UNIVERSITY
OF OSLO

Bachelor's thesis

Thinking inside the box

Explaining AI predictions using Shapley values

Julie Natland Bjørnstad

STK–MAT2011 – Project Work in Finance, Insurance, Risk and Data Analysis
10 ECTS study points

Supervised by Camilla Lingjærde

Department of Mathematics
Faculty of Mathematics and Natural Sciences

Spring 2025

Abstract

As artificial intelligence (AI) becomes increasingly integral to decision-making processes, it is crucial to ensure that these models are transparent, fair, and interpretable. Explainable AI (XAI) seeks to address the issue of "black-box" models by providing insights into their decision-making processes. Among many methods developed for this purpose, Shapley values stand out due to their solid theoretical foundation in cooperative game theory, offering a fair allocation of feature importance in model predictions. This report explores the use of Shapley values in XAI, beginning with a theoretical introduction to the concept and its role in model interpretability. Through a simulation study we compare the performance of different estimation methods for Shapley values. The report concludes with a discussion of the advantages and limitations of estimation methods for Shapley values.

Contents

1 Introduction

As the world grows more and more relient on the use of artificial intelligence (AI), the importance of understanding the models that we use becomes increasingly apparent. Whilst the desire for interpretable AI has been around for long, the field of explainable artifical intelligence (XAI) has only gained significant traction in recent years [4]. With the uprise of more complex and opaque AI-methods like Deep Neural Networks (DNN), one has seen a boost in publications within the field of XAI. Methods like DNN bring an entirely new realm of possibilites, with the complex structure allowing for a broad range of applications [28]. A plethora of fields, including medicine, law, governance, finance and many more, are looking to the world of AI for better, faster and more reliable solutions [36]. Many even hope that the introduction of data-driven decision-making will eliminate human bias, allowing for more sound reasoning and in turn making verdicts more fair [2, 14].

But with the new possibilites, comes new challenges; compared to simpler models used in the early days of AI, newer methods are almost impossible to interpret - an issue commonly described as the "black-box problem" [28]. As the wording entices, this describes a lack of understanding for what goes on inside the machine learning model, i.e. "the box". Even the developers behind the algortihms have little to no insight to exactly how their models make decisions based on different input, and how features are weighted in the decision. Without the ability to argue what features have influenced the predictions, one lacks the ability to ensure accountability, justify decision-making processes, and address potential biases within the model. This lack of transparency

raises significant concerns with regards to the ethics behind a lot of applications, as well as issues with upholdning regulatory frameworks like the General Data Protection Regulation (GDPR) [31]. The new EU AI Act, launched in 2024, is heavily focused on ensuring that AI systems are both trustworthy and comprehensible, and compliance requires a strong emphasis on both transparency and explainability [35].

XAI aims to provide solutions to this lack of transparency, allowing for more insight into the models and what their decisions are based on [13]. By unraveling the black-box one would approach robust, reliable and interpretable models - allowing for ethical and law-abiding applications of AI. Among the many proposed methods, the ones based on Shapley values stand out due to their solid theoretical foundation [15].

The remainder of this report is organised as follows; in Section 2 we provide a general introduction to the field of XAI, including a brief explanation around the practice of transparent models and some post-hoc methods. Section 3 further details Shapley values as mentioned in the previous section; encompassing an overview of Shapley values and approximations. Furthermore, in Section 4 we utilize simulations to explore different methods for approximating Shapley values. Finally the report concludes with Section 5, providing a discussion of the advantages and limitations of the different methods for Shapley value estimation, and of Shapley values in general.

2 Explainable AI

While understanding machine predictions has been an important notion ever since the emergence of rule-based learning in the 1950's, the term "explainable AI" was not coined until 2002 [7]. Since its introduction, the use of the word *explainable* in the context of AI has varied a lot, particularly in regard to the degree of explainability, dependent on the scope, the user and other similar factors [17]. For the purposes of this report the focus will lie on explainability in terms of feature importance (or feature attributions), i.e. determining the importance of each feature or parameter for a decision made by a model. This is a particularly important aspect of modern XAI, as determining feature attributions stands as one of the greater challenges in more complex algorithms and is fundamental to interpreting their decisions.

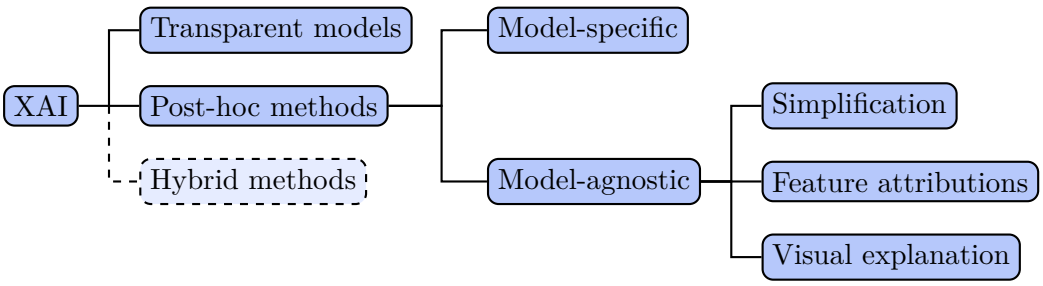

Fig. 1: Flowchart depicting the main categories of XAI, including subcategories.

Irregardless of the aforementioned variations in explainability there exists a range of possibilities regarding how one approaches XAI. The bulk of these can be split into two main categories, as shown in Fig. 1; attempting to make the models themselves more *transparent*, or methods for explaining already-trained models (*post-hoc*) [4]. Additionally some opt for hybrid solutions; including either a combination of models, or aspects and techinches from both realms. [4, 27].

2.1 Transparent models

When attempting to make AI models more transparent and explainable, many follow the principle of "less is more"; mitigating the issue of explainability by using simpler models that are inherently transparent. This includes methods like linear and logistic regression - which require little to no interpetation in terms of explainability, and are commonly used in applications of statistical learning [4]. Among other popular transparent models we find decision trees, K-nearest neighbours, general additive models, and Bayesian models.

However, in this approach to transparency there's a trade-off in model complexity. Already in the choice of these models one is opting away from the possibilites and precision of more complex models like deep neural networks. In addition to this, there are limitations to the amount of features, depth of trees, number of neighbours, etc, that can be allowed while maintaining the inherent transperency of the models [4]. While this proposes no issue for some applications, it is far from a satisfactory solution to the black-box issue as a whole. For many applications, like analysis of pictures and videos, simpler methods do not provide satisfactory perfomance, making it more desirable to find a way to explain the black-box models instead [28, 36].

2.2 Post-hoc methods

In the applications that necessitate more complex models, post-hoc methods are the obvious choice [4]. These are applied after a model has already been trained, aiming to interpret and explain what influences the model in its decision. Some of these methods are model-specfic, i.e. specialized for particular models like trees, support vector machines, and different variations of neural networks [3, 5, 6, 12, 21, 33, 37].

In addition to model-specific approaches as mentioned above, some methods are more generally applicable; called model-agnostic methods. This includes techniques for simplification, [24, 30, 34], visual explanations, [8, 9, 16], and determining feature attributions [10, 11, 20, 23]. Among the latter category we find Shapley values, arguably the post-hoc method with the strongest theoretical foundation [15].

3 Shapley Values

Shapley values were introduced in 1953 by Lloyd Shapley in the context of cooperative game theory [32]. They were originally proposed as a method for fair allocation of the total gain (or loss) among a group of players collaborating in a game. In short, the Shapley value quantifies each player's contribution by measuring the change in the overall outcome when they join every possible group of other players, then averaging these changes. Thus, the Shapley value measures a player's average marginal contribution across all possible player coalitions, shown in Fig.2. As detailed in the next section, the Shapley value is the only solution for determining player contributions that satisfies the fundamental properties widely accepted to define a fair distribution [32].

3.1 Mathematical formulation

Consider a game with M players, the set of players denoted $\mathcal{M}$, and let S denote a subset of the players (a coalition) of size $|S|$. Furthermore, let $v(S)$ provide the return of a game played with players S. In a game setting, this would correspond to the reward won or the

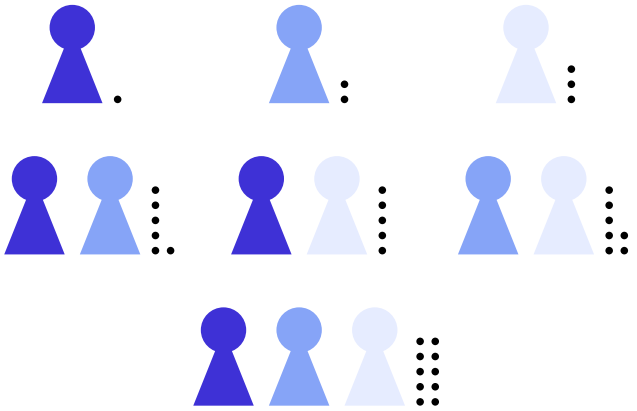

Fig. 2: Illustration of the distribution of contributions across coalitions in a cooperative game. The dots on the right-hand side of each coalition S visualizing the contribution function $v(S)$.

number of points gained, and $v(S)$ is called the characteristic function or *contribution function*. The marginal Shapley-value for player i is computed as follows:

$$\phi_i(v) = \sum_{S \subseteq \mathcal{M} \setminus \{i\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(v(S \cup \{i\}) - v(S) \right). \tag{1}$$

Roughly speaking, one is computing a weighted average of the difference in return with and without the player's contribution for all possible subsets. The average is weighted according to possible combinations of players in the current subset, compared to combinations from the full set of players.

As briefly mentioned, Shapley values satisfy the fundamental properties widely accepted to define a fair distribution [32]; symmetry, additivity, efficiency and the dummy player property.

Symmetry ensures that two players j and k that contribute equally across all subsets, are acredited equally;

$$v(S \cup \{j\}) = v(S \cup \{k\}) \quad \forall S \subseteq \mathcal{M} \setminus \{j, k\} \quad \Rightarrow \quad \phi_j = \phi_k. \tag{2}$$

Additivity, or linearity, ensures that a linear combination of n coalition games $\{v_1, \ldots, v_n\}$ has distributed gains that correspond to the gains of the respective games. In other words, the Shapley value $\phi_j(v)$ of a feature in the combined game v is a linear combination of its Shapley values $\phi_j(v_k)$ in the $k = 1, \ldots, n$ sub-games;

$$v(S) = \sum_{k=1}^{n} c_k v_k(S) \quad \forall S \subseteq \mathcal{M} \quad \Rightarrow \quad \phi_j(v) = \sum_{k=1}^{n} c_k \phi_j(v_k). \tag{3}$$

Efficiency ensures that the total value generated by a coalition is fully distributed among the M players such that no contribution is left unaccounted for, or counted more than once;

$$\sum_{j=1}^{M} \phi_j = v(\mathcal{M}) - v(\emptyset). \tag{4}$$

Lastly, **the dummy player property** ensures that a player j that does not contribute to any coalition, recieves no accreditation;

$$v(S) = v(S \cup \{j\}) \quad \forall S \subseteq \mathcal{M} \setminus \{j\} \quad \Rightarrow \quad \phi_j = 0. \tag{5}$$

As these properties are satsfied, Shapley values uphold a fair distribution - again, making it the only solution to do so. The proofs that Shapley values satisfy these properties are mostly trivial, and can be found in Appendix B.

3.2 Shapley values in XAI

In the context of XAI and feature importance, Shapley values are used to explain a models individual prediction for a specific feature vector $\boldsymbol{x} = \boldsymbol{x}^*$. Now the M-dimensional feature-vector $\boldsymbol{x}$ corresponds to the M players, and model predictions $\hat{y} = f(\boldsymbol{x})$ correspond to game-return.

The choice of the contribution function is however seldom obvious, and often poses a difficult task. In their method SHAP (SHapley Additive exPlanation), Lundberg and Lee proposed the use of conditional probabilities, having $v(S) = E[f(\boldsymbol{x}) \mid \boldsymbol{x}_S]$ [22]. Here $f(\boldsymbol{x})$ is the prediction of a model f on input $\boldsymbol{x}$, and $\boldsymbol{x}_S$ is a feature vector with only the known features included. Letting $\bar{S} = \mathcal{M} \setminus \mathcal{S}$ denote the set of unknown features, the conditional expectation can be written as

$$E\left[f(\boldsymbol{x}) \mid \boldsymbol{x}_S = \boldsymbol{x}_S^*\right] = E\left[f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S) \mid \boldsymbol{x}_S = \boldsymbol{x}_S^*\right] = \int f(\boldsymbol{x}_{\bar{S}}, \boldsymbol{x}_S^*) p(\boldsymbol{x}_{\bar{S}} \mid \boldsymbol{x}_S = \boldsymbol{x}_S^*) d\boldsymbol{x}_{\bar{S}}. \quad (6)$$

The integral computes the expected value of the model prediction over all possible values for the missing features, weighted by their conditional probabilities. In simple terms, we're computing what we expect the unkown feature values to be, based on the known values.

3.3 Approximation methods

Calculating the expectation in Equation (6) entails having knowledge of the underlying distribution in the data, which is rarely known. Additionally, even if the distribution is known the computation of the Shapley values is a highly costly affair, exponentially increasing as the number of features and the complexity of the model increases [25]. As time and computational efficiency are both valuable reasources, it's evidently of interest to find good approximations.

3.3.1 Independence assumption

One of the most common approaches for approximating Shapley values is to assume independence between features [1]. This assumption is the basis of many, if not most, of the existing approximation methods. With independent features, or the assumption thereof, the conditional distribution simplifies to the marginal:

$$p(\boldsymbol{x}_{\bar{S}} \mid \boldsymbol{x}_S = \boldsymbol{x}_S^*) \approx p(\boldsymbol{x}_{\bar{S}}).$$

This greatly eases computations of the expectation in Equation (6), as missing features can be marginalized independently of the observed ones. While assuming independence significantly reduces computational complexity, it might introduce bias when feature correlations are strong, potentially leading to misleading attributions.

3.3.2 Empirical conditional estimation

Instead of assuming independence, one can empirically estimate the conditional distribution from the data [1]. Methods such as nearest-neighbor sampling or local kernel density estimation approximate $p(\boldsymbol{x}_{\bar{S}} \mid \boldsymbol{x}_S = \boldsymbol{x}_S^*)$ by finding training samples close to $\boldsymbol{x}_S^*$. This approach preserves feature dependencies observed in the dataset, but can suffer from the curse of dimensionality in high-dimensional spaces since conditioning on many features might result in few similar observations, making accurate estimation difficult when the number of features grows large.

3.3.3 Parametric modeling (e.g., Gaussian, Copula models)

When assumptions about the underlying data structure are reasonable, parametric models can be fitted to capture feature dependencies [1]. For instance, assuming that the data follows a multivariate Gaussian distribution allows closed-form computations of conditional distributions. Alternatively, copula models offer greater flexibility by modeling the joint distribution as a combination of arbitrary marginal distributions and a separate dependence structure, represented by the copula function. More accurately, we can model a multivariate distribution $F(\boldsymbol{x})$ as

$$F(x_1, x_2, \ldots, x_m) = C(F_1(x_1), F_2(x_2), \ldots, F_m(x_m)) \tag{7}$$

for an appropriate copula function C, with marginal distributions F_m. Using a Gaussian copula allows us to use the same expressions for conditional distributions as a mutivariate Gaussian distribution, but other copulas may be used as well.

Parametric approaches can generalize well beyond the training data but may introduce model mismatch errors if the assumed distribution does not adequately capture the true data-generating process.

3.3.4 Generative models (e.g., VAE, VAEAC)

Generative models, such as Variational Autoencoders (VAEs), approximate the joint distribution of features, enabling conditional sampling by fixing observed inputs and generating plausible completions [19]. VAEs learn a low-dimensional latent representation $\boldsymbol{z}$ such that the full data vector $\boldsymbol{x}$ can be regenerated from it. Once trained, they let us complete missing features by conditioning on the known subset S:

$$p_\theta(\boldsymbol{x}_{\bar{S}} \mid \boldsymbol{x}_S) \;=\; \int p_\theta(\boldsymbol{x}_{\bar{S}} \mid \mathbf{z})\; q_\phi(\mathbf{z} \mid \boldsymbol{x}_S)\, d\mathbf{z}, \tag{8}$$

where $q_\phi(\mathbf{z} \mid \boldsymbol{x}_S)$ is the encoder, and $p_\theta(\boldsymbol{x}_{\bar{S}} \mid \mathbf{z})$ is the decoder. q_ϕ infers a latent code consistent with the observed features, and p_θ outputs plausible values for the missing features $\boldsymbol{x}_{\bar{S}}$. Architectures such as Variational Autoencoders with Arbitrary Conditioning (VAEACs) modify the encoder–decoder pair so that conditional sampling is numerically stable [18, 26]. However, numerical stability alone does not ensure accurate results. Because the synthetic completions are drawn entirely from these learned conditional distributions, any bias or noise in the training data can skew the estimates $\boldsymbol{x}_{\bar{S}}$ and in turn produce poorly approximated shapley values.

4 Exploring Shapley Value methods using simulations

In this section, the performance of different methods for Shapley value estimation is assessed on simulated data. The simulation study setup is inspirered by the study conducted by Berge Olsen et al. [25], and uses the `shapr`-package [29] developed by Norsk Regnesentral to run the different estimation methods. The simulations are restricted to linear models to ensure that the true Shapley values can be computed analytically or approximated to arbitrary precision. By using linear models, a controlled setting is provided in which the "true" contribution of each feature to the prediction can be determined.

4.1 Methodology

4.1.1 Simulation study overview

In this simulation study, a variety of settings are considered to evaluate the performance of the explanation methods across different scenarios. Specifically, the simulations consider simulated data sets with different numbers of features $M \in \{3, 4, 5\}$, levels of feature dependence $\rho \in \{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$, and levels of signal strength (the features' effect on the response). The data generation procedure, model training and Shapley value estimation will be detailed later in this section, but the general setup is as follows; after a data set has been generated, it is split into a training and test set, and a linear regression model is trained on the former. All simulations in this report are run with $N_1 = 1000$ training data samples and $N_2 = 500$ test data samples, unless otherwise specified. Each Shapley value estimation method is subsequently used to explain the predictions obtained when applying the trained model to the test data, and the accuracy of the respective methods is measured. To measure the accuracy, the mean absolute error (MAE) is used, given by

$$\text{MAE}_q = \frac{1}{N_2} \sum_{i=1}^{N_2} \frac{1}{M} \sum_{j=1}^{M} \left| \hat{\phi}_{q,j}(\boldsymbol{x}_i) - \phi_j(\boldsymbol{x}_i) \right|, \tag{9}$$

for each method q. Here $\hat{\phi}_{q,j}$ is the Shapley value estimate produced by method q for feature j on observation i, and $\phi_j(\boldsymbol{x}_i)$ is correspondingly the true Shapley value for the same feature and observation. For each setting, 50 replicates are used and the performance metrics are averaged over these runs. The simulation setup for one replicate, for a given number of features M and coefficient set is described in Algorithm 1.

Algorithm 1: Simulation study

forall $\rho \in \{0, 0.1, 0.3, 0.5, 0.7, 0.9\}$ **do**
- $\mathbf{X}_{\text{train}}$, $\mathbf{y}_{\text{train}} \leftarrow (N_1 \times M)$ matrix of generated data, $(N_1 \times 1)$ response vector
- $\mathbf{X}_{\text{test}}$, $\mathbf{y}_{test} \leftarrow (N_2 \times M)$ matrix of generated data, $(N_2 \times 1)$ response vector
- $\hat{f} \leftarrow$ linear regression model trained on $\mathbf{y}_{\text{train}} \sim \mathbf{X}_{\text{train}}$;
- $\phi(\hat{f}, \mathbf{X}_{\text{test}}) \leftarrow (N_2 \times M)$ matrix of true Shapley values
- **forall** $q \in Q$ **do**
  - $\hat{\phi}_q(\hat{f}, \mathbf{X}_{\text{test}}) \leftarrow (N_2 \times M)$ matrix of Shapley value estimates by q
  - $\text{MAE}_q \leftarrow$ mean absolute error for q, Eq. (9).

Here Q represents the set of chosen methods, which were $Q = \{$`independence`, `empirical`, `gaussian`, `copula`$\}$ for the simulations presented in this study. All of the methods were run with default values for optional paramters, notably including 2^M (the total number of existing coalitions for M features), as the number of coalitions to be used in computations. The prediction value for unseen data was set to $\phi_0 = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} y_i$, where y_i, $i \in 1, \dots, N$ are the values of the respons in the training data.

To account for the cost of computation when comparing the methods, the CPU time used to estimate the matrix $\hat{\phi}_q$ for each of the estimation methods is measured. In addition seperate simulations were timed for $M = \{6, 7, 8, 9, 10\}$ features, with $\rho = 0.5$ and strong signal strength. The extra features were achived with an extension of $\boldsymbol{\beta}_2$ of the same magnitude; $\boldsymbol{\beta} = \{0.3, 0.1, 0.3, -0.5, 0.7, 0.2, 0.7, 0.1, 0.3, 0.4, 0.2\}$. Time measurements were also averaged over 50 replicates.

4.1.2 Data generation

Given the level of feature dependence ρ, signal strength (intercept and coefficent set), number of observations N, and number of features M, a $(N \times M)$ data matrix $\mathbf{X}$ and $(N \times 1)$ response vector $\mathbf{y}$ is generated. The samples in $\boldsymbol{X}$ are drawn from a multivariate Gaussian distribution $\mathcal{N}_M(\mathbf{0}, \boldsymbol{\Sigma})$ with correlations $\rho \in [0, 1]$, corresponding to the covariance matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho & \dots & \rho \\ \rho & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \rho \\ \rho & \dots & \rho & 1 \end{bmatrix} \in \mathbb{R}^{M \times M}.$$

For a given covariance matrix $\boldsymbol{\Sigma}$, data can be sampled from $\mathcal{N}_M(\mathbf{0}, \boldsymbol{\Sigma})$ using the function `rmvnorm` in `R`.

Next, the responses $\boldsymbol{y}$ are sampled according to $y_i = f(\boldsymbol{x}_i) + \epsilon_i$, with $\epsilon_i \sim \mathcal{N}(0, 1)$, $i = 1, \dots, N$. The linear function f is decided by the set of intercept and coefficients, $\{\beta_0, \boldsymbol{\beta}\}$, yielding $f(\boldsymbol{x}_i) = \beta_0 + \sum_{j=1}^{M} \beta_j X_{i,j}$. For each combination of the number of features M and feature dependence level ρ, two different sets of coefficients and intercept are considered; a "lower-impact" setting with weaker signal $\boldsymbol{\beta}^1 \subseteq \{0.3, 0.1, 0.3, -0.5, 0.7, 0.2\}$, and a "higher-impact" with stronger signal $\boldsymbol{\beta}^2 \subseteq \{3, 1, 3, -5, 7, 2\}$. For M features, the first M coefficients of the respective sets are used in addtion to the intercept.

4.1.3 Computation of true Shapley values

Computations of the true Shapley values are done according to Eq. 1 and 6, split into two scenarios for the conditional expectations; data with independent features and data with dependent features. The two computations differ only in how unknown features are estimated when other features are known.

For independent data, unkown features are drawn from a mariginal distribution, unrelated to other known features. Here we simply use the mean of feature values in the training data, i.e. $\mu_m = \frac{1}{N} \sum_{i=0}^{N} X_{i,m}$ for each feature $m \in M$.

$$v_{\rho=0}(S, \mathbf{x}^*) = \begin{cases} \hat{\beta}_0 + \sum_{j=1}^{M} \hat{\beta}_j \mu_j, & \text{if } |S| = 0 \\ \hat{\beta}_0 + \sum_{j \in S} \hat{\beta}_j x_j^* + \sum_{j \notin S} \hat{\beta}_j \mu_j, & \text{otherwise} \end{cases} \tag{10}$$

$\hat{\beta}_0$ and $\hat{\beta}_j$ denote the intercept and coefficients taken from the trained model $\hat{f}$. S is a coalition of features, while $|S|$ is it's size. $\mathbf{x}^*$ is the observation for which the Shapley value is calculated.

These same μ's are used for dependent data in the case of no known features. However, when some are known, i.e. $|S| > 0$, we calculate the conditional mean $\boldsymbol{\mu}_{\boldsymbol{x}_{|}\boldsymbol{x}^*_S}$ in stead based on the provided ρ;

$$\boldsymbol{\mu}_{\boldsymbol{x}_{|}\boldsymbol{x}^*_S} = \boldsymbol{\mu} \; + \; \boldsymbol{\Sigma}_{\bar{S},S}\,\boldsymbol{\Sigma}^{-1}_{S,S}(\boldsymbol{x}_S - \boldsymbol{\mu}_S). \tag{11}$$

$\boldsymbol{\Sigma}_{\bar{S},S}$ and $\boldsymbol{\Sigma}_{S,S}$ above stem from the decomposition of $\boldsymbol{\Sigma}$ as

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{S,S} & \boldsymbol{\Sigma}_{S,\bar{S}} \\ \boldsymbol{\Sigma}_{\bar{S},S} & \boldsymbol{\Sigma}_{\bar{S},\bar{S}} \end{pmatrix}.$$

With Eq. 11 we get the following contribution function for data with dependency;

$$v_{\rho>0}(S, \boldsymbol{x}^*) = \begin{cases} \hat{\beta}_0 + \sum_{j=1}^{M} \hat{\beta}_j \mu_j, & \text{if } |S| = 0 \\ \hat{\beta}_0 + \sum_{j \in S} \hat{\beta}_j x^*_j + \sum_{j \notin S} \hat{\beta}_j \mu_{\boldsymbol{x}_{|}\boldsymbol{x}^*_S, j} & \text{otherwise} \end{cases} \tag{12}$$

4.1.4 Elimination of the VAE-AC method

During initial simulations, the `vaeac`-method exhibited significantly higher computational demands, taking on average over 40 times longer to run than the average of the other methods (see Table 1, Appendix C). A full test of the simulation with 30 runs excluding the `vaeac`-method required roughly 4 hours, thus including `vaeac` would have rendered the expanded 50-run simulation computationally impractical.

The `vaeac`-method offers numerous parameters to tailor the training process, including number of epochs, depth and width of the network etc. These were all run with default values, and given the vast number of possible combinations, it is possible that a certain set of parameters could provide good estimations while shortening computation time. However, this would add an additional and time-consuming layer of tuning, without necessarily yielding satisfactory results. Due to the limited computational resources avaible, we opted to exclude the `vaeac`-method from the final set of simulations.

4.2 Results

Fig. 3 shows the results of a simulation run with $M = 3$ features and the low-impact $\boldsymbol{\beta}$-set for different levels of feature dependence. As one would expect, the `independence`-method performs close to perfect for data with true independent structure, and increasingly worse as the level of dependence increases. Both `copula` and `gaussian` are seemingly robust to varying levels of dependence; this is consistent with expectations, as the underlying data follows a true gaussian distribution. The `empirical`-method exhibits the worst results for lower dependence levels, only surpassed by `independence` for medium and high levels. However, as with `copula` and `gaussian`, the method appears to be robust to varying levels of dependence.

Fig. 4 shows the results of a simulation run with 5 features and the low-impact $\boldsymbol{\beta}$-set for different levels of feature dependence. The relative performance of the methods is similar to what it was in the setting with $M = 3$ low-impact features.

Fig. 5 shows the results of a simulation run with 3 features and the high-impact $\boldsymbol{\beta}$-set for different levels of feature dependence. The relative performances of the methods are

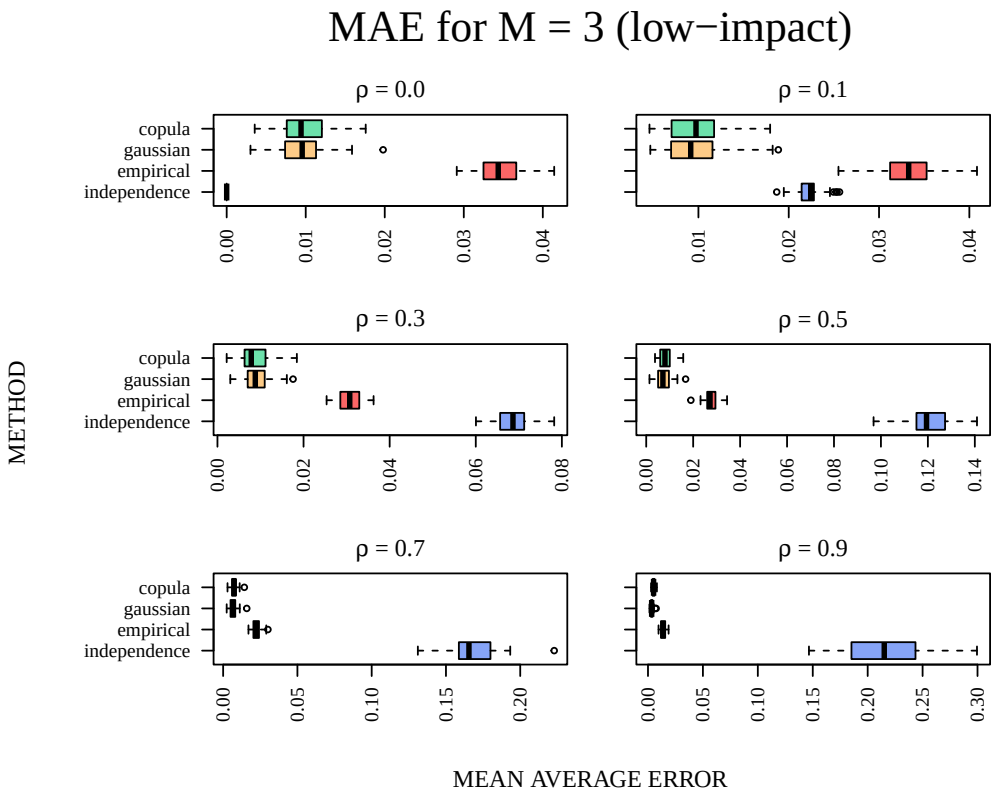

Fig. 3: Mean average error per method for `beta1_m3` with 50 replicates.

similar as in the simulations with lower impact features. Compared to the simulation with $M = 3$ and lower impact features, the median error values are almost exactly ten times larger. Because the two simulations differ only by a factor of ten in β, this behaviour is reasonable; the magnitude of the response and hence the errors will increase similarly.

Fig. 6 shows the results of a simulation run with five features and the high-impact $\boldsymbol{\beta}$ set for different levels of feature dependence. Again, we see that while the performance of the `independence` method deteriorates as the level of feature dependence increases, the other methods demonstrate more robustness. The `Gaussian` and `copula` methods have, not unexpectedly, the overall best performance for the Gaussian simulated data.

As mentioned in Sec. 4.1.1, we also ran simulations with $M = 4$ features for both $\boldsymbol{\beta}$-sets. However, as the results did not provide additional insight beyond what is already shown, they are not included here (see Appendix C).

Fig. 7 shows how the time of computation changes with an increasing number of features. The `empirical` method displays the slowest computations of the four, using several tenfolds longer then the remaining methods for all number of features. Reasonably `gaussian` and `copula` use roughly the same time relative to the other methods, with the former slightly outperforming the latter. Across all number of features the `gaussian` method yields the lowest computational time. For the higher numbers of features `independence` starts to flatten out compared to the others, leaving it within a few tenfolds of `gaussian` and `copula`. This could indicate that it might outperform these methods for even higher numbers of features.

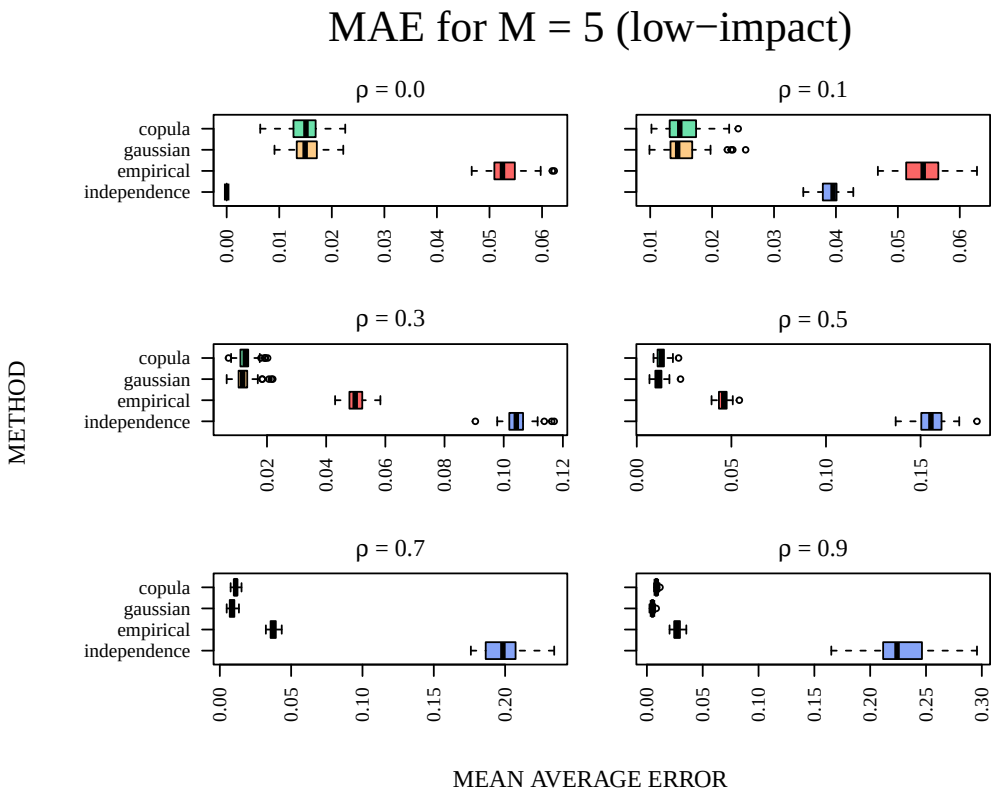

Fig. 4: Mean average error per method for `beta1_m5` with 50 replicates.

5 Discussion

This report set out to explore the practical applications of Shapley values in XAI; exploring accuracy and limitations, and the performance of different approximation methods. Shapley values provide a theoretically sound framework for discussing feature importance in AI-models. Rooted in cooperative game theory, they ensure a fair distribution through the properties efficiency, symmetry, additivity and the so called dummy player property. This makes them particularly useful in areas where it's important to understand how models make decisions, such as in healthcare, finance, or public policy. While they can be computationally expensive, Shapley values still provide a reliable and well-founded way to explain complex model behavior, and through estimation methods the exact solutions can be approximated for a lower computational cost.

In simulations the `empirical` method, exhibited the worst results in terms of computational time, as well as being outperformed by the other methods in comparisons to the true Shapley values. The method approximates unkown values by determining the closest points in the training data, and will unavoidably suffer from the curse of dimensionality yielding increasingly poor accuracy for higher number of features. Both `gaussian` and `copula` exhibited robust results across varying number of features and dependence levels, performing the overall best of the four methods in time and accuracy - with the former method slightly outperfoming the latter. As the `gaussian` method assumes a Gaussian distribution underlying the data, and the `copula` method uses a Gaussian copula, they both benefit `gaussian` from the advantage of a true Gaussian distribution in the data. Lastly the `independence` method, which assumes independent

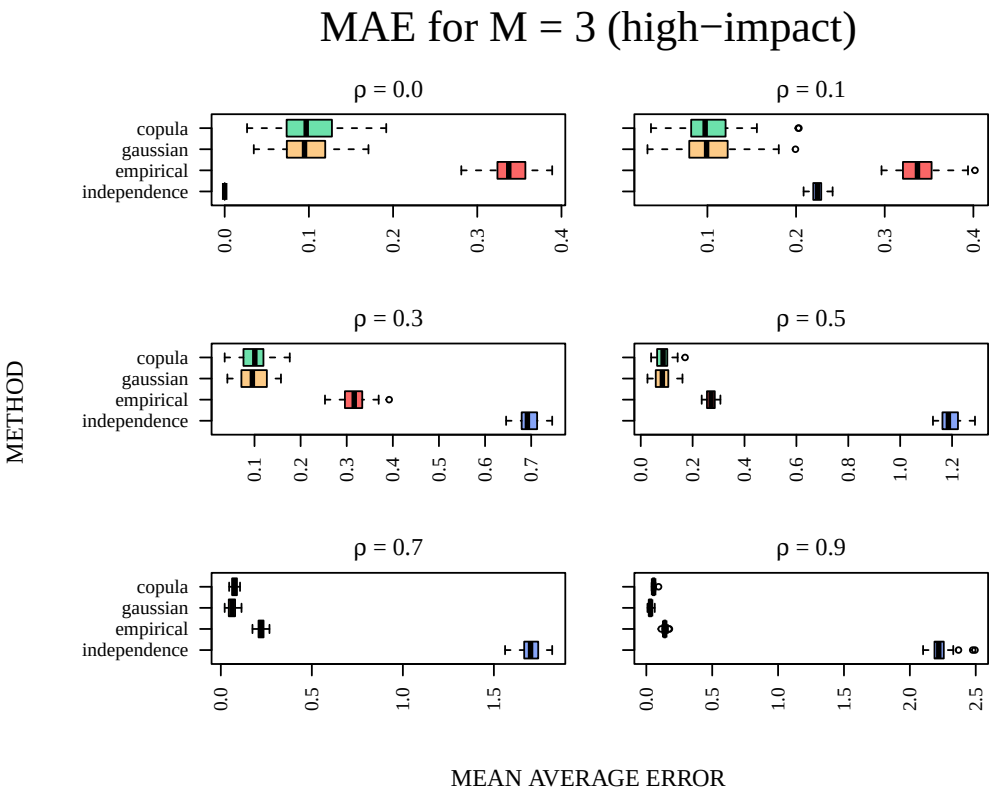

Fig. 5: Mean average error per method for `beta2_m3` with 50 replicates.

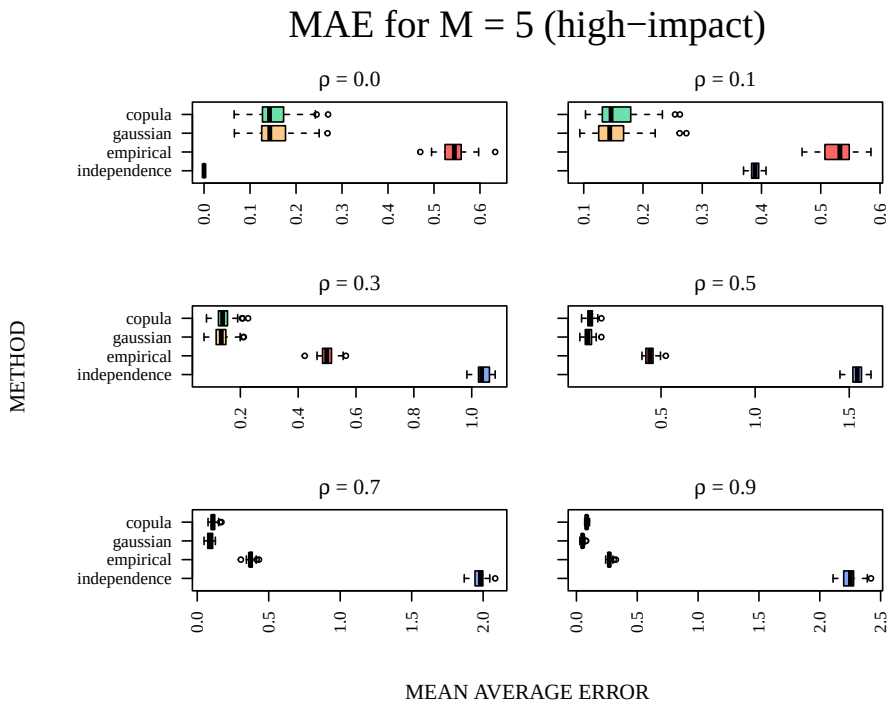

Fig. 6: Mean average error per method for `beta2_m5` with 50 replicates.

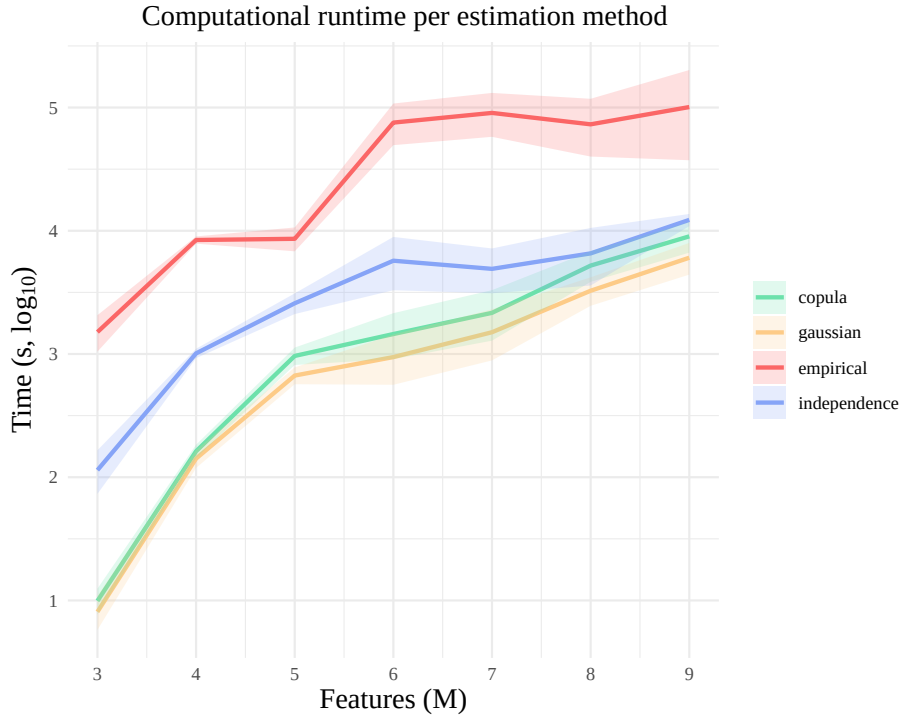

Fig. 7: Average runtime on a log-scale for different numbers of features M, for each method, measured according to Sec. 4.1.1. The results are averaged over 50 replicates. The means are represented in bold lines, with standard deviation shown as the shaded area around the mean. Data generated with $\rho = 0.5$ and the low-impact β-set.

features, predictibly displayed the worst accuracy of the methods for higher levels of dependency, but provided close to perfect results with truly independent features. In terms of computational time, `independence` displays a midrange perfomance; being slightly outperfomed by the parametric methods `gaussian` and `copula`, but by far outperforming `empirical`.

These findings emphasize the need to be concious of what assumptions underles different methods for approximation. Making sound and reasonable assumptions, and taking the necessery steps to asses the validity of them, is crucial for chosing the appropriate methods. Many of the common approximation methods assume an independent distribution in the data, but real-life data seldom follows a true independent structure. Thus its of great importance to move beyond this assumption and develop methods taking into account higher levels of dependence. In particular, the simulation underlined the necessity of good approximations to Shapley Values that can handle high-dimensional settings. XAI is a field with a great need for improvements to enable large-scale, accurate approximations while balancing computational demands.

On a similar note, the findings highlight a clear limitation of the simulation study in this report; both the data-generation process and the learner used to make predictions follow a linear model. Evaluating the performance of the Shapley value estimation methods on more complex data distributions and machine learning algorithms would better reflect real-world applications, and give a more comprehensive understanding of the respective merits of the estimation methods. However, computing the true Shapley values in those settings would require much more extensive computations and might not even be feasible in practice, placing it beyond the scope of this project.

For further research it would be interesting to upscale the study, including a broader range of methods, higher numbers of features and exploring the explainability of more complex models. While more comprehensive simulation studies are challenging to conduct, this is key to the further development of methods for black-box prediction explanation that are both robust to different settings and scalable to real-life high-dimensional problems.

References

[1] Aas, K., Jullum, M. and Løland, A. 'Explaining Individual Predictions When Features Are Dependent: More Accurate Approximations to Shapley Values'. In: *Artificial Intelligence* 298 (2021), p. 103502. DOI: 10.1016/j.artint.2021.103502.

[2] Araujo, T., Helberger, N., Kruikemeier, S. and Vreese, C. H. de. 'In AI we trust? Perceptions about automated decision-making by artificial intelligence'. In: *AI & SOCIETY* 35.3 (2020), pp. 611–623. DOI: 10.1007/s00146-019-00931-w.

[3] Barakat, N. H. and Bradley, A. P. 'Rule extraction from support vector machines: A sequential covering approach'. In: *IEEE Transactions on Knowledge and Data Engineering* 19.6 (2007), pp. 729–741.

[4] Barredo Arrieta, A. et al. 'Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI'. In: *Information Fusion* 58 (2020), pp. 82–115. DOI: https://doi.org/10.1016/j.inffus.2019.12.012.

[5] Bau, D., Zhou, B., Khosla, A., Oliva, A. and Torralba, A. 'Network dissection: Quantifying interpretability of deep visual representations'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6541–6549.

[6] Breiman, L. *Classification and regression trees*. Routledge, 2017.

[7] Christian Meske Enrico Bunde, J. S. and Gersch, M. 'Explainable Artificial Intelligence: Objectives, Stakeholders, and Future Research Opportunities'. In: *Information Systems Management* 39.1 (2022), pp. 53–63. DOI: 10.1080/10580530.2020.1849465. eprint: https://doi.org/10.1080/10580530.2020.1849465.

[8] Cortez, P. and Embrechts, M. J. 'Opening black box data mining models using sensitivity analysis'. In: *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE. 2011, pp. 341–348.

[9] Cortez, P. and Embrechts, M. J. 'Using sensitivity analysis and visualization techniques to open black box data mining models'. In: *Information Sciences* 225 (2013), pp. 1–17.

[10] Dabkowski, P. and Gal, Y. 'Real time image saliency for black box classifiers'. In: *Advances in neural information processing systems* 30 (2017).

[11] Datta, A., Sen, S. and Zick, Y. 'Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems'. In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 598–617.

[12] Deng, H. 'Interpreting tree ensembles with intrees'. In: *International Journal of Data Science and Analytics* 7.4 (2019), pp. 277–287.

[13] Doshi-Velez, F. and Kim, B. 'Towards A Rigorous Science of Interpretable Machine Learning'. In: *arXiv preprint arXiv:1702.08608* (2017).

[14] Duan, Y., Edwards, J. S. and Dwivedi, Y. K. 'Artificial intelligence for decision making in the era of Big Data – evolution, challenges and research agenda'. In: *International Journal of Information Management* 48 (2019), pp. 63–71. DOI: https://doi.org/10.1016/j.ijinfomgt.2019.01.021.

[15] Fryer, D., Strümke, I. and Nguyen, H. *Shapley values for feature selection: The good, the bad, and the axioms*. 2021. arXiv: 2102.10936 [cs.LG].

[16] Goldstein, A., Kapelner, A., Bleich, J. and Pitkin, E. 'Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation'. In: *journal of Computational and Graphical Statistics* 24.1 (2015), pp. 44–65.

[17] Guidotti, R. et al. 'A Survey of Methods for Explaining Black Box Models'. In: *ACM Comput. Surv.* 51.5 (Aug. 2018). DOI: 10.1145/3236009.

[18] Ivanov, O., Figurnov, M. and Vetrov, D. 'Variational Autoencoder with Arbitrary Conditioning'. In: *arXiv preprint arXiv:1806.02382* (2018). arXiv: 1806.02382 [stat.ML].

[19] Kingma, D. P. and Welling, M. 'Auto-Encoding Variational Bayes'. In: *arXiv preprint arXiv:1312.6114* (2013). arXiv: 1312.6114 [stat.ML].

[20] Koh, P. W. and Liang, P. 'Understanding black-box predictions via influence functions'. In: *International conference on machine learning*. PMLR. 2017, pp. 1885–1894.

[21] Krakovna, V. and Doshi-Velez, F. 'Increasing the interpretability of recurrent neural networks using hidden markov models'. In: *arXiv preprint arXiv:1606.05320* (2016).

[22] Lundberg, S. M. and Lee, S. 'A Unified Approach to Interpreting Model Predictions'. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. 2017, pp. 4765–4774.

[23] Martens, D. and Provost, F. 'Explaining data-driven document classifications'. In: *MIS quarterly* 38.1 (2014), pp. 73–100.

[24] Mishra, S., Sturm, B. L. and Dixon, S. 'Local interpretable model-agnostic explanations for music content analysis.' In: *ISMIR*. Vol. 53. 2017, pp. 537–543.

[25] Olsen, L. H. B., Glad, I. K., Jullum, M. and Aas, K. 'A Comparative Study of Methods for Estimating Conditional Shapley Values and When to Use Them'. In: *Data Mining and Knowledge Discovery* (2024). DOI: 10.1007/s10618-024-01016-z.

[26] Olsen, L. H. B., Glad, I. K., Jullum, M. and Aas, K. 'Using Shapley Values and Variational Autoencoders to Explain Predictive Models with Dependent Mixed Features'. In: *Journal of Machine Learning Research* 23 (2022). Article 213, pp. 1–51.

[27] Quesada, G., Jesus, M. J. del and González, P. 'Explainable Artificial Intelligence: An Overview on Hybrid Models'. In: *Proceedings of the MAI-XAI'24 Workshop*. 2024.

[28] Rai, A. 'Explainable AI: from black box to glass box'. In: *Journal of the Academy of Marketing Science* 48 (2020), pp. 137–141. DOI: 10.1007/s11747-019-00710-5.

[29] Regnesentral, N. *SHAP for R: General usage*. Accessed: 2025-02-17. 2025.

[30] Ribeiro, M. T., Singh, S. and Guestrin, C. '" 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*. 2016, pp. 1135–1144.

[31] Sartor, G. and Lagioia, F. *The Impact of the General Data Protection Regulation (GDPR) on Artificial Intelligence*. European Parliament, 2020. DOI: 10.2861/293.

[32] Shapley, L. S. 'A value for n-person games'. In: *Contribution to the Theory of Games* 2 (1953).

[33] Springenberg, J. T., Dosovitskiy, A., Brox, T. and Riedmiller, M. 'Striving for simplicity: The all convolutional net'. In: *arXiv preprint arXiv:1412.6806* (2014).

[34] Tan, S., Caruana, R., Hooker, G. and Lou, Y. 'Distill-and-compare: Auditing black-box models using transparent model distillation'. In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. 2018, pp. 303–310.

[35] *The European Union's AI Act (2024)*. https://digital-strategy.ec.europa.eu/en/policies/regulatory-framework-ai. Accessed: 2025-02-18.

[36] Velden, B. H. M. van der, Kuijf, H. J., Gilhuijs, K. G. A. and Viergever, M. A. 'Explainable Artificial Intelligence (XAI) in Deep Learning-Based Medical Image Analysis'. In: *Medical Image Analysis* 79 (2022), p. 102470. DOI: 10.1016/j.media.2022.102470.

[37] Zeiler, M. D., Taylor, G. W. and Fergus, R. 'Adaptive deconvolutional networks for mid and high level feature learning'. In: *2011 international conference on computer vision*. IEEE. 2011, pp. 2018–2025.

A Source code

The code used for this project is available at https://github.com/julienbj/XAI.

B Proof of properties

Symmetry:
Let $v(S \cup \{j\}) = v(S \cup \{k\})$, $\forall S \subseteq \mathcal{M} \setminus \{i, k\}$. Then;

$$\begin{aligned}\phi_j(v) &= \sum_{S \subseteq \mathcal{M} \setminus \{j,k\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(v(S \cup \{j\}) - v(S)\right) \\ &= \sum_{S \subseteq \mathcal{M} \setminus \{j,k\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(v(S \cup \{k\}) - v(S)\right) = \phi_k(v)\end{aligned}$$

□

Additivity:
Let $v(S) = \sum_{k=1}^{n} c_k v_k(S)$, $\forall S \subseteq \mathcal{M}$. Then;

$$\begin{aligned}\phi_j(v) &= \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(v(S \cup \{j\}) - v(S)\right) \\ &= \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(\sum_{k=1}^{n} c_k v_k(S \cup \{j\}) - \sum_{k=1}^{n} c_k v_k(S)\right) \\ &= \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(\sum_{k=1}^{n} c_k \left(v_k(S \cup \{j\}) - v_k(S)\right)\right) \\ &= \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \sum_{k=1}^{n} \frac{|S|!(M - |S| - 1)!}{M!} c_k \left(v_k(S \cup \{j\}) - v_k(S)\right) \\ &= \sum_{k=1}^{n} c_k \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(v_k(S \cup \{j\}) - v_k(S)\right) = \sum_{k=1}^{n} c_k \phi_j(v_k)\end{aligned}$$

□

Effiency:

$$\sum_{j=1}^{M} \phi_j = \sum_{j=1}^{M} \sum_{S \subseteq \mathcal{M} \setminus \{j\}} \frac{|S|!(M - |S| - 1)!}{M!} \left(v(S \cup \{j\}) - v(S)\right) \tag{13}$$

Because the sums are finite, we can interchange them and isolate the coefficient of each $v(T)$ to determine how many times it contributes across all permutations, and whether those contributions sum to zero or not.

Letting $T \subseteq \mathcal{M}$ and $t = |T|$, there are two ways that $v(T)$ can appear in (13); as $v(S \cup \{j\})$ when $S = T \setminus \{j\}$, and as $-v(S)$ when $S = T$. To find the contribution of all interior coalitions (not $\varnothing$ or $\mathcal{M}$) in the sum in (13), we can therefore write:

$$\text{coeff}(v(T)) \;=\; t\,\alpha_{t-1} \;-\; (M - t)\,\alpha_t, \quad \text{with} \quad \alpha_r = \frac{r!\,(M - r - 1)!}{M!}. \tag{14}$$

For $1 \leq t \leq M - 1$, substituting the α–expressions into the coeff$(v(T))$ from (14) yields

$$t\,\frac{(t-1)!\,(M-t)!}{M!} \;-\; (M - t)\,\frac{t!\,(M-t-1)!}{M!} \;=\; 0.$$

Hence every coalition that is not $\varnothing$ or $\mathcal{M}$ cancels out in (13).

For $T = \varnothing$ ($t = 0$) only the "$-\,v(S)$" part appears, once for each of the M players:

$$\text{coeff}\big(v(\varnothing)\big) \;=\; -M\,\alpha_0 \;=\; -1.$$

For $T = \mathcal{M}$ ($t = M$) only the "$+\,v(S \cup \{i\})$" part appears, again once for each player:

$$\text{coeff}\big(v(\mathcal{M})\big) \;=\; M\,\alpha_{M-1} \;=\; +1.$$

As all interior coalitions vanish, we're left with

$$\sum_{i=1}^{M} \phi_i(v) \;=\; v(\mathcal{M}) - v(\varnothing).$$

The dummy player property
Let $v(S) = v(S \cup \{j\})$, $\forall S \subseteq \mathcal{M} \setminus \{j\}$. Then;

$$\begin{aligned}\phi_j(v) &= \sum_{S \subseteq \mathcal{M}\setminus\{j\}} \frac{|S|!(M-|S|-1)!}{M!}\left(v(S \cup \{j\}) - v(S)\right) \\ &= \sum_{S \subseteq \mathcal{M}\setminus\{j\}} \frac{|S|!(M-|S|-1)!}{M!} \cdot 0 = 0\end{aligned}$$

□

C Additional tables and plots

Method	Average time
Independent	27.6s
Empirical	46.5s
Gaussian	15.3s
Copula	19.2s
VAE-AC	1216.1s

Table 1: Runtime for all 5 inital methods. Dataset with 5 features, generated with $\beta_0 = 5$ and $\beta = [1, 3, 1, 2, 0.5]$. Averaged over 10 runs and rounded to the first decimal.

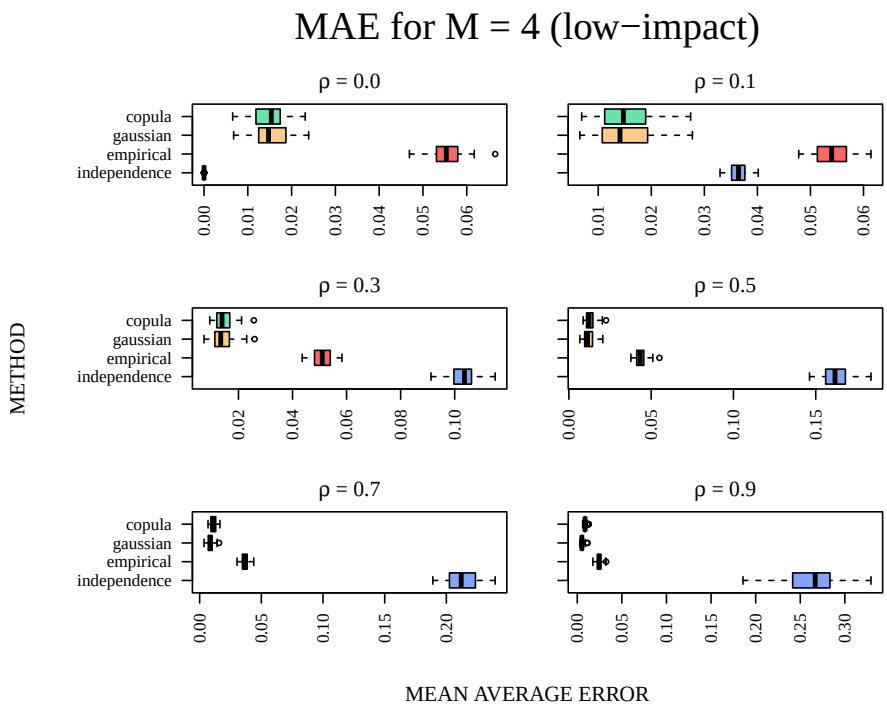

Fig. 8: Mean average error per method for data generated with 4 features and the low-impact β-set.

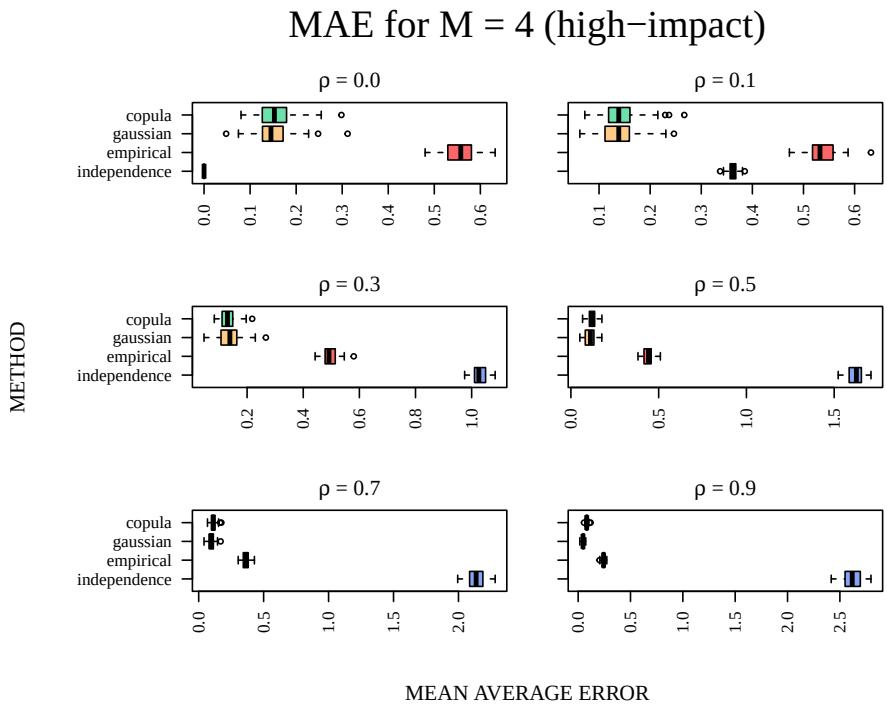

Fig. 9: Mean average error per method for data generated with 4 features and the high-impact β-set.