$, $\phi^{k_{ADD},i'}$ and $\phi^{Kernel,i'}$, respectively. The squared error between $\phi^{exact,i'}$ and $\phi^{k_{ADD},i'}$ is given as follows:

$$\varepsilon_{k_{ADD},i'} = \sum_{j=1}^{m} \left(\phi_j^{exact,i'} - \phi_j^{k_{ADD},i'} \right)^2. \tag{29}$$

In order to calculate the squared error with respect to the Kernel SHAP, one only needs to replace $\phi_j^{k_{ADD},i'}$ by $\phi^{Kernel,i'}$. By increasing $n_{\mathcal{M}}$, i.e., the number of coalitions selected to calculate the expected prediction evaluations used in the estimation procedure, the aim is to verify the convergence to the exact SHAP values. We show the obtained results by taking the median (50th percentile or $q_{0.5}$ - a central tendency measure), the 90th percentile and the 10th percentile ($q_{0.9}$ and $q_{0.5}$, respectively, both used to indicate the dispersion around the median) over $s = 501$ simulations. For

Table 2: Summary of the LSAC dataset.

Attributes	**Mean (± std)**	**Attributes**	**Mean (± std)**	**Attributes**	**Mean (± std)**
decile1b	5.709 (±2.838)	decile3	5.729 (±2.830)	lsat	36.990 (±5.278)
ugpa	0.409 (±0.179)	zfygpa	0.145 (±0.921)	zgpa	0.080 (±0.977)
fulltime	1.073 (±0.261)	fam_inc	3.498 (±0.833)	male	0.564 (±0.495)
race	0.935 (±0.245)				

Table 3: Summary of the Pima Indian Diabetes dataset.

Attributes	**Mean (± std)**	**Attributes**	**Mean (± std)**	**Attributes**	**Mean (± std)**
Pregnancies	3.845 (±3.369)	Glucose	120.894 (±31.972)	BloodPressure	69.105 (±19.355)
SkinThickness	20.536 (±15.952)	Insulin	79.799 (±115.244)	BMI	31.992 (±7.884)
DiabetesPedigreeFunction	0.471 (±0.331)	Age	33.240 (±11.760)		

each simulation, we calculate the errors when estimating the exact SHAP values of all test samples. For the proposed k_{ADD}-SHAP, the percentile q_a, $a = 0.1, 0.5, 0.9$, is calculated as follows:

$$\bar{\varepsilon}_{a,k_{ADD}} = q_a \left(\frac{1}{n_{te}} \sum_{i'=1}^{n_{te}} \varepsilon^1_{k_{ADD},i'}, \ldots, \frac{1}{n_{te}} \sum_{i'=1}^{n_{te}} \varepsilon^s_{k_{ADD},i'} \right) \tag{30}$$

where $\varepsilon^r_{k_{ADD},i'}$, $r = 1, \ldots, s$ represents the squared error for test sample i' in simulation r. Equation (30) can be easily adapted to calculate the metrics when adopting the Kernel SHAP.

The results obtained when varying $n_{\mathcal{M}}$ (i.e., the number of coalitions selected to calculate the expected prediction evaluations) are presented in Figures 1, 2, 3 and 4. The central line represents the average median and the shaded area indicates the averaged dispersion between the 10th and 90th percentiles. For all datasets and trained models, the 3_{ADD}-SHAP leads to a faster approximation to the exact SHAP values in comparison with the Kernel SHAP. Moreover, the dispersion was lower for

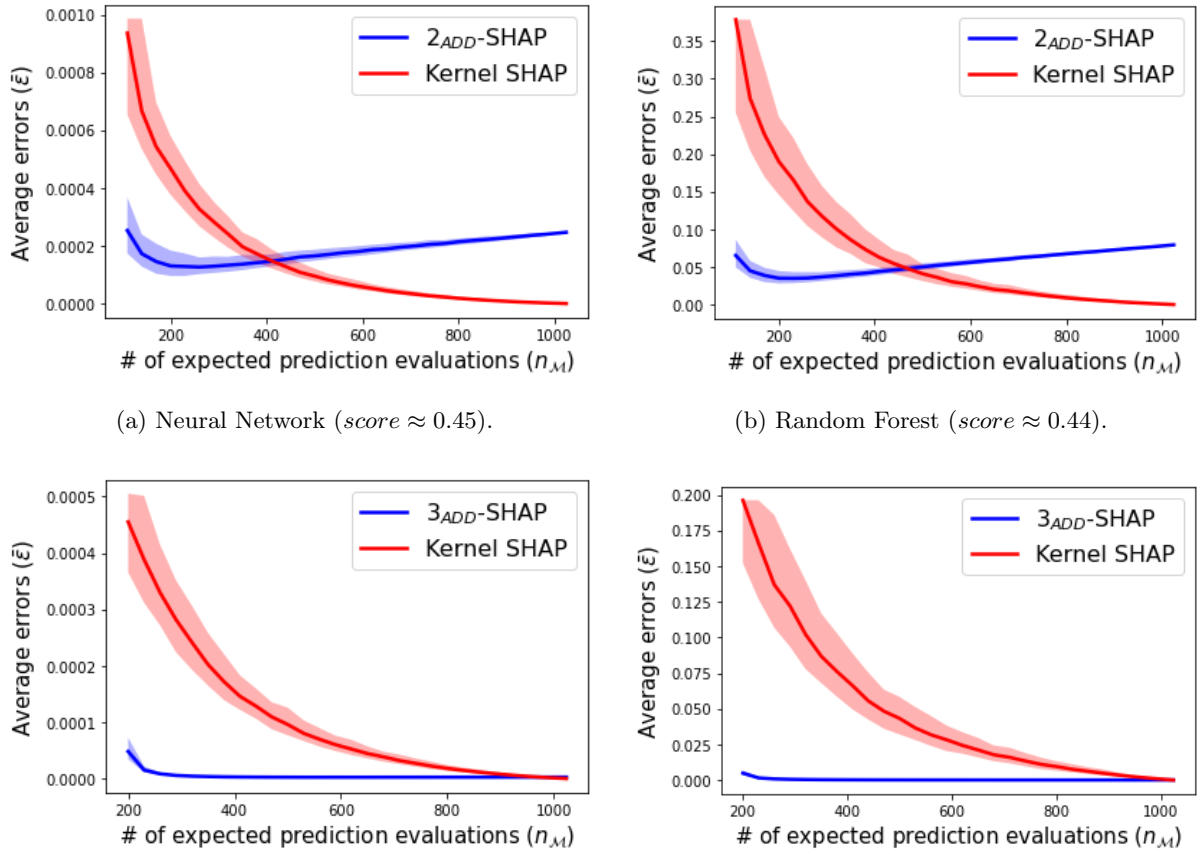
the 3_{ADD}-SHAP even with reduced numbers of expected prediction evaluations. For Kernel SHAP, one achieves a high dispersion for low numbers of expected prediction evaluations (see, especially, Figures 1, 3 and 4), which decreases as one includes more samples. With respect to the 2_{ADD}-SHAP, it has a good performance (better than the Kernel SHAP) for few evaluations, however, it diverges as more samples are include in the SHAP values estimation. An explanation for these results is that the 2_{ADD}-SHAP could rapidly approximate the exact SHAP values when less evaluations are used because it can avoid over-parametrization when only few data are considered. However, when increasing the number of expected prediction evaluations, the 2_{ADD}-SHAP has not enough flexibility to model the data and the adjusted parameters could not converge to the correct ones. As can also be seen in Figures 2 and 4, the 3_{ADD}-SHAP also diverges for a high number of evaluations (recall from Section 3.2 that we can not guarantee to achieve the exact SHAP values even if $\mathcal{M} = \mathcal{P}(M)$), however, it still achieves a very low error. Clearly, as we increase k, the parameters become more flexible to model the data and estimate the exact SHAP values.

4.2. The impact on computational time

In the previous experiment, we showed that the 3_{ADD}-SHAP requires less expected prediction evaluations in order to converge to the exact SHAP values. In this section, we discuss how our proposal impacts the computational time. For this purpose, it is convenient to analyze the computational cost when calculating $\mathbf{S}_{\mathcal{M},k}$ and $\bar{\mathbf{f}}$ (or $\mathbf{S}$ and $\mathbf{f}$ for the Kernel SHAP).

Both $\mathbf{S}_{\mathcal{M},k}$ and $\mathbf{S}$ require to compute a pseudo-inverse. The main difference between the Kernel SHAP and our proposal is that we have more parameters to be identified. Indeed, while the Kernel SHAP only provides the marginal contribution of each attribute, in our proposals (for $k \geq 2$), we also provide the interaction indices. When considering the Kernel SHAP, 2_{ADD}-SHAP and 3_{ADD}-SHAP, one has m, $m(m+1)/2$ and $m(m^2+5)/6$ parameters to be identified, respectively. As a consequence, the computational time needed to calculate $\mathbf{S}_{\mathcal{M},k}$ is greater than the time spent in $\mathbf{S}$ (for a fixed number of expected prediction evaluations and given all matrices and vectors used to calculate both of them). Figure 5 presents the average computational time (over 501 simulations) of 3_{ADD}-SHAP, 2_{ADD}-SHAP and Kernel SHAP for different numbers of expected prediction evaluations[7]. Clearly, for most of the datasets, the computational time of 3_{ADD}-SHAP when calculating $\mathbf{S}_{\mathcal{M},k}$ increases faster than the other methods as $n_{\mathcal{M}}$ increases. However, one can notice an exception with the Pima Indian Diabetes dataset. Note that this dataset has the lowest number of attributes. Therefore, in

[7]Computing device: Intel Core i7-8565U, CPU 1.80 GHz, 8.00 GB RAM, software Python 3.9.



(a) Neural Network ($score \approx 0.45$).

(b) Random Forest ($score \approx 0.44$).

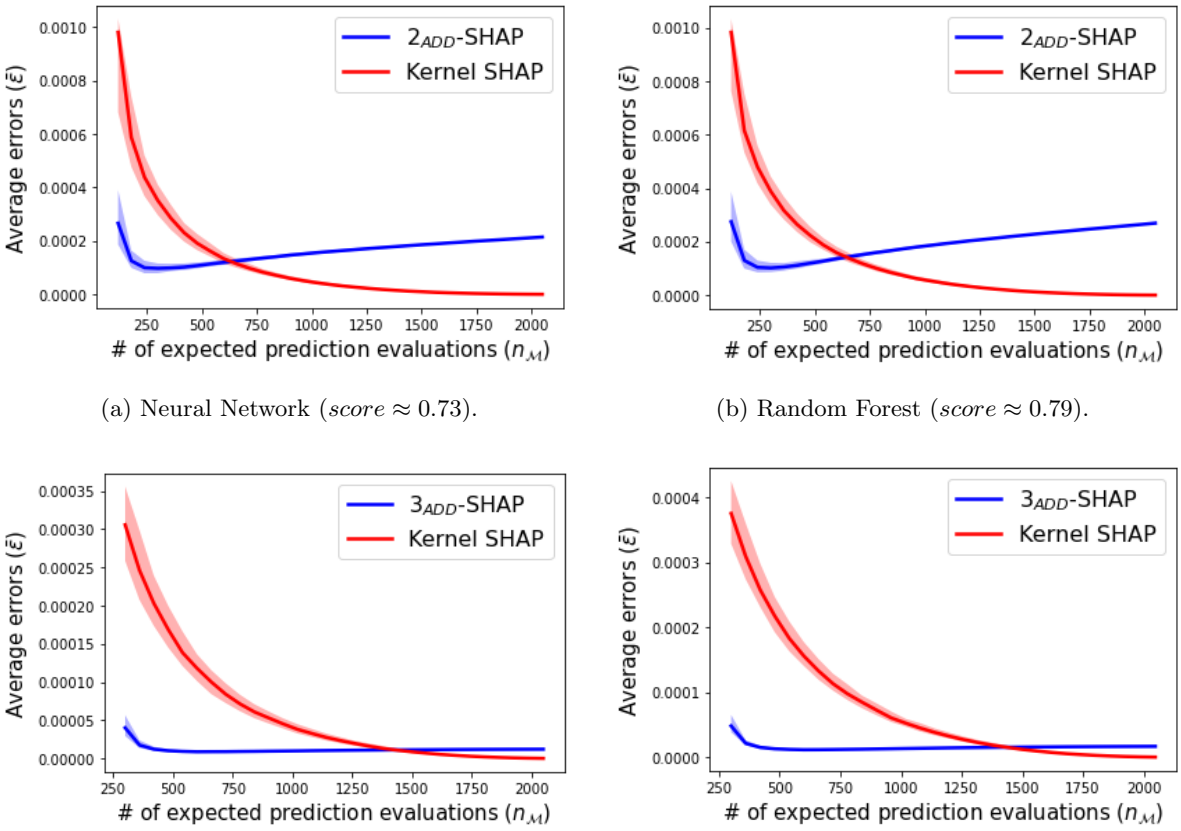
(c) Neural Network ($score \approx 0.45$).

(d) Random Forest ($score \approx 0.44$).

Figure 1: Comparison between the convergence of 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP (Diabetes dataset). For both Neural Network and Random Forest, the *score* indicates the coefficient of determination of the predicted outcomes given the test samples.

this case, all methods spent practically the same (very low) computational time.

The other computational cost that is worth to be studied is when calculating the vectors $\bar{\mathbf{f}}$ and $\mathbf{f}$, which consists in $n_{\mathcal{M}}$ expected prediction evaluations obtained from a trained machine learning model (i.e., this cost is also dependent on the adopted ML model). Indeed, the proposed 3_{ADD}-SHAP requires much less expected prediction evaluations to converge in comparison with the Kernel SHAP. Therefore, our goal is to show that, by reducing $n_{\mathcal{M}}$, the overall computational time (required to calculate both $\mathbf{S}_{\mathcal{M},k}$ and $\bar{\mathbf{f}}$) of our proposal is lower than the one of Kernel SHAP. For this analysis, we consider the 3_{ADD}-SHAP, which led to the faster convergence and has the highest cost when calculating $\mathbf{S}_{\mathcal{M},k}$. Table 4 presents the averaged computational cost for a single evaluation as well as until the convergence. We considered that the 3_{ADD}-SHAP converges to the exact SHAP values for the Diabetes, Red Wine, LSAC and Pima Indian Diabetes datasets with 290, 500, 400



(a) Neural Network ($score \approx 0.73$).

(b) Random Forest ($score \approx 0.79$).

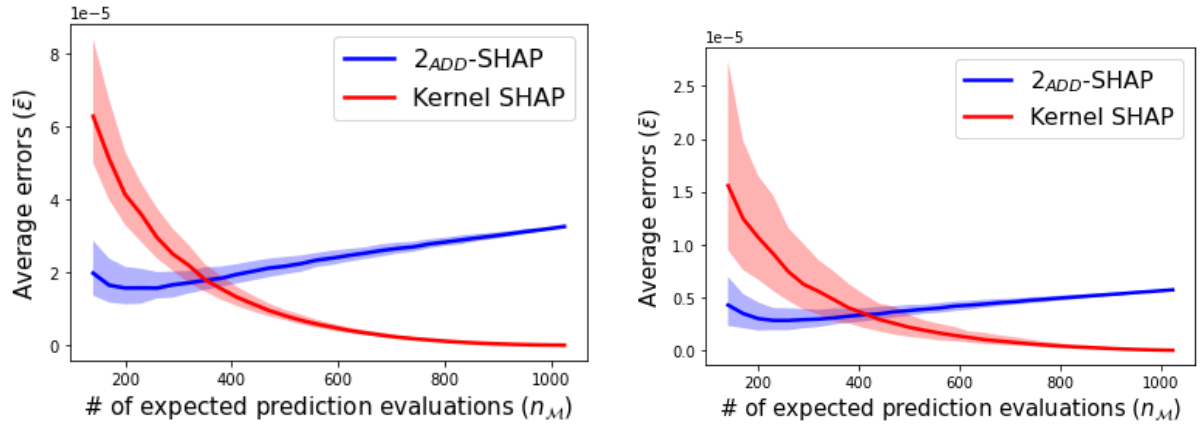
(c) Neural Network ($score \approx 0.73$).

(d) Random Forest ($score \approx 0.79$).

Figure 2: Comparison between the convergence of 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP (Red Wine dataset). For both Neural Network and Random Forest, the *score* indicates the accuracy given the test samples.

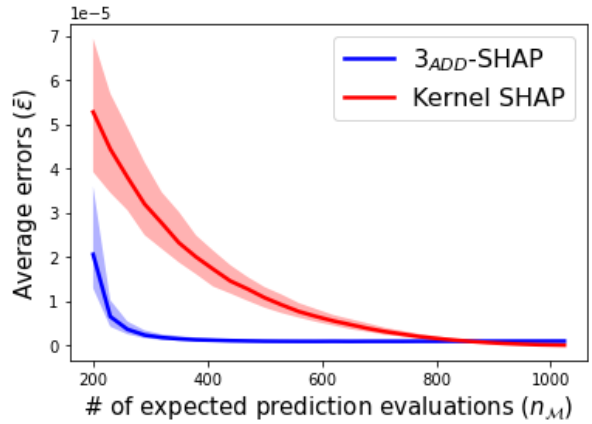
and 200 expected prediction evaluations, respectively (for both ML models). In Kernel SHAP, we assumed convergence for the Diabetes, Red Wine, LSAC and Pima Indian Diabetes datasets with 1000, 1400, 850 and 245 expected prediction evaluations, respectively (for both ML models).

As a first remark from the results presented in Table 4, the Random Forest takes much more time in comparison to the Neural Network in order to provide the expected predictions (see the cost for a single evaluation). Clearly, this result is independent on the approximation strategy. Another important remark is that the computational time to compute a few tens of expected predictions already exceeds the cost to calculate either $\mathbf{S}_{\mathcal{M},k}$ or $\mathbf{S}$. Therefore, as the Kernel SHAP needs much more expected prediction evaluations to converge to the exact SHAP values in comparison with the 3_{ADD}-SHAP, its overall computational time is much higher. This result was consistent with all datasets, even in the Pima Indian Diabetes whose number of attributes is lower than the others. It is also worth highlighting that, besides the gain in computational time, our proposal also provides

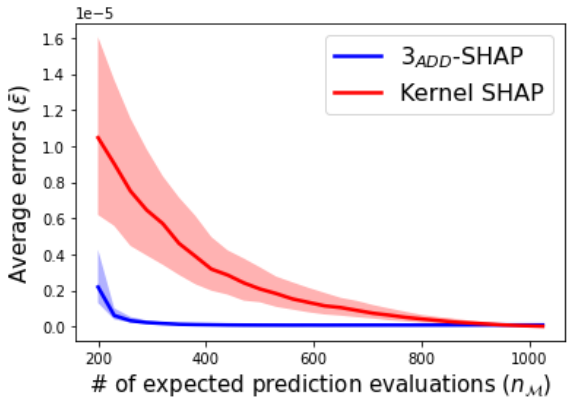


(a) Neural Network (*score* ≈ 0.90).

(b) Random Forest (*score* ≈ 0.90).



(c) Neural Network (*score* ≈ 0.90).



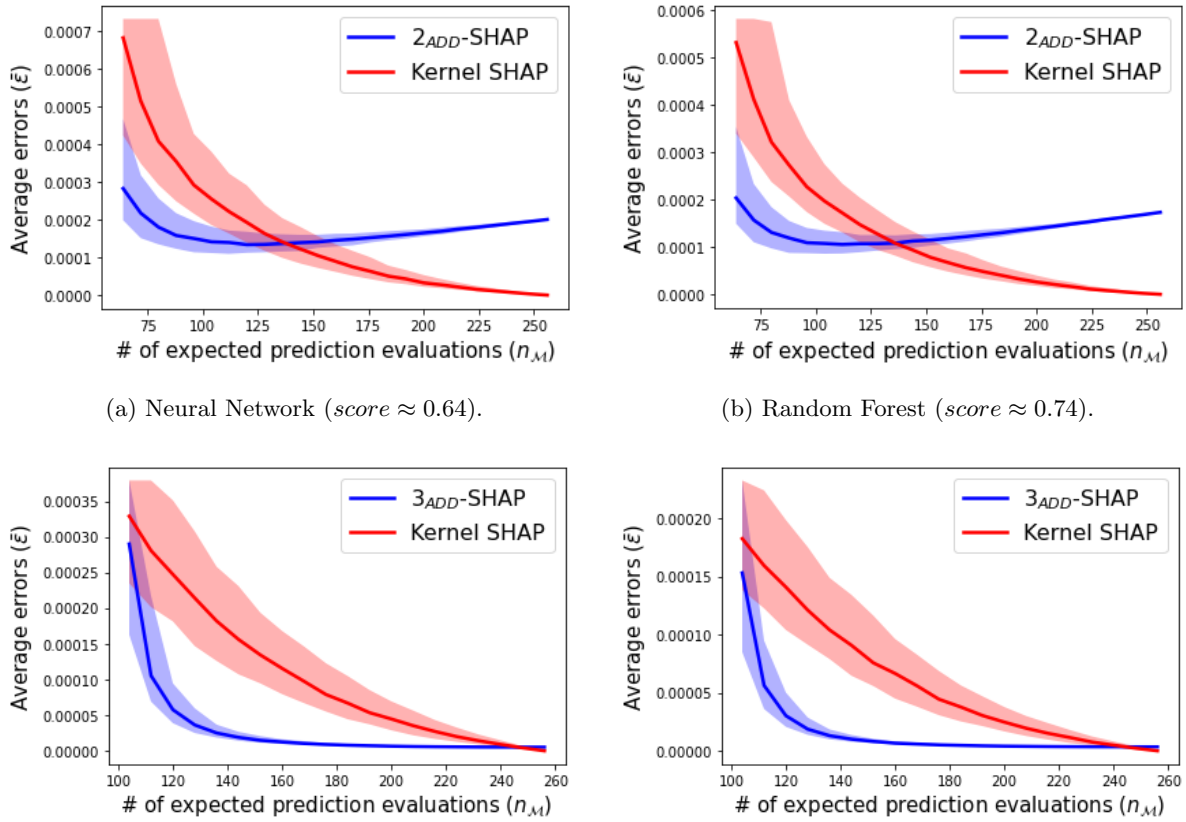
(d) Random Forest (*score* ≈ 0.90).

Figure 3: Comparison between the convergence of 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP (LSAC dataset). For both Neural Network and Random Forest, the *score* indicates the accuracy given the test samples.

the interaction indices between attributes, which is not the case in the Kernel SHAP.

4.3. Experiment comparing the obtained SHAP values

In this experiment, we compare the obtained SHAP values with the exact ones. For an instance of interest among the test data, we use the experiment described in Section 4.1 and select the SHAP values that lead to the median error over all simulations. For ease of visualization, we only plotted the five attributes that contribute the most (either positively or negatively) according to the exact SHAP values. As an illustrative example and without loss of generality, we selected a test sample $\mathbf{x}^*$ from the Diabetes dataset that has the attributes values described in Table 5 (recall that this dataset is already centered with zero mean). The predicted measure of diabetes progression is equal to 84, which is less than the overall expected prediction provided by both Neural Network



(a) Neural Network ($score \approx 0.64$).

(b) Random Forest ($score \approx 0.74$).

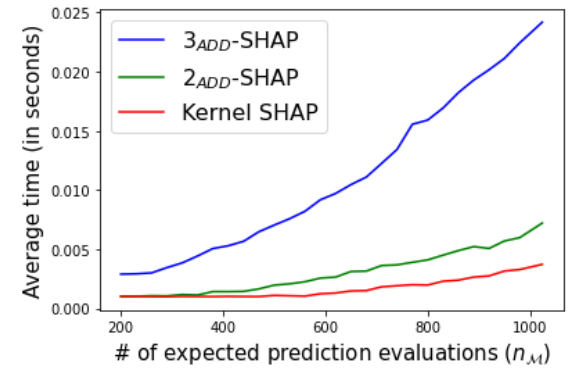
(c) Neural Network ($score \approx 0.64$).

(d) Random Forest ($score \approx 0.74$).

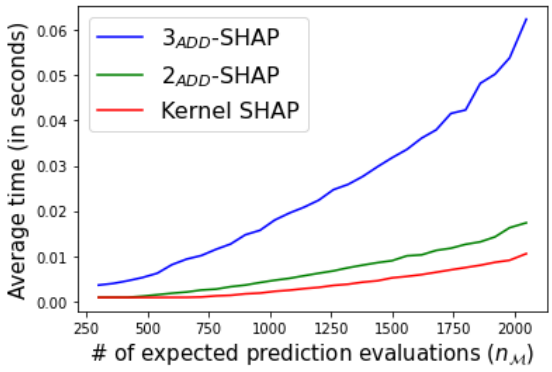
Figure 4: Comparison between the convergence of 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP (Pima Indian Diabetes dataset). For both Neural Network and Random Forest, the *score* indicates the accuracy given the test samples.

and Random Forest (154.92 and 153.81, respectively). This means that the SHAP values help to explain, for the instance of interest $\mathbf{x}^*$, how each attribute value contributes to decrease the diabetes progression measure from the overall prediction until the actual 84.

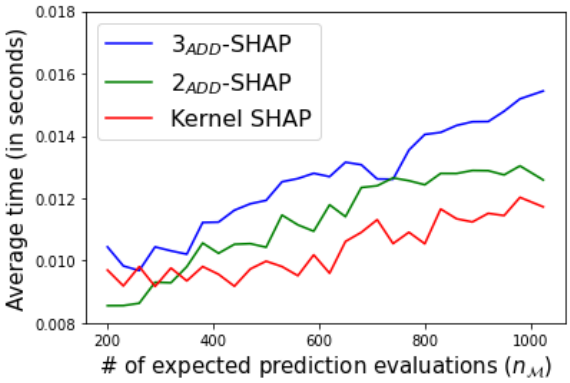
Figure 6 presents the estimated SHAP values when using $n_{\mathcal{M}} = 290$, $n_{\mathcal{M}} = 590$, $n_{\mathcal{M}} = 890$ different coalitions of attributes to calculate the expected prediction evaluations. As a first remark, we note that the estimated SHAP values (specially the illustrated five ones) for the Neural Network (Figures 6a, 6c and 6e) practically do not change regardless the number of predicted evaluations. All approaches led to very small errors, i.e., they could rapidly approximate the exact SHAP values associated with the Neural Networks model. For the Random Forest, we see that the contributions provided by the 3_{ADD}-SHAP are close to the exact ones even with small number of predicted evaluations (see Figure 6b). As one increases the number of evaluations, the Kernel SHAP converges to the exact SHAP values.



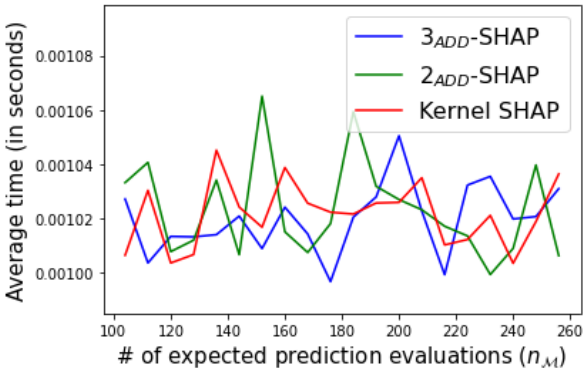
(a) Diabetes dataset.



(b) Red Wine dataset.



(c) LSAC dataset.



(d) Pima Indian Diabetes dataset.

Figure 5: Comparison between the computational time, when calculating $\mathbf{S}_{\mathcal{M},k}$ and $\mathbf{S}$, of 3_{ADD}-SHAP, 2_{ADD}-SHAP and Kernel SHAP.

Regarding the Red Wine dataset, we selected as an illustrative example a test sample classified as a good wine. The attributes values are described in Table 6. The overall expected probability prediction for class 1 (good wine) for both Neural Network and Random Forest is approximately 0.53. In this case, the SHAP values indicate the contributions of attributes that increase the probability of being classified as a good wine from the overall expected probability until the actual classification (class value equals to 1).

Figure 7 presents the estimated SHAP values when using $n_{\mathcal{M}} = 420$, $n_{\mathcal{M}} = 1020$ and $n_{\mathcal{M}} = 1800$ coalitions of attributes. As in the previous dataset, we can see that, even for a reduced number of samples, the 3_{ADD}-SHAP converges faster to the exact SHAP values. When the number of expected prediction evaluations increases, the Kernel SHAP converges to the exact SHAP values while the 2_{ADD}-SHAP slightly diverges.

Based on the LSAC dataset, we selected a candidate that succeeded the bar exam and is de-

Table 4: Comparison between the computational time in the expected prediction calculation.

Dataset	**ML model**	**Averaged computational time (in seconds)**				
		For a single evaluation	**Until convergence (3_{ADD}-SHAP)**		**Until convergence (Kernel SHAP)**	
			$\bar{\mathbf{f}}$	$\bar{\mathbf{f}} + \mathbf{S}_{\mathcal{M},k}$	$\mathbf{f}$	$\mathbf{f} + \mathbf{S}$
Diabetes	Neural Network	0.0046	1.334	1.338	4.600	4.604
	Random Forest	0.1699	49.271	49.275	169.900	169.904
Red Wine	Neural Network	0.0010	0.500	0.508	1.400	1.405
	Random Forest	0.3701	185.050	185.058	518.140	518.145
LSAC	Neural Network	0.0162	6.480	6.491	13.770	13.781
	Random Forest	2.6260	1050.400	1050.411	2232.100	2231.111
Pima Indian Diabetes	Neural Network	0.0005	0.100	0.101	0.122	0.123
	Random Forest	0.2488	49.760	49.761	60.956	60.957

Table 5: Summary of the selected test sample - Diabetes dataset.

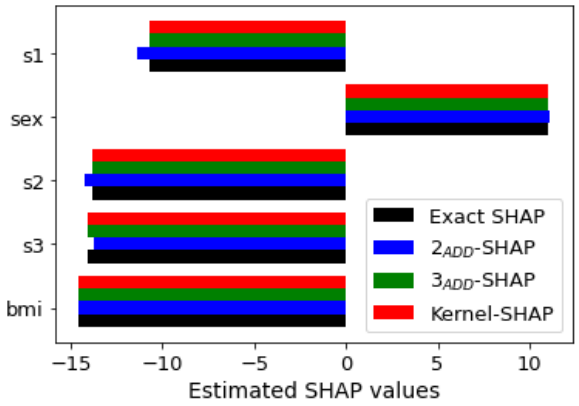
Attribute	**Value**	**Attribute**	**Value**	**Attribute**	**Value**
age	0.009	blood serum 1	0.099	blood serum 5	−0.021
sex	−0.045	blood serum 2	0.094	blood serum 6	0.007
body mass index	−0.024	blood serum 3	0.071		
average blood pressure	−0.026	blood serum 4	−0.002		

Table 6: Summary of the selected test sample - Red Wine dataset.

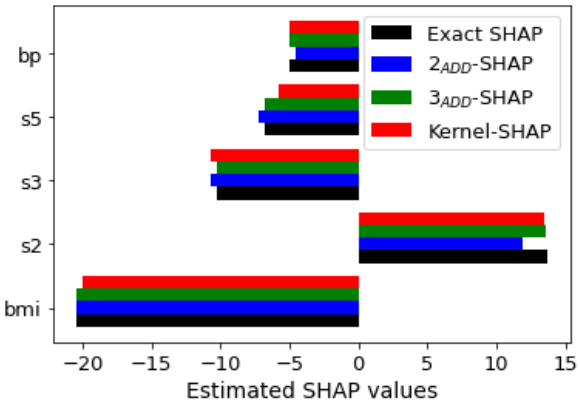
Attribute	**Value**	**Attribute**	**Value**	**Attribute**	**Value**
fixed acidity	9.4	chlorides	0.08	pH	3.15
volatile acidity	0.3	free sulfur dioxide	6	sulphates	0.92
citric acid	0.56	total sulfur dioxide	17	alcohol	11.7
residual sugar	2.8	density	0.9964		

scribed by the attributes presented in Table 7. With overall expected probability predictions for class 1 (pass the bar exam) for the Neural Network and the Random Forest equal to 0.97 and 0.91, respectively, the SHAP values will indicate the contributions of attributes until the actual classification (class value equals to 1).

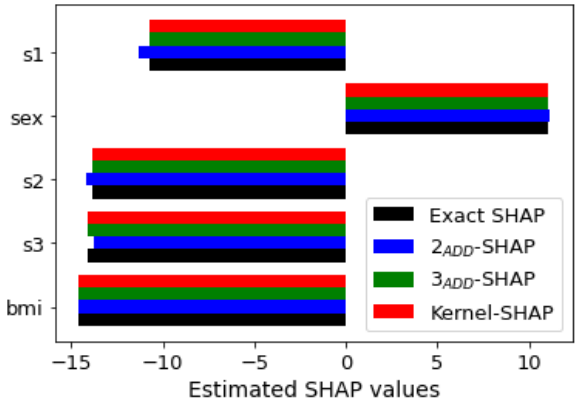
Figure 8 presents the estimated SHAP values for $n_{\mathcal{M}} = 290$, $n_{\mathcal{M}} = 590$ and $n_{\mathcal{M}} = 890$ coalitions



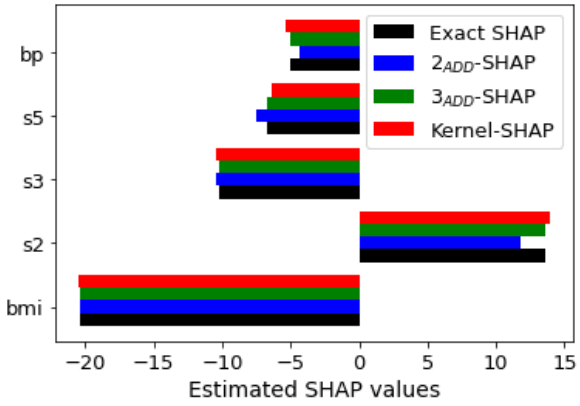
(a) Neural Network, $n_{\mathcal{M}} = 290$.



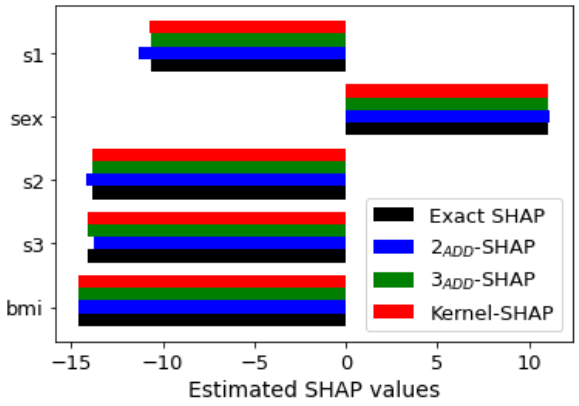
(b) Random Forest, $n_{\mathcal{M}} = 290$.



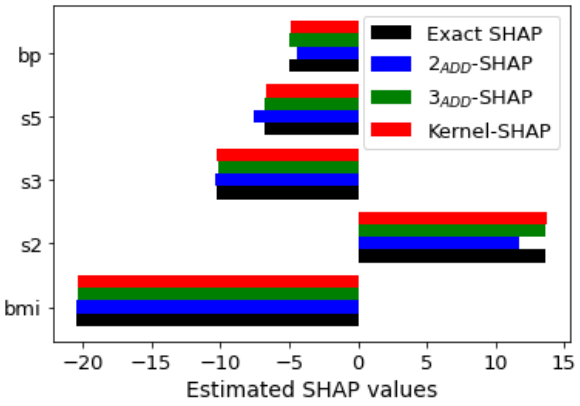
(c) Neural Network, $n_{\mathcal{M}} = 590$.



(d) Random Forest, $n_{\mathcal{M}} = 590$.

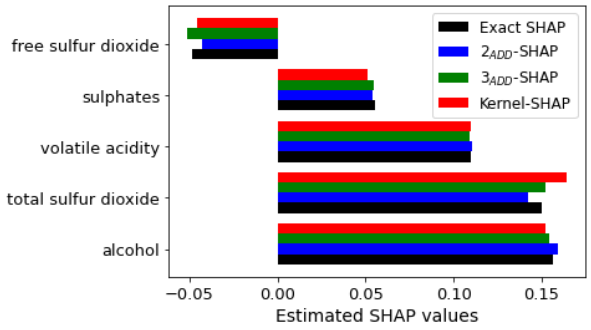


(e) Neural Network, $n_{\mathcal{M}} = 890$.

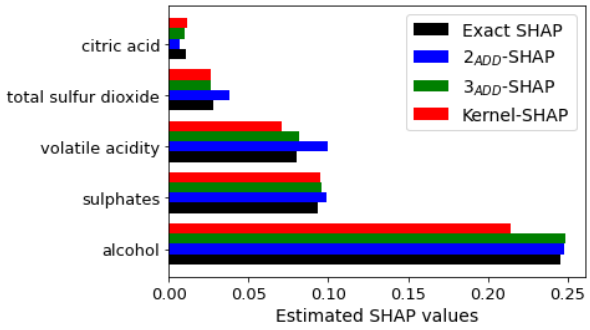


(f) Random Forest, $n_{\mathcal{M}} = 890$.

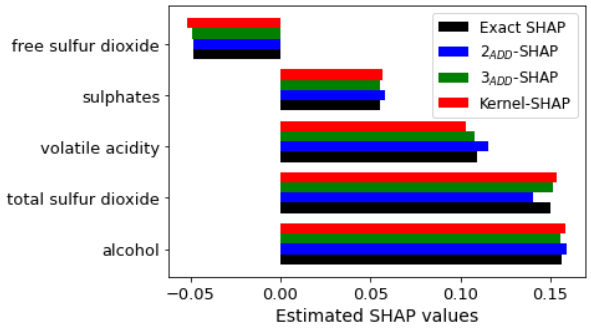
Figure 6: Comparison between the estimated SHAP values provided by the 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP for different machine learning models and varying the number of coalitions used to calculate the expected prediction evaluations (Diabetes dataset).



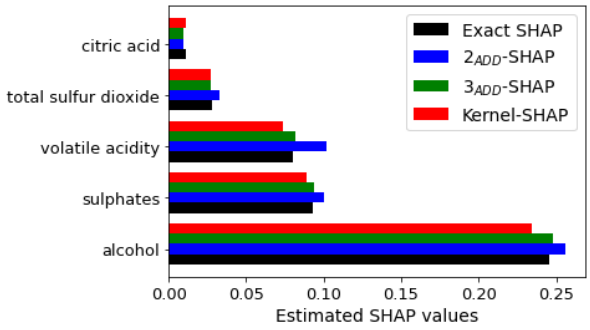
(a) Neural Networks, $n_{\mathcal{M}} = 420$.



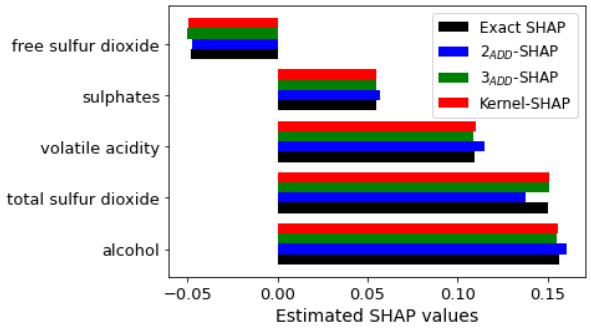
(b) Random Forest, $n_{\mathcal{M}} = 420$.



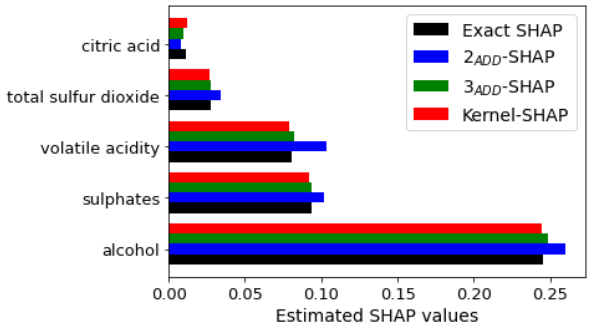
(c) Neural Networks, $n_{\mathcal{M}} = 1020$.



(d) Random Forest, $n_{\mathcal{M}} = 1020$.



(e) Neural Networks, $n_{\mathcal{M}} = 1800$.



(f) Random Forest, $n_{\mathcal{M}} = 1800$.

Figure 7: Comparison between the estimated SHAP values provided by the 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP for different machine learning models and varying the number of coalitions used to calculate the expected prediction evaluations (Red Wine dataset).

of attributes. The results lead to the same conclusion as for the previous case. The 3_{ADD}-SHAP converges faster to the exact SHAP values and the 2_{ADD}-SHAP slightly diverges as $n_{\mathcal{M}}$ increases.

Finally, for the Pima Indian Diabetes dataset, we selected a person without diabetes. This person is described by the attributes presented in Table 8. For the Neural Network and the Random Forest, the overall expected probability predictions for class 1 (positive for diabetes) are equal to 0.34 and

Table 7: Summary of the selected test sample - LSAC dataset.

Attribute	Value	Attribute	Value	Attribute	Value	Attribute	Value
decile1b	3	decile3	3	lsat	34	ugpa	3.2
zfygpa	−0.47	zgpa	−0.79	fulltime	1	fam_inc	4
male	1	race	1				

0.36, respectively. The SHAP values will, then, indicate the impact of each feature in decreasing the probability of having diabetes.

Table 8: Summary of the selected test sample - Pima Indian Diabetes dataset.

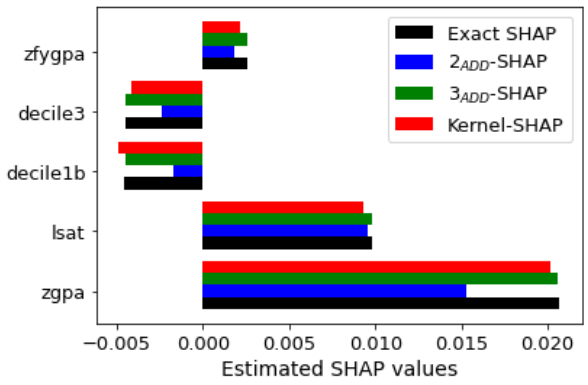
Attribute	Value	Attribute	Value	Attribute	Value
Pregnancies	2	Glucose	108	BloodPressure	64
SkinThickness	0	Insulin	0	BMI	30.8
DiabetesPedigreeFunction	0.158	Age	21		

Figure 9 presents the estimated SHAP values for $n_{\mathcal{M}} = 144$, $n_{\mathcal{M}} = 200$ and $n_{\mathcal{M}} = 240$ coalitions of attributes. The results are consistent with the previous datasets, with a faster convergence for 3_{ADD}-SHAP. However, it is interesting to note in Figures 8e and 8f that the 3_{ADD}-SHAP slightly diverges with the use of practically all coalitions. This is in accordance with the error presented in Figure 4.

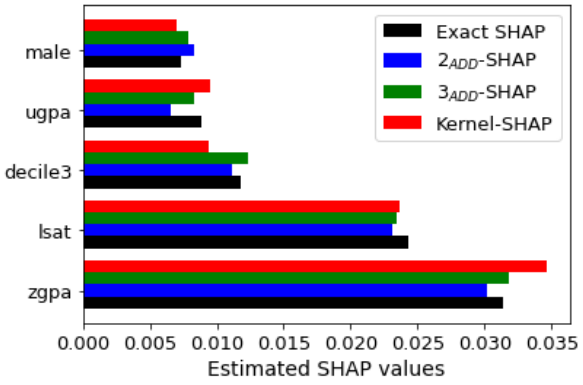
4.4. Illustrative example and results visualization

The purpose of this last experiment is to apply our proposal to visualize the attributes contribution towards the actual predicted outcome. We use as an illustrative example the Red Wine dataset and applied the 3_{ADD}-SHAP. We also consider the test sample used in the previous experiment, which is classified as a good wine. Based on 1500 predicted evaluations and using the Random Forest, the contributions of attributes are presented in Figure 10. Note that there are three attributes that contribute the most into the predicted outcome: alcohol, sulphates and volatile acidity. They are all positively contributing to predict the sample as a good wine.

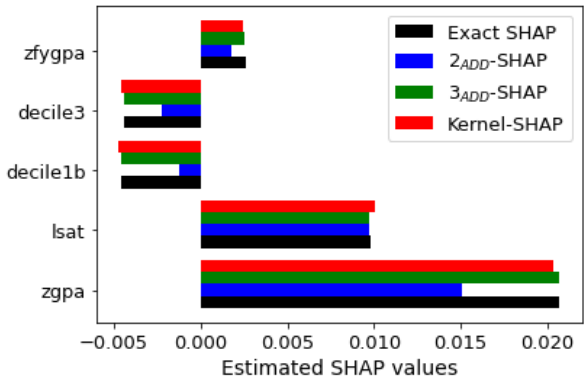
Recall that, more than the contribution of features, our proposal automatically provides the interaction degree between them. We highlight that these interaction effects do not come up with the original Kernel SHAP formulation. Indeed, further adaptations must be made in Kernel SHAP in order to retrieve the interaction effects (Lundberg et al., 2020). Figure 11 shows the interaction



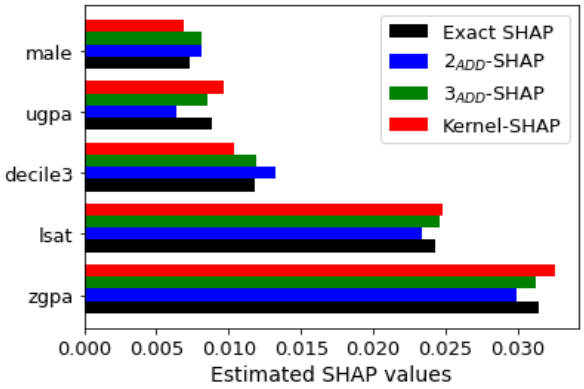
(a) Neural Networks, $n_{\mathcal{M}} = 290$.



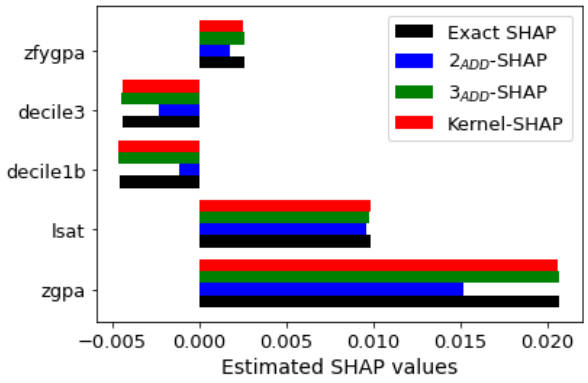
(b) Random Forest, $n_{\mathcal{M}} = 290$.



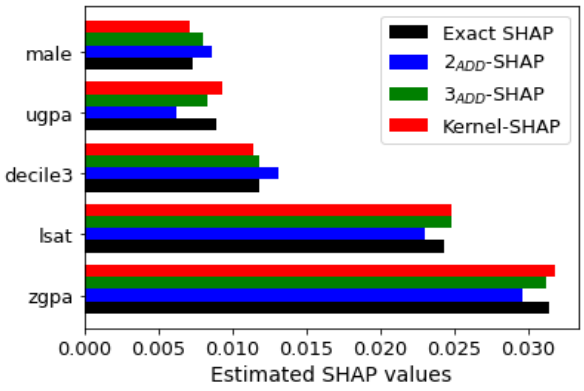
(c) Neural Networks, $n_{\mathcal{M}} = 590$.



(d) Random Forest, $n_{\mathcal{M}} = 590$.



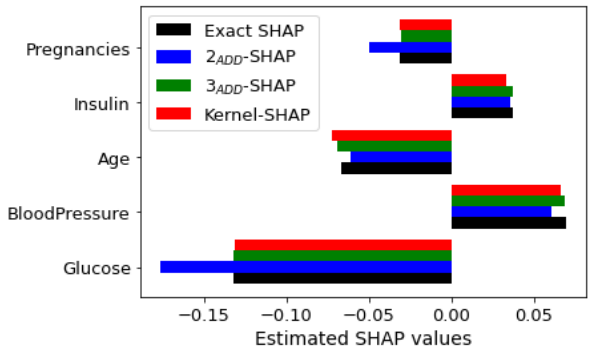
(e) Neural Networks, $n_{\mathcal{M}} = 890$.



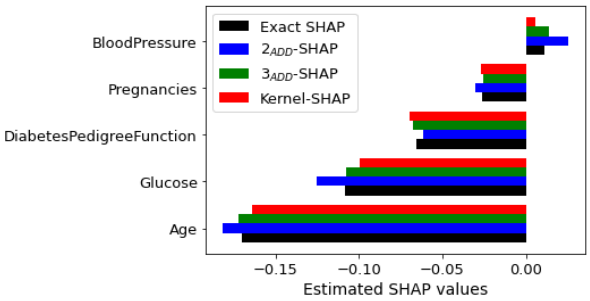
(f) Random Forest, $n_{\mathcal{M}} = 890$.

Figure 8: Comparison between the estimated SHAP values provided by the 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP for different machine learning models and varying the number of coalitions used to calculate the expected prediction evaluations (LSAC dataset).

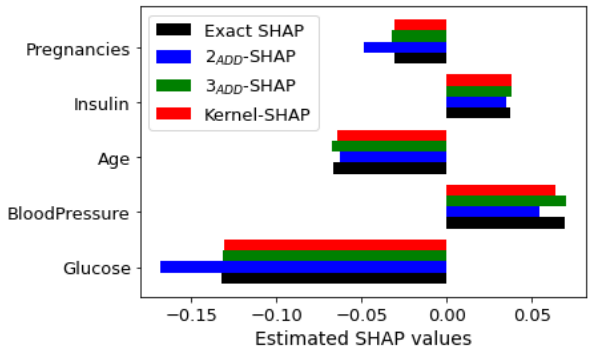
degree between attributes for the considered test sample. It indicates that, although volatile acidity, sulphates and alcohol (attributes 1, 9 and 10, respectively) contribute the most to the predicted



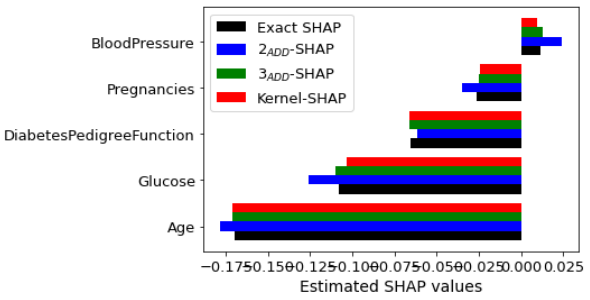
(a) Neural Networks, $n_{\mathcal{M}} = 144$.



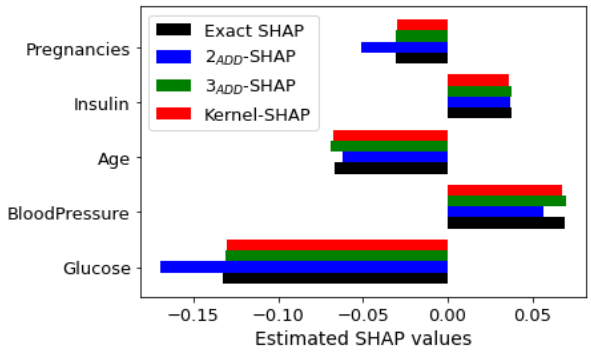
(b) Random Forest, $n_{\mathcal{M}} = 144$.



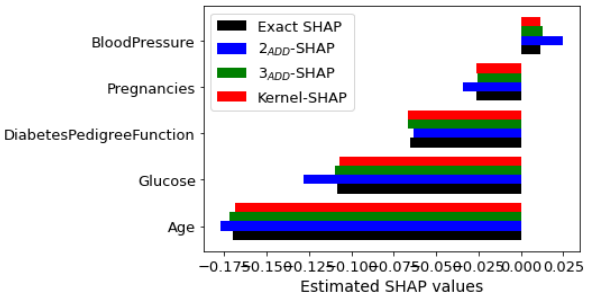
(c) Neural Networks, $n_{\mathcal{M}} = 200$.



(d) Random Forest, $n_{\mathcal{M}} = 200$.



(e) Neural Networks, $n_{\mathcal{M}} = 240$.



(f) Random Forest, $n_{\mathcal{M}} = 240$.

Figure 9: Comparison between the estimated SHAP values provided by the 2_{ADD}-SHAP, 3_{ADD}-SHAP and Kernel SHAP for different machine learning models and varying the number of coalitions used to calculate the expected prediction evaluations (Pima Indian Diabetes dataset).

outcome, there are negative interactions between alcohol and both volatile acidity and sulphates. This suggests that there are some redundancies between alcohol and the other two attributes when predicting the sample as a good wine.

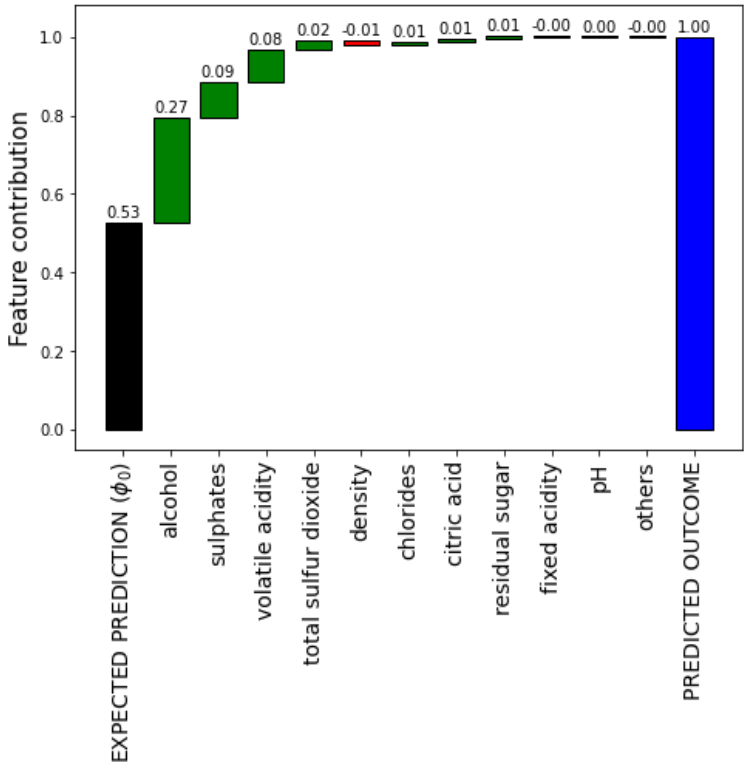


Figure 10: Attributes contribution towards the predicted outcome - 3_{ADD}-SHAP and Red Wine dataset.

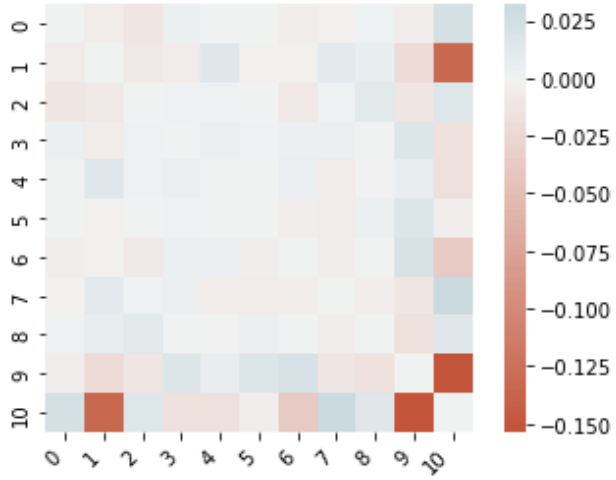


Figure 11: Interaction degree between attributes - 3_{ADD}-SHAP and Red Wine dataset.

5. Conclusions and future perspectives

Interpretability in machine learning has become as important as accuracy in real problems. For instance, even if there is a correct classification (e.g., a denied credit), the explanation about how this result was achieved is required to ensure the model trustfulness. A very famous model-agnostic

algorithm for machine learning interpretability is the SHAP method. Based on the Shapley values, the SHAP method indicates the contribution of each attribute in the predicted outcome. For this purpose, we look at the machine learning task as a cooperative game theory problem and calculate the marginal contribution of each attribute by taking the predicted outcomes of all possible coalitions of attributes. A point of attention in this calculation is that, as the number of predicted outcomes evaluations exponentially increases with the number of attributes, one may not be able to obtain the exact SHAP values.

In order to reduce the computational effort of SHAP method, the Kernel SHAP emerges as a clever strategy to approximate the SHAP values. However, its formulation is not easy to follow and no further considerations about the modeled game are assumed when approximating the SHAP values. In this paper, we first proposed a straightforward Choquet integral-based formulation for local interpretability. As the parameters used in the Choquet integral are directly associated with the Shapley values, our formulation also leads to the SHAP values. Therefore, we can also exploit the benefits of the SHAP values when interpreting local predictions. Moreover, our formulation also provides the interaction effects between attributes without further adaptations in the algorithm. Therefore, we can interpret the marginal contribution of each attribute towards the predicted outcome and how they interact between them.

As a second contribution, we exploit the concept of k-additive games. The use of k-additive models has revealed to be useful in multicriteria decision making problems in order to reduce the number of parameters in capacity-based aggregation functions (such as the Choquet integral) while keeping a good level of flexibility in data modeling. Therefore, as attested by the numerical experiments, when adopting k-additive games (specially the 3-additive, which leads to the proposed 3_{ADD}-SHAP), we could approximate the SHAP values using less predicted outcomes evaluations in comparison with the Kernel SHAP. As one reduced the number of parameters in the Choquet integral formulation, one avoided over-parametrization in scenarios with a low number of predicted outcomes evaluations. On the other hand, as we restricted the modeling data domain, in the scenario with all evaluations the proposed k_{ADD}-SHAP may slightly diverge from the exact SHAP values. However, as could be seen in the experiments, this difference is very low (mainly for the 3_{ADD}-SHAP) and it does not affect the interpretability.

Future works include to extend the proposed approach when assuming that the attributes are dependent. In such a scenario, the formulation could be adjusted in order to better approximate the Shapley values (Aas et al., 2021). Moreover, we also would like to investigate an automatic

approach to assess the value of k. Although 3-additive models achieved very interesting results, such automatic strategy would be useful to verify if $k > 3$ can better deal with specific situations, such as high dimensional data. Another perspective consists in evaluating the use of other game-based aggregation functions to deal with local interpretability. However, as some of them do not ensure the efficiency property, one must be careful in how one can apply them in the context of machine learning in a way that the feature attribution makes sense for local or global interpretability.

Acknowledgments

Work supported by São Paulo Research Foundation (FAPESP) under the grants #2020/09838-0 (BI0S - Brazilian Institute of Data Science), #2020/10572-5 and #2021/11086-0. L. T. Duarte would like to thank the National Council for Scientific and Technological Development (CNPq, Brazil) for the financial support.

Appendix A

We here describe the desired properties satisfied by SHAP values, which are derived from the Shapley values properties (Shapley, 1953; Young, 1985). Recall that $f(\mathbf{x})$ is the predicted outcome of a trained model $f(\cdot)$, $\mathbf{x}$ is the instance to be explained and $\mathbf{z}'$ is a binary vector. The proofs are provided in the original SHAP paper (Lundberg & Lee, 2017).

Property 1. Local accuracy (or efficiency)

$$f(\mathbf{x}) = \phi_0 + \sum_{j=1}^{m} \phi_j(f, \mathbf{x}) \tag{31}$$

The local accuracy property states that the predicted outcome $f(\mathbf{x})$ can be decomposed by the sum of the SHAP values and the overall expected prediction ϕ_0.

Property 2. Missingness

If, for all subset of attributes represented by the coalition $\mathbf{z}'$,

$$f\left(h_{\mathbf{x}}(\mathbf{z}')\right) = f\left(h_{\mathbf{x}}(\mathbf{z}'\backslash j)\right), \tag{32}$$

then $\phi_j(f, \mathbf{x}) = 0$. This property states that, if adding attribute j into the coalition the expected prediction remains the same, the marginal contribution of such an attribute is null.

Property 3. Consistency (or monotonicity)

For any two models $f(\cdot)$ and $f'(\cdot)$, if

$$f'\left(h_{\mathbf{x}}(\mathbf{z}')\right) - f'\left(h_{\mathbf{x}}(\mathbf{z}'\backslash j)\right) \geq f\left(h_{\mathbf{x}}(\mathbf{z}')\right) - f\left(h_{\mathbf{x}}(\mathbf{z}'\backslash j)\right) \tag{33}$$

for any binary vector $\mathbf{z}' \in \{0,1\}^m$, then $\phi_j(f', \mathbf{x}) \geq \phi_j(f, \mathbf{x})$. The consistency property states that, if one changes the trained model and the contribution of an attribute j increases or stays the same regardless of the other inputs, the marginal contribution of such an attribute should not decrease.

References

Aas, K., Jullum, M., & Løland, A. (2021). Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *Artificial Intelligence*, *298*, 103502.

Ahsan, M. M., & Siddique, Z. (2022). Machine learning-based heart disease diagnosis: A systematic literature review. *Artificial Intelligence in Medicine*, *128*, 102289.

Ben-Israel, D., Jacobs, W. B., Casha, S., Lang, S., Ryu, W. H. A., de Lotbiniere-Bassett, M., & Cadotte, D. W. (2020). The impact of machine learning on patient care: A systematic review. *Artificial Intelligence in Medicine*, *103*, 101785.

Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. *Artificial Intelligence Review*, *54*, 1937–1967.

Biau, G., & Scornet, E. (2016). A random forest guided tour. *Test*, *25*, 197–227.

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, *8*, 1–34.

Chen, H., Lundberg, S., & Lee, S.-I. (2021). Explaining models by propagating Shapley values of local components. In *Explainable AI in Healthcare and Medicine* (pp. 261–270). Springer, Cham.

Choquet, G. (1954). Theory of capacities. *Annales de l'Institut Fourier*, *5*, 131–295.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T., & Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems*, *47*, 547–553.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, *32*, 407–499.

Fawagreh, K., Gaber, M. M., & Elyan, E. (2014). Random forests: From early developments to recent advancements. *Systems Science and Control Engineering*, *2*, 602–609.

Garreau, D., & von Luxburg, U. (2020). Looking deeper into tabular lime. *ArXiv ID: 2008.11092*, . URL: `http://arxiv.org/abs/2008.11092`.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2018). Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA 2018)* (pp. 80–89). IEEE.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Grabisch, M. (1996). The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Research*, *89*, 445–456.

Grabisch, M. (1997a). Alternative representations of discrete fuzzy measures for decision making. *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*, *5*, 587–607.

Grabisch, M. (1997b). k-order additive discrete fuzzy measures and their representation. *Fuzzy Sets and Systems*, *92*, 167–189.

Grabisch, M. (2016). *Set Functions, games and capacities in decision making*. Springer International Publishing.

Grabisch, M., Duchêne, J., Lino, F., & Perny, P. (2002). Subjective evaluation of discomfort in sitting positions. *Fuzzy Optimization and Decision Making*, *1*, 287–312.

Grabisch, M., & Labreuche, C. (2010). A decade of application of the Choquet and sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, *175*, 247–286.

Grabisch, M., Prade, H., Raufaste, E., & Terrier, P. (2006). Application of the Choquet integral to subjective mental workload evaluation. *IFAC Proceedings Volumes*, *39*, 135–140.

Kruppa, J., Schwarz, A., Arminger, G., & Ziegler, A. (2013). Consumer credit risk: Individual probability estimates using machine learning. *Expert Systems with Applications*, *40*, 5125–5131.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning* volume 521.

Lipton, Z. C. (2018). The mythos of machine learning interpretability. *Machine Learning*, *16*, 31–57.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, *2*, 56–67.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30* (pp. 4765–4774).

Lundberg, S. M., Nair, B., Vavilala, M. S., Horibe, M., Eisses, M. J., Adams, T., Liston, D. E., Low, D. K. W., Newman, S. F., Kim, J., & Lee, S. I. (2018). Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nature Biomedical Engineering*, *2*, 749–760.

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, *267*, 1–38.

Molnar, C. (2021). *Interpretable machine learning*. URL: `https://christophm.github.io/interpretable-ml-book/`.

Murofushi, T., & Soneda, S. (1993). Techniques for reading fuzzy measures (iii): interaction index. In *9th fuzzy system symposium* (pp. 693–696).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pelegrina, G. D., Duarte, L. T., Grabisch, M., & Romano, J. M. T. (2020). The multilinear model in multicriteria decision making: The case of 2-additive capacities and contributions to parameter identification. *European Journal of Operational Research*, *282*.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135–1144).

Setzu, M., Guidotti, R., Monreale, A., Turini, F., Pedreschi, D., & Giannotti, F. (2021). Glocalx - from local to global explanations of black box AI models. *Artificial Intelligence*, *294*, 103457.

Shapley, L. S. (1953). A value for n-person games. In W. Kuhn, & A. W. Tucker (Eds.), *Annals of mathematics studies: Vol. 28. Contributions to the theory of games, Vol. II* (pp. 307–317). Princeton University Press.

Smith, J. W., Everhart, J. E., Dickson, W. C., Knowler, W. C., & Johannes, R. S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Annual Symposium on Computer Applications in Medical Care* (pp. 261–265). American Medical Informatics Association.

Wightman, L. F. (1998). *LSAC national longitudinal bar passage study*. Technical Report.

Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H., & Wang, C. (2018). Machine learning and deep learning methods for cybersecurity. *IEEE Access*, *6*, 35365–35381.

Young, H. P. (1985). Monotonic solutions of cooperative games. *International Journal of Game Theory*, *14*, 65–72.

Štrumbelj, E., & Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, *11*, 1–18.

Štrumbelj, E., & Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, *41*, 647–665.