Research Article **Open Access**

Kjersti Aas*, Thomas Nagler, Martin Jullum, and Anders Løland

Explaining predictive models using Shapley values and non-parametric vine copulas

https://doi.org/10.1515/demo-2021-0103
Received February 8, 2021; accepted April 30, 2021

Abstract: In this paper the goal is to explain predictions from complex machine learning models. One method that has become very popular during the last few years is Shapley values. The original development of Shapley values for prediction explanation relied on the assumption that the features being described were independent. If the features in reality are dependent this may lead to incorrect explanations. Hence, there have recently been attempts of appropriately modelling/estimating the dependence between the features. Although the previously proposed methods clearly outperform the traditional approach assuming independence, they have their weaknesses. In this paper we propose two new approaches for modelling the dependence between the features. Both approaches are based on vine copulas, which are flexible tools for modelling multivariate non-Gaussian distributions able to characterise a wide range of complex dependencies. The performance of the proposed methods is evaluated on simulated data sets and a real data set. The experiments demonstrate that the vine copula approaches give more accurate approximations to the true Shapley values than their competitors.

Keywords: Prediction explanation, Shapley values, conditional distribution, vine copulas, non-parametric

MSC: 62G05, 62H05, 68T01, 91A12

1 Introduction

In many applications complex machine learning models like Gradient Boosting Machines, Random Forest and Deep Neural Networks are outperforming traditional regression models. It is often hard to understand why the machine learning models perform so well, and the last few years, a new line of research has emerged focusing on interpreting the predictions from these models. Existing work on explaining complex models may be divided into two main categories; global and local explanations. The former try to describe the model as whole, in terms of which variables/features influenced the general model the most. Local explanations, on the other hand, try to identify how the different input variables/features influenced a specific prediction/output from the model, and are often referred to as individual prediction explanation methods. Such explanations are particularly useful for complex models which behave rather different for different feature combinations, meaning that the global explanation is not representative for the local behavior.

In this paper, the focus is on local explanations. One method that has become very popular the last few years is Shapley values [2, 23, 41, 42]. This method, which is based on concepts from cooperative game theory, was originally invented for assigning payout to players depending on their contribution towards the total payout [24]. When interpreting machine learning models, the model features are the players and the prediction is the total payout, and the aim is to distribute the difference between the prediction and the average

***Corresponding Author: Kjersti Aas:** Norwegian Computing Center, E-mail: Kjersti.Aas@nr.no
Thomas Nagler: Leiden University, E-mail: t.w.nagler@math.leidenuniv.nl
Martin Jullum: Norwegian Computing Center, E-mail: Martin.Jullum@nr.no
Anders Løland: Norwegian Computing Center, E-mail: Anders.Loland@nr.no

prediction between the features in an optimal way. It can be shown that Shapley values is the only additive feature attribution method that adhers to certain important properties [23].

The original development of Shapley values for prediction explanation [23, 41, 42] relied on the assumption that the model features are independent. [2] showed that if there is a high degree of dependence among some or all the features, this may lead to severely inaccurate Shapley value estimates and incorrect explanations. In the same paper, the authors deal with this problem, proposing three different approaches appropriately modelling/estimating the dependence between the features; the Gaussian approach, the Gaussian copula approach, and the empirical approach. Although all three methods clearly outperform the traditional approach assuming independence, they have their weaknesses. The Gaussian approach assumes that features are multivariate Gaussian distributed, while the Gaussian copula approach represents the marginal distributions of the features with their empirical margins and model the dependence structure by a Gaussian copula [18]. Hence, these approaches will work well if respectively the distribution or the dependence structure of the features is Gaussian. The empirical approach is inspired by the kernel estimator. Like most other non-parametric density estimation approaches, this method suffers from the curse of dimensionality. It would therefore require a large data set to be accurate in problems with many features.

In this paper, we propose two alternative approaches to estimate Shapley values. In both approaches, the multivariate joint density function of the features is represented by a vine copula [18], but they differ in the way the Shapley contribution function is evaluated. A vine copula is a multivariate copula that is constructed from a set of bivariate ones, so-called *pair-copulas*. All of these bivariate copulas may be selected completely freely, meaning that vine copulas are able to characterise a wide range of complex dependencies. Hence, the new approaches are expected to outperform the existing ones in cases where the feature distribution is far from the Gaussian.

The main part of the methodology proposed in this paper may be used for many other applications than computing Shapley values. It may e.g. be regarded as a contribution to the field of non-parametric conditional density estimation. It should further be noted that not even a linear regression model is easily interpretable if the explanatory variables are dependent, see e.g. [13, 19]. Shapley values have been used for assessing global feature importance in such models, by partitioning the R^2 quantity among the features in a way that takes the dependence into account [22, 29, 38].

The rest of the paper is organized as follows. We begin by explaining the fundamentals of the Shapley value framework in an explanation setting in Section 2, while Section 3 reviews some of of the previously proposed Shapley methods for prediction explanation. In Section 4 we introduce the two new methods for computing Shapley values based on vine copulas. Section 5 presents various simulation studies that demonstrate that our method works in a variety of settings, while Section 6 gives a real data example. Finally, in Section 7, we conclude.

2 Shapley values

2.1 Shapley values in game theory

Suppose we are in a cooperative game setting with M players, $j = 1, \ldots, M$, trying to maximize a payoff. Let $\mathcal{M}$ be the set of all players and $\mathcal{S}$ any subset of $\mathcal{M}$. Then the Shapley value [36] for the jth player is defined as

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

Here, $v(\mathcal{S})$ is the contribution function which maps subsets of players to real numbers representing the worth or contribution of the group $\mathcal{S}$ and $|\mathcal{S}|$ is the number of players in subset $\mathcal{S}$.

In the game theory sense, each player receives ϕ_j as their payout. From the formula, we see that this payout is just a weighted sum of the player's marginal contributions to each group $\mathcal{S}$. Lloyd Shapley proved

that distributing the total gains of the game in this way is 'fair' in the sense that it obeys certain important axioms [36].

2.2 Shapley values for prediction explanation

In a machine learning setting, imagine a scenario where we have M features, $\boldsymbol{x} = (x_1, \ldots, x_M)$ and a univariate response y, and have fitted the model $g(\boldsymbol{x})$ which is supposed to predict y. We now use this model to predict the response for a test observation $\boldsymbol{x}^\star$ and want to know how the different features $x_1, \ldots, x_M$ influenced the prediction $g(\boldsymbol{x}^\star)$. The papers [23, 41, 42] suggest doing this with Shapley values where the predictive model replaces the cooperative game and the features replace the players. The prediction $g(\boldsymbol{x}^\star)$ is decomposed as follows:

$$g(\boldsymbol{x}^\star) = \phi_0 + \sum_{j=1}^{M} \phi_j^\star,$$

where $\phi_0 = \mathrm{E}[g(\boldsymbol{x})]$ and $\phi_j^\star$ is the Shapley value for variable j for test observation $\boldsymbol{x}^\star$. That is, the Shapley values $\phi_1^\star, \ldots, \phi_M^\star$ explain the difference between the prediction $g(\boldsymbol{x}^\star)$ and the average prediction for the observations used to fit the model.

To use (1), [23] defines the contribution function $v(\mathcal{S})$ as the following expected prediction

$$v(\mathcal{S}) = \mathbb{E}[g(\boldsymbol{x})|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star]. \tag{2}$$

Here, $\boldsymbol{x}_\mathcal{S}$ denotes the features in subset $\mathcal{S}$ and $\boldsymbol{x}_\mathcal{S}^\star$ is the subset $\mathcal{S}$ of the feature vector $\boldsymbol{x}^\star$ that we want to explain. Thus, $v(\mathcal{S})$ denotes the expected prediction given that the features in subset $\mathcal{S}$ take the value $\boldsymbol{x}_\mathcal{S}^\star$.

If the features are continuous, we can write the conditional expectation in (2) as

$$\mathbb{E}[g(\boldsymbol{x})|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star] = \mathbb{E}[g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_\mathcal{S})|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star] = \int g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_\mathcal{S}^\star) f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star)\, d\boldsymbol{x}_{\bar{\mathcal{S}}}, \tag{3}$$

where $\boldsymbol{x}_{\bar{\mathcal{S}}}$ is the vector of features not in $\mathcal{S}$ and $f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star)$ is the conditional density of $\boldsymbol{x}_{\bar{\mathcal{S}}}$ given $\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star$. To compute Shapley values in practice, the conditional expectation in (3) needs to be approximated empirically. Note that in the rest of the paper we use lower case x-s for both random variables and realizations to keep the notation simple.

3 Estimating Shapley values

3.1 The independence approach

Since the conditional probability density is rarely known and difficult to estimate, [23] replaces it with the simple (unconditional) probability density

$$f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star) = f(\boldsymbol{x}_{\bar{\mathcal{S}}}). \tag{4}$$

The integral is thus approximated by

$$\mathbb{E}[g(\boldsymbol{x})|\boldsymbol{x}_\mathcal{S} = \boldsymbol{x}_\mathcal{S}^\star] \approx \int g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_\mathcal{S}^\star) f(\boldsymbol{x}_{\bar{\mathcal{S}}})\, d\boldsymbol{x}_{\bar{\mathcal{S}}}, \tag{5}$$

which is estimated by randomly drawing K times from the full training data set and calculating

$$v_{\mathrm{KerSHAP}}(\mathcal{S}) = \frac{1}{K} \sum_{k=1}^{K} g(\boldsymbol{x}_{\bar{\mathcal{S}}}^k, \boldsymbol{x}_\mathcal{S}^\star). \tag{6}$$

Here, $\boldsymbol{x}_{\bar{S}}^k$, $k = 1, \ldots, K$ are the samples from the training set and $g(\cdot)$ is the estimated prediction model. In the Shapley literature, the approximation (5) is sometimes termed the interventional conditional expectation, while (3) is denoted the observational conditional expectation. See e.g. [7] for more details.

Unfortunately, when the features are not independent, [2] demonstrates that naively replacing the conditional probability function with the unconditional one leads to very inaccurate Shapley values.

3.2 The Gaussian copula method

In [2], one of the proposed methods for estimating $f(\boldsymbol{x}_{\bar{S}}|\boldsymbol{x}_S = \boldsymbol{x}_S^\star)$ without relying on the naive assumption of independence is based on the *Gaussian copula*. A copula is a function that characterizes the dependence in a random vector. By Sklar's theorem, any joint distribution function F with marginal cdf's $F_1, \ldots, F_M$ can be written as

$$F(\boldsymbol{x}) = C(F_1(x_1), \ldots, F_M(x_M)),$$

where C is the copula function. Copulas are distribution functions with uniform margins. The corresponding density is denoted by c.

There are several parametric families for the copula function. The Gaussian copula is a special case. It is derived by inverting the above display i.e.,

$$C(\boldsymbol{u}) = F(F_1^{-1}(u_1), \ldots, F_M^{-1}(u_M)),$$

and taking F as a multivariate Gaussian distribution. This gives rise to a parametric model (parametrized by a correlation matrix) that reflects Gaussian dependence, but can be combined with arbitrary marginal distributions.

To compute Shapley values, we first need an estimate of the marginal distributions and copula parameters. [2] proposed to approximate the marginals $F_1, \ldots, F_M$ by the corresponding empirical *cdf*s and parametrize the copula by the empirical correlation matrix of corresponding normal scores. Together this gives us an estimated model for the joint distribution

$$\hat{F}(\boldsymbol{x}) = \hat{C}(\hat{F}_1(x_1), \ldots, \hat{F}_M(x_M)).$$

The conditional expectation in (3) can now be approximated by simulating conditionally from the estimated model. More precisely, let $\boldsymbol{x}_{\bar{S}}^k$, $k = 1, \ldots, K$ be simulated values of $\boldsymbol{x}_{\bar{S}}$ given $\boldsymbol{x}_S = \boldsymbol{x}_S^\star$ and compute

$$v_{\text{KerSHAP}}(S) = \frac{1}{K}\sum_{k=1}^{K} g(\boldsymbol{x}_{\bar{S}}^k, \boldsymbol{x}_S^\star). \tag{7}$$

Conditional simulation from the Gaussian copula can be achieved in essentially the same way as for the multivariate Gaussian distribution, see [2] for more details.

The Gaussian copula model is very flexible with regard to the marginal distributions, but quite restrictive in the dependence structures it can capture. It can only represent radially symmetric dependence relationships and does not allow for tail dependence (i.e., joint occurrence of extreme events has small probability). We therefore wish to use more flexible copula models and we shall focus on vine copula models specifically in what follows.

4 Extending the Shapley framework with vine copulas

A vine copula is a multivariate copula that is constructed from a set of bivariate ones, so-called *pair-copulas*. All of these bivariate copulas may be selected completely freely as the resulting structure is guaranteed to be

a valid copula. Hence, vine copulas are highly flexible, being able to characterise a wide range of complex dependencies.

Vine copulas have become very popular over the last decade. The main idea was originally proposed by [18] and further explored and discussed by [3, 4] and [21]. However, it was the paper [1], putting them in an inferential context, that really spurred a surge in empirical applications of these constructions. In this paper we use vine copulas to model the multivariate distributions involved in the Shapley framework. After a brief introduction to vine copulas in Section 4.1, we introduce two new methods for approximating the Shapely value contributions based on these structures in Sections 4.2 and 4.3. Finally, computationally efficient selection of the D-vine order is discussed in Section 4.4.

4.1 Background on vine copulas

In a vine copula the multivariate copula density is decomposed into a product of pair-copula densities. This decomposition is not unique. To organize all possible decompositions, the notion of *regular vines* (R-vines) was introduced by [4], and described in more detail in e.g. [9] and [21]. It involves the specification of a sequence of trees, each edge of which corresponds to a pair-copula. These pair-copulas constitute the building blocks of the joint R-vine distribution.

In this paper we use a special case of R-vines called D-vines [20] where each tree is a path. The density $f(x_1, \dots, x_M)$ corresponding to a D-vine may be written as

$$f(x_1, \dots, x_M) = \prod_{j=1}^{M} f_j(x_j) \times \tag{8}$$

$$\prod_{i=1}^{M-1} \prod_{j=1}^{M-i} c_{j,j+i|j+1,\dots,j+i-1} \left(F(x_j|x_{j+1}, \dots, x_{j+i-1}), F(x_{j+i}|x_{j+1}, \dots, x_{j+i-1}) \right),$$

where index i identifies the trees, and j runs over the edges in each tree. The inner product in the second line of (8) is a product of $M(M-1)/2$ bivariate copula densities, and is called a *D-vine copula* density. Note that the arguments of the pair-copulas are conditional distributions in all trees except the first, where they are the univariate margins. Figure 1 shows a 5-dimensional D-vine with 4 trees and 10 edges.

The density in (8) implies a specific order of conditioning. This order can be changed by a simple relabelling of the variables. For example, we can switch the roles of variables x_1 and x_M. Instead of pair-copulas $c_{1,2}$ and $c_{M-1,M}$ we will then get pair-copulas $c_{M,2}$ and $c_{M-1,1}$ in the first tree. Each permutation of $(1, 2, \dots, M)$ therefore gives rise to a different model. These permutations are called *orders* of the D-vine and will play an important role later on.

The key to the construction in (8) is that all copulas involved in the decomposition are bivariate and can belong to different families. There are no restrictions regarding the copula types that can be combined; the resulting structure is guaranteed to be valid. A further advantage with R-vine copulas is that the conditional distributions $F(x|\boldsymbol{v})$ constituting the pair-copula arguments can be evaluated using a recursive formula derived in [18]:

$$F(x|\boldsymbol{v}) = \frac{\partial C_{xv_j|\boldsymbol{v}_{-j}}(F(x|\boldsymbol{v}_{-j}), F(v_j|\boldsymbol{v}_{-j}))}{\partial F(v_j|\boldsymbol{v}_{-j})}. \tag{9}$$

Here $C_{xv_j|\boldsymbol{v}_{-j}}$ is a bivariate copula, v_j is an arbitrary component of $\boldsymbol{v}$ and $\boldsymbol{v}_{-j}$ denotes the vector $\boldsymbol{v}$ excluding v_j. By construction, R-vines have the important characteristic that the copulas in question are always present in the preceding trees of the structure, so that they are available without extra computations.

In their general form, vine copulas can represent all continuous multivariate distributions. However, to keep them tractable for inference, it is usually assumed that the pair copulas

$$c_{j,j+i|j+1,\dots,j+i-1} \left(F(x_j|x_{j+1}, \dots, x_{j+i-1}), F(x_{j+i}|x_{j+1}, \dots, x_{j+i-1}) \right)$$

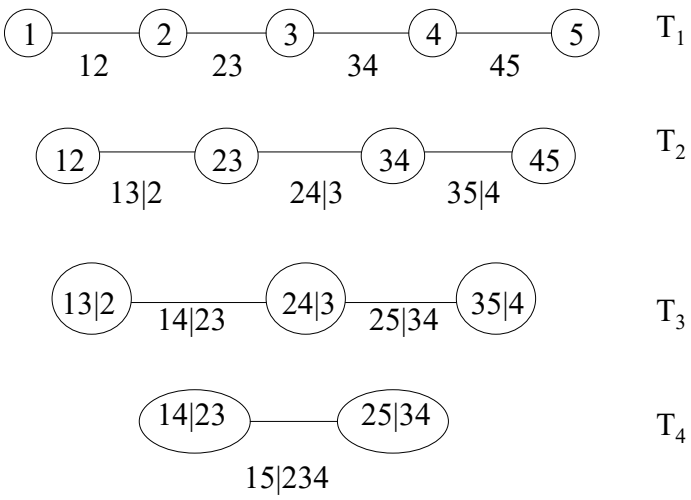

Figure 1: A D-vine with 5 variables, 4 trees and 10 edges. Each edge may be may be associated with a pair-copula.

are independent of the conditioning variables $x_{j+1}, \ldots, x_{j+i-1}$ except through the conditional marginal distributions. This leads to the so-called *simplified* vine copulas. We will consider both parametric and non-parametric models for the pair-copulas. More details about parametric and non-parametric estimation can be found in [1] and [24], respectively.

4.2 Shapley contributions: The conditional simulation method

Having determined the multivariate distribution of the explanatory variables, the next step is to compute the contribution function $v(\mathcal{S})$. We propose two different methods for estimating $v(\mathcal{S})$. In the first, to be described in this section, we generate samples from an estimate of the conditional distribution $f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}})$ and use these samples to estimate $v(\mathcal{S})$. In the second, which is treated in Section 4.3, $v(\mathcal{S})$ is estimated using ratios of copula densities.

To generate the samples from conditional distributions, we can use the *Rosenblatt transform* [31] and its inverse. The Rosenblatt transform $\boldsymbol{u} = T(\boldsymbol{v})$ of a random vector $\boldsymbol{v} = (v_1, \ldots, v_M) \sim F$ is defined as

$$u_1 = F(v_1), \quad v_2 = F(v_2|v_1), \quad \ldots, \quad u_M = F(v_M|v_1, \ldots, v_{M-1}),$$

where $F(v_m|v_1, \ldots, v_{m-1})$ is the conditional distribution of v_m given $v_1 \ldots, v_{m-1}$, $m = 2, \ldots, M$. The variables $u_1, \ldots u_M$ are then independent standard uniform variables. The inverse operation

$$v_1 = F^{-1}(u_1); \quad v_2 = F^{-1}(u_2|u_1); \quad \ldots; \quad v_M = F^{-1}(u_M|u_1, \ldots, u_{M-1}),$$

can be used to simulate from a distribution. For any joint distribution F, if $\boldsymbol{u}$ is a vector of independent random variables, $\boldsymbol{v} = T^{-1}(\boldsymbol{u})$ has distribution F.

In what follows we outline the procedure for generating the kth sample from the conditional distribution $F(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^*_{\mathcal{S}})$:

1. For each $j \in \mathcal{S}$, let $u^*_j = \hat{F}_j(x^*_j)$, where $\hat{F}_j$ is the empirical distribution function of x_j.
2. Let $\boldsymbol{w}_{\bar{\mathcal{S}}}$ be a vector with $|\bar{\mathcal{S}}|$ elements with arbitrary values between 0 and 1. Set $\boldsymbol{u} = (\boldsymbol{w}_{\bar{\mathcal{S}}}, \boldsymbol{u}^*_{\mathcal{S}})$ and let $\boldsymbol{v} = T(\boldsymbol{u})$, where $T(\cdot)$ is the Roseblatt transform.
3. Generate the vector $\boldsymbol{z}_{\bar{\mathcal{S}}}$ by sampling $|\bar{\mathcal{S}}|$ independent uniform U[0,1] distributed variates.

4. Replace the $|\bar{\mathcal{S}}|$ elements corresponding to the subset $\bar{\mathcal{S}}$ in $\boldsymbol{v}$ by $\boldsymbol{z}_{\bar{\mathcal{S}}}$.
5. Obtain $\boldsymbol{u} = T^{-1}(\boldsymbol{v})$ using the inverse Rosenblatt transform $T^{-1}(\cdot)$.
6. Finally, for each $j \in \bar{\mathcal{S}}$, let $x_j = \hat{F}_j^{-1}(u_j)$, where $\hat{F}_j^{-1}$ is the empirical quantile function of x_j.

Step 2 ensures that the values of the conditioning variables are the same in all samples. Having generated K samples $\boldsymbol{x}_{\bar{\mathcal{S}}}^1, \dots \boldsymbol{x}_{\bar{\mathcal{S}}}^K$ from the conditional distribution $f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^\star)$ we use (7) to compute $v_{\text{KerSHAP}}(\mathcal{S})$.

Whether or not we can compute $T^{-1}(\cdot)$ easily for a given D-vine depends on its *implied sampling orders* [10]. In particular, the conditioning variables have to appear either first or last in the D-vine structure. For example, a D-vine with order $1-2-3-4$ allows to easily simulate $\bar{\mathcal{S}} = \{3,4\}$ given $\mathcal{S} = \{1,2\}$ and $\bar{\mathcal{S}} = \{1,2\}$ given $\mathcal{S} = \{3,4\}$, but simulating $\bar{\mathcal{S}} = \{2,3\}$ given $\mathcal{S} = \{1,4\}$ is only possible through expensive multivariate numerical integration.

More formally, assume that we have a certain permutation $\boldsymbol{\pi} = (\pi_1, \dots \pi_M)$ of $(1, \dots, M)$. The corresponding D-vine may then be used to generate samples from conditional distributions $f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^\star)$ where $\mathcal{S}$ either is of the form $\mathcal{S} = \{\pi_1, \dots, \pi_k\}$ or $\mathcal{S} = \{\pi_M, \dots, \pi_{M-k+1}\}$ for $k = 1, \dots, M$. In Section 4.4, we use this fact to search for a small set of models that allows for simulation conditionally on any viable coalition $\mathcal{S}$.

4.3 Shapley contributions: The ratio method

For vine copula models, conditional simulation often involves numerical integration or inversion, which significantly slows down the algorithms. [25] proposed an alternative way to approximate conditional expectations based on copulas. The idea is to weight every sample in (7) in a way that accounts for the dependence.

It turns out that the appropriate weights are given by a ratio of copula densities. For simplicity denote $u_j = F(x_j)$, $j = 1, \dots, M$. If we have continuous variables, we can compute

$$
\begin{aligned}
v(\mathcal{S}) = \mathbb{E}[g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}})|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^\star] &= \int g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) f(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^\star)\, d\boldsymbol{x}_{\bar{\mathcal{S}}} \\
&= \int g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) f(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star)/f(\boldsymbol{x}_{\mathcal{S}}^\star)\, d\boldsymbol{x}_{\bar{\mathcal{S}}} \\
&= \int g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) \frac{c(\boldsymbol{u}_{\bar{\mathcal{S}}}, \boldsymbol{u}_{\mathcal{S}}^\star) \prod_{j=1}^M f(x_j)}{c(\boldsymbol{u}_{\mathcal{S}}^\star) \prod_{j\in\mathcal{S}} f(x_j)}\, d\boldsymbol{x}_{\bar{\mathcal{S}}} \\
&= \int g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) \frac{c(\boldsymbol{u}_{\bar{\mathcal{S}}}, \boldsymbol{u}_{\mathcal{S}}^\star)}{c(\boldsymbol{u}_{\mathcal{S}}^\star)} \frac{f(\boldsymbol{x}_{\bar{\mathcal{S}}})}{c(\boldsymbol{u}_{\bar{\mathcal{S}}})}\, d\boldsymbol{x}_{\bar{\mathcal{S}}} \\
&= E_{\boldsymbol{x}_{\bar{\mathcal{S}}}} \left[g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) \frac{c(\boldsymbol{u}_{\bar{\mathcal{S}}}, \boldsymbol{u}_{\mathcal{S}}^\star)}{c(\boldsymbol{u}_{\bar{\mathcal{S}}})} \right] / c(\boldsymbol{u}_{\mathcal{S}}^\star) \\
&= E_{\boldsymbol{x}_{\bar{\mathcal{S}}}} \left[g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) \frac{c(\boldsymbol{u}_{\bar{\mathcal{S}}}, \boldsymbol{u}_{\mathcal{S}}^\star)}{c(\boldsymbol{u}_{\bar{\mathcal{S}}})} \right] / E_{\boldsymbol{u}_{\bar{\mathcal{S}}}} \left[\frac{c(\boldsymbol{u}_{\bar{\mathcal{S}}}, \boldsymbol{u}_{\mathcal{S}}^\star)}{c(\boldsymbol{u}_{\bar{\mathcal{S}}})} \right].
\end{aligned}
$$

The expression in the third line follows from the definition of a copula given in Section 3.2, while the one in the fourth line is obtained using

$$
f(\boldsymbol{x}_{\bar{\mathcal{S}}}) = c(\boldsymbol{u}_{\bar{\mathcal{S}}}) \prod_{j\in\bar{\mathcal{S}}} f(x_j).
$$

To approximate the last line, we can estimate a vine copula model $\hat{c}$ and replace the expectations by a sample average over a (possibly random) subset of the training data:

$$
v_{\text{KerSHAP}}(\mathcal{S}) = \frac{\sum_{k=1}^K g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}_{\mathcal{S}}^\star) \hat{c}(\boldsymbol{u}_{\bar{\mathcal{S}}}^k, \boldsymbol{u}_{\mathcal{S}}^\star)/\hat{c}(\boldsymbol{u}_{\bar{\mathcal{S}}}^k)}{\sum_{k=1}^K \hat{c}(\boldsymbol{u}_{\bar{\mathcal{S}}}^k, \boldsymbol{u}_{\mathcal{S}}^\star)/\hat{c}(\boldsymbol{u}_{\bar{\mathcal{S}}}^k)}. \tag{10}
$$

We use the denominator in (10) instead of $c(\boldsymbol{u}_{\mathcal{S}}^\star)$ directly, since using the latter, for all subsets $\mathcal{S}$, one needs the marginal for the subset $\mathcal{S}$ in addition to the marginal for subset $\bar{\mathcal{S}}$. As will be discussed below and in Section 4.4, not all marginals are available in closed form for a given D-vine. Hence, we need to fit more than one D-vine to be able to obtain all marginals in (10) in closed form. Using $c(\boldsymbol{u}_{\mathcal{S}}^\star)$ would mean that we had to use

even more different D-vines. Similar expressions for discrete or mixed data and theoretical guarantees for (10) can be found in [25]. Note that the formula in (10) is very similar to the one for the empirical method in [2]. However, while the weights in that paper were computed using a Gaussian kernel, they are here given as ratios of copula densities.

The joint vine copula density $\hat{c}(\boldsymbol{u}_{\mathbb{S}}, \boldsymbol{u}^{\star}_{\mathbb{S}})$ is easily computed from (8) irrespective of the D-vine order. However, only some of the marginals $\hat{c}(\boldsymbol{u}_{\bar{\mathbb{S}}})$ are available in closed form for a given D-vine. For example, a D-vine with order 1−2−3−4 allows to easily compute the marginals $c_{1,2}$, $c_{2,3}$, $c_{3,4}$, $c_{1,2,3}$ and $c_{2,3,4}$, but not any other marginals. To formalize this, we again identify the D-vine structure with a permutation $\boldsymbol{\pi} = (\pi_1, \dots \pi_M)$ of $(1, \dots, M)$. From this permutation we may easily compute all marginals $\hat{c}(\boldsymbol{u}_{\bar{\mathbb{S}}})$ where $\bar{\mathbb{S}} = \{\pi_k, \pi_{k+1}, \dots, \pi_\ell\}$ for $1 \leq k \leq \ell \leq M$.

4.4 Choice of D-vine structures

We can use D-vine copula models in both the conditional simulation method and the ratio method. Depending on our choice of method, we need to either simulate conditionally from an estimated model or compute a ratio of copula densities. How efficiently we can do this numerically depends on the interplay of the coalition $\mathbb{S}$ and the order of variables in the D-vine. Generally, there are $M!/2$ distinct D-vines when we have M variables. Usually, when using vines, one looks for the D-vine maximising dependence in the first trees. The nature of the problem treated in this paper is a bit different from the ones previously discussed in the literature.

Let $\mathbb{Z}$ be the set of all conditional distributions $f(\boldsymbol{x}_{\bar{\mathbb{S}}}|\boldsymbol{x}_{\mathbb{S}} = \boldsymbol{x}^{\star}_{\mathbb{S}})$ to be used in the conditional simulation method, or all copula marginals $\hat{c}(\boldsymbol{u}_{\bar{\mathbb{S}}})$ to be computed in the ratio method. In the previous two sections, we identified the conditional distributions or copula marginals that may be easily obtained for a given D-vine. In this section we propose a randomized search method that minimizes computational complexity by finding a small set of D-vine models that covers $\mathbb{Z}$. The procedure is as follows:

1. Generate B random permutations of $(1, \dots, M)$.
2. For each permutation, find the number of conditional distributions or copula marginals that may be easily obtained (see Sections 4.2 and 4.3).
3. Pick the permutation that covers most of the remaining sets in $\mathbb{Z}$. Remove the covered sets from $\mathbb{Z}$.
4. Go back to step 1 until no subsets are remaining.

The result is a collection D-vine structures based on which all conditional distributions/copula marginals may be easily computed. We have used $B = 100$ permutations, which gave fairly stable results in our experiments with 10 features. Note that the permutation the algorithm picks for any given margin is somewhat arbitrary. Since each coalition only contributes a small part to the final Shapley value, an additional step to find the optimal permutation is unlikely to be worth the effort. Empirically, this approach reduces the number of D-vine models to estimate from 2^M to around 2^{M-2} for conditional simulation and to 2^{M-3} for the ratio method. That is, the computational time is reduced by 75% − 87.5%.

5 Simulation studies

In this section, we discuss a simulation study designed to compare different ways to estimate Shapley values. Specifically, we compare our suggested approaches with [23]'s independence estimation approach (below called *independence*) and [2]'s empirical, Gaussian and Gaussian copula estimation approaches. A short description of each approach is given in Table 1. For the approaches presented in this paper, we have fitted both a non-parametric and a parametric vine. The independence, empirical, Gaussian and Gaussian copula approaches are all implemented in the `R` package `shapr` [35], and the plan is to also include the approaches proposed in this paper. In the empirical method we used the default value 0.1 for kernel bandwidth.

The simulation model is detailed in Section 5.1 and the actual design of the experiments is given in Section 5.2. Further, Section 5.3 describes the evaluation measure used to quantify the accuracy of the different methods, and finally, Section 5.4 gives the results.

Table 1: A short description of the approaches used to estimate (2) in the simulation studies.

Method	Citation	Description
Independence	[23]	Assume the features are independent. Estimate (2) by (6) where $x_{\bar{S}}^k$ are sub-samples from the training data set.
Empirical	[2]	Calculate the Mahalanobis distance between the observation being explained and every training instance. Use this distance to calculate a weight for each training instance. Approximate (2) using a function of these weights.
Gaussian	[2]	Assume the features are jointly Gaussian. Sample N times from the corresponding conditional distribution. Estimate (2) with (7) using this sample.
Gaussian copula	[2]	Assume the dependence structure of the features can be approximated by a Gaussian copula. Sample N times from the corresponding conditional distribution. Estimate (2) with (7) using this sample.
Parametric cond. sim.		Assume the features are from a vine with all pair-copulas chosen as Clayton Survival copulas. Sample N times from the corresponding conditional distribution. Estimate (2) with (7) using this sample.
Non-parametric cond. sim.		Assume the features are from a non-parametric vine. Sample N times from the corresponding conditional distribution. Estimate (2) with (7) using this sample.
Parametric ratio		Assume the features are from a vine with all pair-copulas chosen as Clayton Survival copulas. Estimate (2) with (10).
Non-parametric ratio		Assume the features are from a non-parametric vine. Estimate (2) with (10).

5.1 Simulation model

To evaluate the different approaches, we need cases for which we know the true feature distribution. Moreover, we have to use multivariate distributions that have known conditional distributions. There are not many such distributions, but one example, which allows for heavy-tailed and skewed marginals and non-linear dependence, is the multivariate Burr distribution.

The M-dimensional Burr distribution has the density [40]

$$f_M(\boldsymbol{x}) = \frac{\Gamma(p+M)}{\Gamma(p)} \left(\prod_{m=1}^{M} b_m r_m \right) \frac{\prod_{m=1}^{M} x_m^{b_m - 1}}{\left(1 + \sum_{m=1}^{M} r_m x_m^{b_m} \right)^{p+M}},$$

for $x_m > 0$. Here, p, $b_1, \ldots, b_M$ and $r_1, \ldots r_M$ are the parameters of the distribution. The Burr distribution is a compound Weibull distribution with the gamma distribution as compounder [40]. It can be regarded as a special case of the Pareto IV distribution [44].

Any conditional distribution of the multivariate Burr distribution is also a multivariate Burr distribution [40]. The conditional density $f(x_1, \ldots, x_S | x_{S+1} = x_{S+1}^\star, \ldots, x_M = x_M^\star)$ is an S-dimensional Burr density with

parameters $\tilde{p}, \tilde{b}_1, \ldots, \tilde{b}_S, \tilde{r}_1, \ldots, \tilde{r}_S$, where $\tilde{p} = p + M - S$ and for all $j = 1, \ldots, S$,

$$\tilde{b}_j = b_j, \qquad \tilde{r}_j = \frac{r_j}{1 + \sum_{m=S+1}^{M} r_m \left(x_m^\star\right)^{b_m}}.$$

According to [8], the copula corresponding to the multivariate Burr distribution is a Clayton survival copula. Thus, the multivariate Burr distribution may be represented by a vine copula where the pair-copulas are bivariate Clayton survival copulas. For this reason, we have fitted both a parametric and a non-parametric vine for the two approaches presented in this paper. The parametric vine is a simplified vine with bivariate Clayton survival copulas, while the non-parametric vine is fitted using the methodology in [24]. We use the `R` package `rvinecopulib` [26] for parameter estimation. Note that the parametric vines are correctly specified in this simulation example, meaning that if the non-parametric vines have similar performance to the corresponding parametric vines, it indicates that the non-parametric vines provide a satisfactory fit to the multivariate Burr distribution.

In our experiments, we simulate data from 3 different 10-dimensional Burr distributions. All three distributions have

$$\boldsymbol{b} = (2, 4, 6, 2, 4, 6, 2, 4, 6, 6)$$
$$\boldsymbol{r} = (1, 3, 5, 1, 3, 5, 1, 3, 5, 5),$$

while they have p equal to 0.5, 1, and 1.5, respectively. The three values of p correspond to pairwise Kendall's τ values of 0.5, 0.33, and 0.25, respectively.

In addition to the feature distribution, we need to specify the sampling model for the response y and the machine learning approach used to fit the predictive model $g(\boldsymbol{x})$. Inspired by [6] we chose the following non-linear and heteroscedastic function for y:

$$y = u_1 u_2 \exp(1.8\, u_3 u_4) + u_5 u_6 \exp(1.8\, u_7 u_8) + u_9 \exp(1.8\, u_{10}) + 0.5(u_1 + u_5 + u_9)\epsilon, \tag{11}$$

where $u_m = F_m(x_m)$. Further we assume that $\boldsymbol{x}$ is multivariate Burr distributed and that $F_m(\cdot)$ is the true parametric distribution function. Finally, ϵ is standard normal distributed and independent of all the x_ms.

5.2 Experimental design

We perform 3 different experiments with training sample sizes N_{train} equal to 100, 1000 and 10000, respectively. In each experiment we repeat the following steps 50 times for each of the 3 different Burr distributions described in Section 5.1:

1. Generate simulated training data by
   - Sampling N_{train} training observations from the chosen Burr distribution
   - Computing the corresponding y values using (11).
2. Select the predictive model $g(\boldsymbol{x})$ as a Random forest with 500 trees, and fit this model using the R package `ranger` [43] (with default parameter settings) to the training data.
3. Sample $N_{\text{test}} = 100$ test observations from the chosen Burr distribution.
4. For all methods, possible subsets $\mathcal{S}$, and test observations $\boldsymbol{x}^\star$:
   - If one of the ratio methods: Compute $v_{\text{KerSHAP}}(\mathcal{S})$ using (10) with $K = 1000$.
   - If the empirical method: Compute $v_{\text{KerSHAP}}(\mathcal{S})$ using the formula given in Section 3.3 in [2] with $\eta = 0.95$.
   - If one of the remaining methods in Table 1: Generate $K = 1000$ samples from the estimated conditional distribution $p(\boldsymbol{x}_{\bar{\mathcal{S}}} | \boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}_{\mathcal{S}}^\star)$ and compute $v_{\text{KerSHAP}}(\mathcal{S})$ using (7).
5. For all test observations $\boldsymbol{x}^\star$ and all methods, compute the Shapley value using (1) with the $v_{\text{KerSHAP}}(\mathcal{S})$ values for all subsets $\mathcal{S}$ for the current test observation.

For all approaches, the multivariate model for the features is fitted using the training data.

5.3 Evaluation method

We measure the performance of each method based on the mean absolute error (MAE), across both the features and the sample space. MAE is defined as

$$\text{MAE(method } q) = \frac{1}{T}\sum_{i=1}^{T}\frac{1}{M}\sum_{j=1}^{M}|\phi_{j,\text{true}}(\boldsymbol{x}_i) - \phi_{j,q}(\boldsymbol{x}_i)|, \tag{12}$$

where $\phi_{j,q}(\boldsymbol{x})$ and $\phi_{j,\text{true}}(\boldsymbol{x})$ denote, respectively, the Shapley value estimated with method q and the corresponding true Shapley value for the prediction $g(\boldsymbol{x})$. Further, M is the number of features and T is the number of test observations. The true Shapley values are computed using the algorithm in Section 5.2, except that in step 4, we generate $K = 1000$ samples from the true conditional Burr distribution $p(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^{\star}_{\mathcal{S}})$, and use Monto Carlo integration with 10000 samples to compute $v(\mathcal{S})$.

As stated in Section 5.2, for each feature distribution and choice of N_{train} we repeat the test procedure 50 times and report the average MAE over those 50 repetitions. Hence, the quality of the Shapley values is evaluated based on a total of $T = 5000$ test observations. Sampling new data for each batch reduces the influence of the exact shape of the fitted predictive model.

5.4 Results

The results of the simulation study are shown in Figure 2. The nine panels correspond to different combinations of sample size N_{train} (columns) and dependence parameter p (rows). Each bar represents the MAE achieved by a particular method, where smaller values indicate higher accuracy.

Analogous to [2], we clearly see that the independence method is not suitable for estimating Shapley values when covariates are dependent. The other methods have more similar performance, but with the vine-based methods favoured overall. The parametric vine methods perform slightly better than the non-parametric ones in all scenarios. This is to be expected, because, as previously stated, the true simulation model can be represented as a vine copula with survival Clayton pair-copulas. Hence, the parametric models are correctly specified. On real data sets, the parametric assumption will rarely hold and can be severely violated, however.

For very small sample sizes ($N_{\text{train}} = 100$), even the (correctly specified) parametric vine methods have only a small advantage and the non-parametric methods are outperformed by some of their competitors when $p = 1.5$ (weaker dependence). This is not surprising, since we are estimating very complex models — up to 90 parameters for parametric vines — from very limited information. For medium to large samples ($N_{\text{train}} \geq 1000$), the vine-based methods outperform their competitors by a decent margin. When the dependence is strong ($p = 0.5$), the MAE of the non-parametric ratio method is only approximately 20% of the MAE of the best of the previously proposed methods for $N_{\text{train}} = 10000$ and the corresponding ratio for $p = 1.5$ is 50%.

It also becomes apparent that the non-parametric ratio method performs slightly better than its Monte Carlo analogue. The likely reason is that conditional simulation involves many numerical approximations for integration and inversion that accumulate. The two parametric vine methods on the other hand have virtually the same accuracy.

We can conclude that our vine-based methods improve over previous methods for estimating Shapley values. However, that comes at a computational cost. Figure 3 shows the average computation time required to estimate Shapley values for 100 test observations (including fitting copula models). We can confirm that the vine-based methods are slower than its competitors. The ratio methods are around 10x slower than the Gaussian copula method, and the Monte Carlo methods even up to 100x. For practical purposes, the ratio method is therefore preferred. We also note that computation times are generally large. This is mainly due to the fact that we have to compute a large number of different Shapley contributions $v(\mathcal{S})$ ($2^{10} = 1,024$ for 10 covariates). This can be mitigated substantially by parallelizing computations and/or using the approximate weighted least squares method proposed by [23]. The latter approach, which is thoroughly described in Section 2.3.1 in [2] requires only a subset of Shapley contributions to be computed.

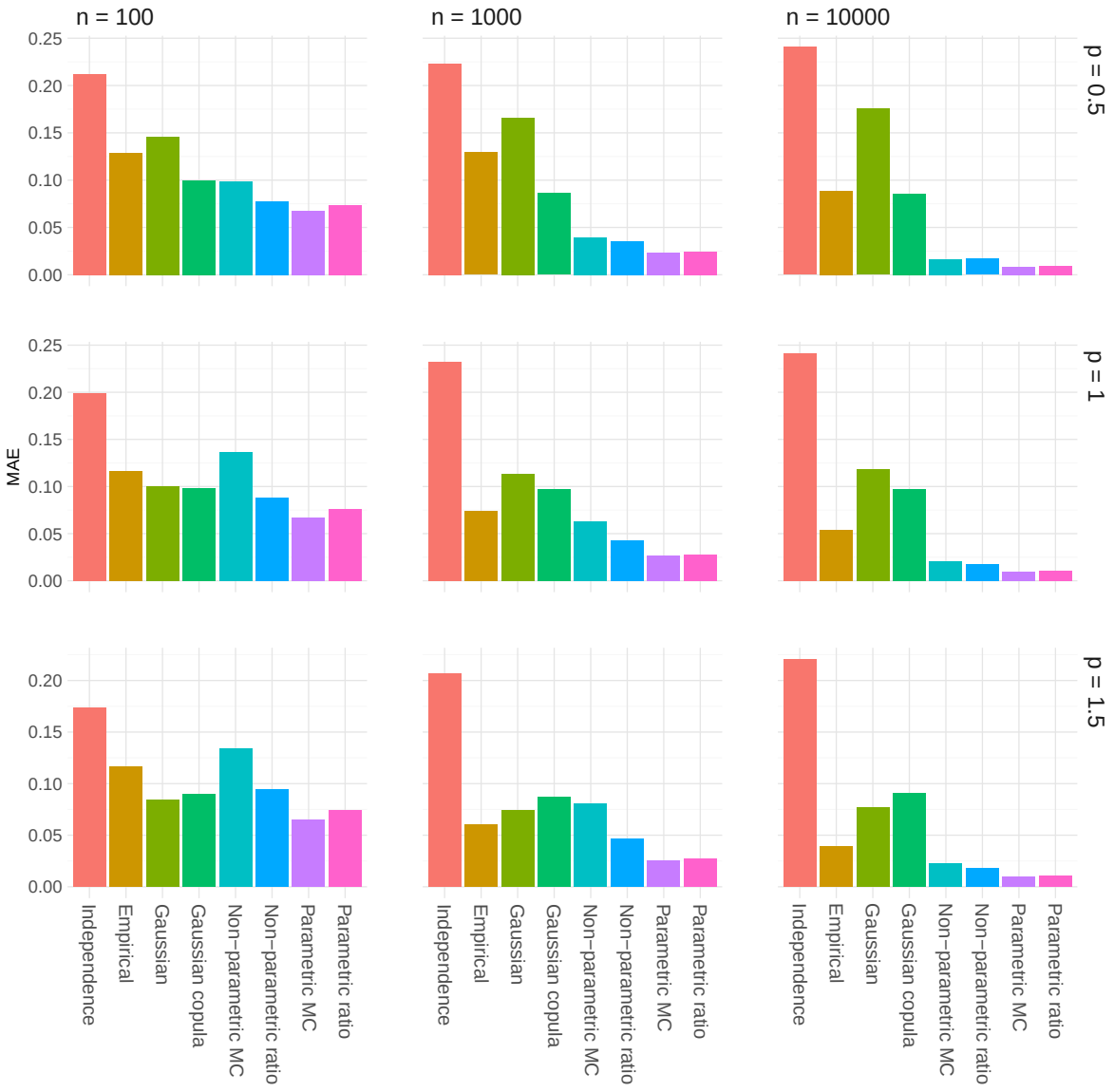

Figure 2: MAE for each combination of sample size, Burr distribution parameters and method. Each MAE-value is computed from 5000 test observations. The values 0.5, 1, and 1.5 of p correspond to pairwise Kendall's τs of 0.5, 0.33, and 0.25, respectively.

6 Real data example

In this section, we apply the methods discussed in this paper on the Abalone data set (available at http://archive.ics.uci.edu/ml/datasets/Abalone). It has previously been used in several machine learning studies, see e.g. [33, 37]. Moreover, it has been used in the related vine copula studies [6, 11, 14]. The data originate from a study by the Tasmanian Aquaculture and Fisheries Institute. An abalone is a kind of edible sea snail, the harvest of which is subject to quotas. These quotas are based partly on the age distribution of the abalones. To determine an abalone´s age, one cuts the shell through the cone, stains it, and counts the number of rings through a microscope. This is a highly time-consuming task. Hence, one would like to predict the age based on physical measurements that are easier to obtain. The Abalone data set was originally used

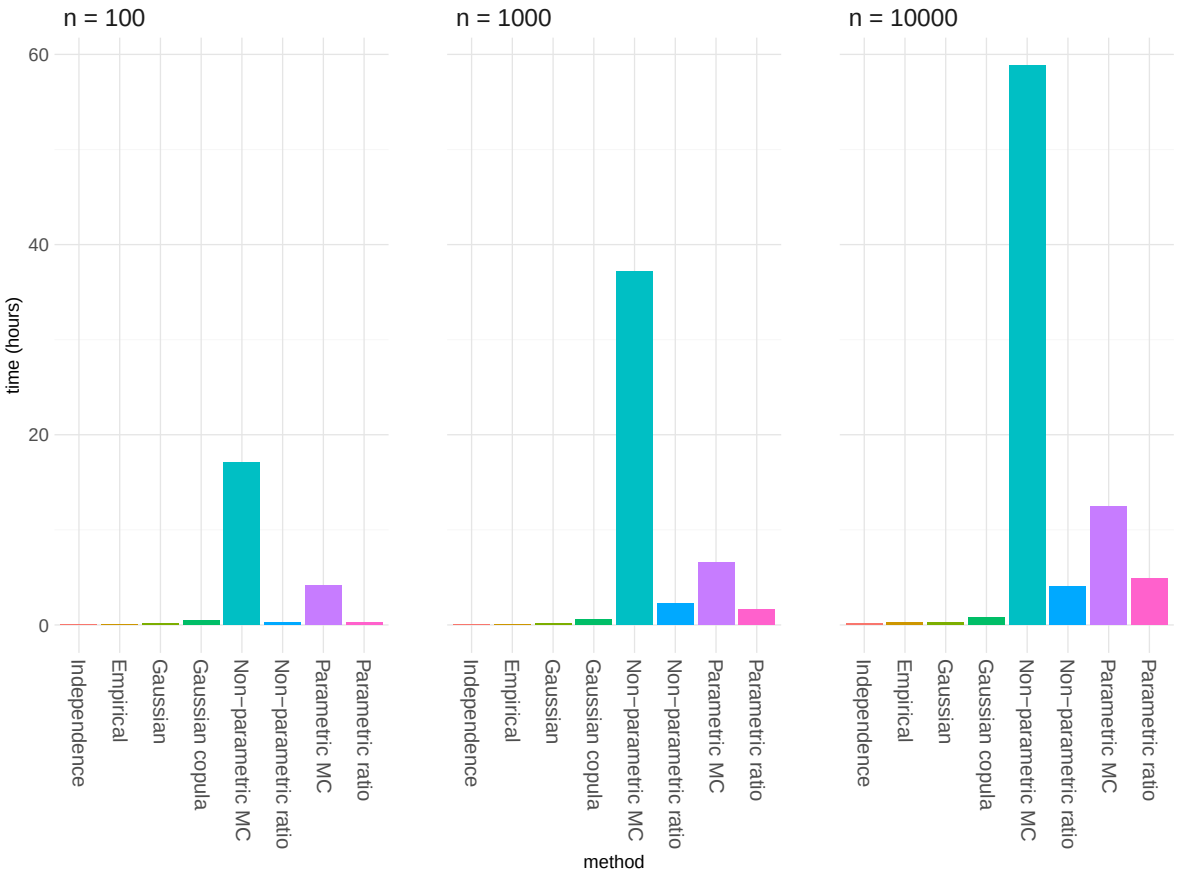

Figure 3: Average computation time (CPU hours) required for each sample size, method, on 100 test observations.

for this purpose. It consists of 4,177 samples on the following 9 variables: `Sex`, `Length`, `Diameter`, `Height`, `Whole weight`, `Shucked weight`, `Viscera weight`, `Shell weight` and `Age` measured by number of rings.

We do not include the variable `Sex` in our study since it is a discrete variable. Note that the use of regular vines does not exclude discrete data; examples of discrete and mixed discrete vines may be found for instance in [30] and [39]. However, many of the methods become more complicated when discrete data are involved.

Figure 4 shows the pairwise scatter plots, marginal density functions and pairwise Pearson correlation coefficients. There is clear non-linearity and heteroscedasticity among the pairs of variables. Moreover, it can be noted that all pairwise correlations between the explanatory variables are higher than 0.775.

We treat the age prediction as a regression problem. To be able to detect any potential non-linear relationships between the response and the explanatory variables, we use a Random forest model instead of linear regression. The Abalone data set was divided into a training set and a test set, containing 4,077 and 100 observations, respectively. The Random forest model was fitted to the training data, using the R package `ranger` with 500 trees and default parameter settings. Then, this model was used to predict the age (number of rings) for the observations in the test data set.

Since the non-parametric ratio method was the fastest, most stable, and best performing of the new vine copula methods in the simulation study, we compare the performance of this method with the independence, empirical, Gaussian and Gaussian copula approaches. Figure 5 shows the Shapley values for two of the test observations. As stated above, the Shapley values for a test observation explain the difference between the prediction for this test observation and the prediction from a model without any explanatory variables. For the Abalone data set the latter is 10 (number of rings), while the predicted values for test observations A and B are 11.9 and 6.2, respectively. This is in correspondence with most Shapley values being positive for test

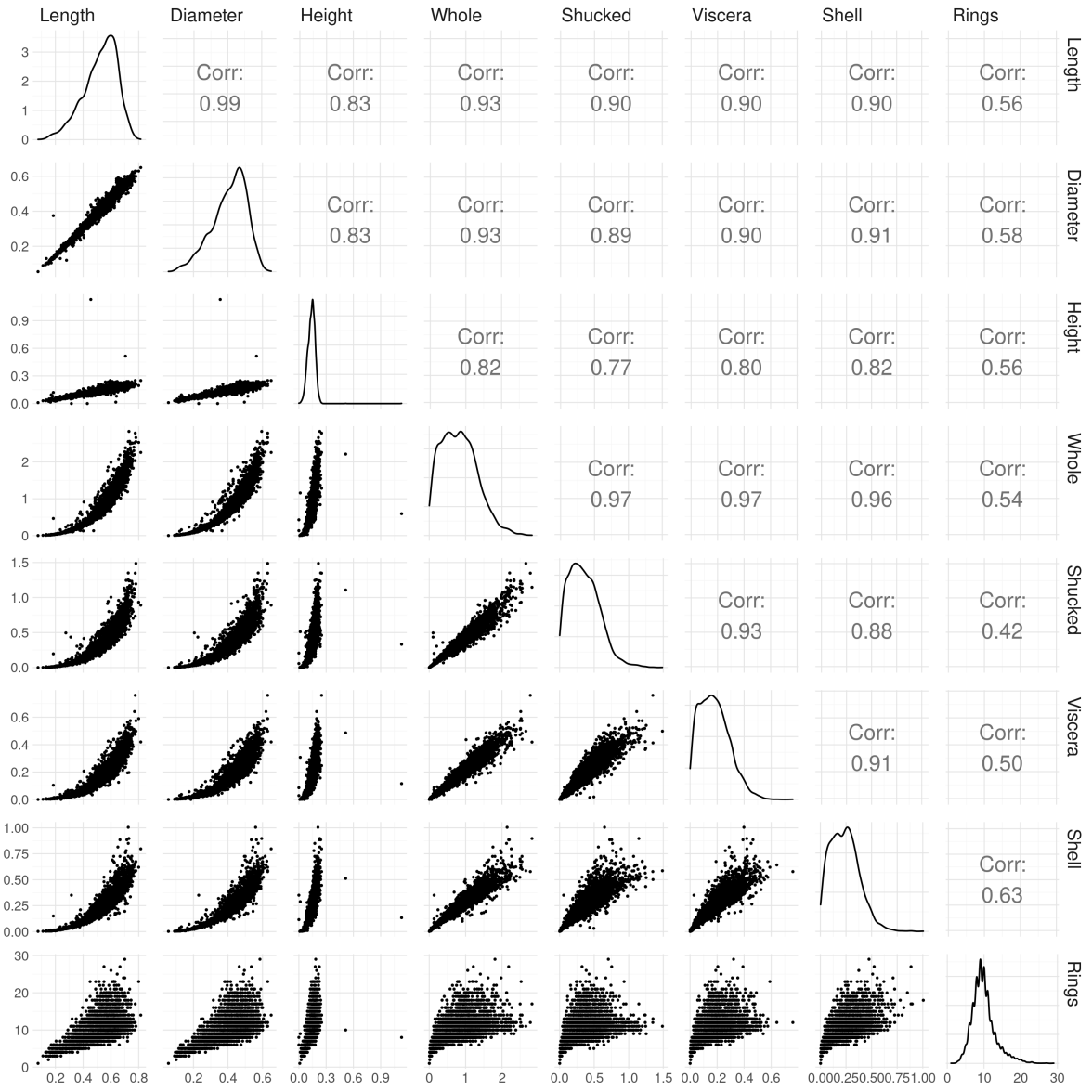

Figure 4: Pairwise scatter plots, marginal density functions and pairwise correlation coefficients for the explanatory variables and the response variable.

observation A and all (except for those for `Shucked weight` and `Length` using the independence method) being negative for test observation B.

Further, for observation A, all methods seem to agree with `Shell weight` being the most important variable, while `Shucked weight` is the second most important. For test observation B, the variables `Viscera weight`, `Height` and `Diameter` seem to be equally important according to the non-parametric ratio method. This is also the case for the other methods taking feature dependence into account. However, for this observation, the Shapley values obtained by the independence method are quite different from those obtained using the other methods. According to the independence method, the variables `Shell weight`, `Whole weight` and `Shucked weight` are the most important. Moreover, the Shapley values for `Shucked weight` and `Length` have even opposite signs from those obtained by the other approaches, showing that using this method, one may get misleading explanations.

Although we observe some differences, e.g. for the variables `Shucked weight` and `Shell Weight` for test observation A, the Shapley values produced by the Gaussian copula method and the non-parametric ratio method are quite similar. This is somewhat surprising, both in view of the results from the simulation study (see Figure 2) and the conditional distribution example to be discussed below. We have not been able to come up with a full explanation, but one theory is that the Gaussian copula method makes errors in opposite directions which cancel each other out in the Shapley formula.

A problem with evaluating Shapley values for real data is that there is no ground truth. Hence, we have to justify the results in other ways. In what follows, we use mainly the same framework as that proposed in [2]. For all approaches treated in this paper, the Shapley value is a weighted sum of differences $v(\mathcal{S} \cup \{j\}) - v(\mathcal{S})$ for several subsets $\mathcal{S}$. However, the approaches differ in how $v(\mathcal{S})$, or more specifically, the conditional distribution $p(\boldsymbol{x}_{\bar{\mathcal{S}}}|\boldsymbol{x}_{\mathcal{S}} = \boldsymbol{x}^\star_{\mathcal{S}})$, is estimated. Hence, if we are able to show that the samples from the conditional distributions generated using the non-parametric method are more representative than the samples generated using the previously proposed methods, it is likely that the Shapley values obtained using the non-parametric method are the most accurate.

Since there are many conditional distributions involved in the Shapley formula, we will not show all here. However, we have included some examples that illustrate that the non-parametric method gives more correct approximations to the true conditional distributions than the other approaches. First, Figure 6 shows plots of `Length` against `Shell weight` and `Viscera weight` against `Shell weight`. The grey dots are the training data. The blue dots are the samples from the conditional distribution of the variable at the x-axis given that `Shell weight` is equal to 0.1, generated using our method. The green and red dots are the corresponding samples generated using the Gaussian copula and independence approaches, respectively. It should be noted that the non-parametric ratio method does not involve any simulation. However, the method has an implicit statistical model that we can sample from for illustrative purposes. (10) can be seen as an expectation $E_P[g(\boldsymbol{x}_{\bar{\mathcal{S}}}, \boldsymbol{x}^\star_{\mathcal{S}})]$ with respect to a model P that assigns probability

$$\pi_\ell = \frac{\hat{c}(\boldsymbol{u}^\ell_{\bar{\mathcal{S}}}, \boldsymbol{u}^\star_{\mathcal{S}})/\hat{c}(\boldsymbol{u}^k_{\bar{\mathcal{S}}})}{\sum_{k=1}^K \hat{c}(\boldsymbol{u}^k_{\bar{\mathcal{S}}}, \boldsymbol{u}^\star_{\mathcal{S}})/\hat{c}(\boldsymbol{u}^k_{\bar{\mathcal{S}}})}$$

to the ℓth observation $\boldsymbol{x}^\ell$. Hence, we can simulate from this implicit model by drawing with replacement from the original observations $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^K$ using $(\pi_1, \ldots, \pi_K)$ as sampling probabilities. Both the Gaussian copula approach and the independence approach generate samples that are unrealistic, in the sense that they are far outside the range of what is observed in the training data. It might be confusing that this is the case for the Gaussian copula. This is due to the fact that we are sampling in the lower tail, where there is very strong tail dependence that the Gaussian copula is missing out on. It is well known that evaluation of predictive machine learning models far from the domain at which they have been trained, can lead to spurious predictions. Thus, it is important that the explanation methods are evaluating the predictive model at appropriate feature combinations. The samples generated by the ratio method are inside the range of what is observed in the training data.

In Figure 7 we study three different conditional distributions involved in the Shapley formula:

- The conditional distribution of `Shell weight` given all the other variables.
- The conditional distribution of `Length` and `Shucked weight` given `Viscera weight` and `Shell weight`.
- The conditional distribution of all variables except `Shucked weight` given `Shucked weight`.

For all the three distributions, we generate 1000 samples for three of the test observations using the non-parametric ratio, Gaussian copula and independence approaches. That is, we condition on four different sets of values. For each combination of test observation, conditional distribution and method, we compute the mean Mahalanobis distance between each sample and its ten nearest training samples, resulting in 1000 different mean distances. Each panel of Figure 7 shows the probability densities of such mean distances for a specific test observation and a specific conditional distribution (test observations A and B are the same as those in Figure 5). If the generated samples are realistic, we would expect the majority of the mean distances to be small.

For all conditional distributions and all test observations, the mode of the density corresponding to the independence approach is larger than those of the two other densities, indicating that the samples gener-

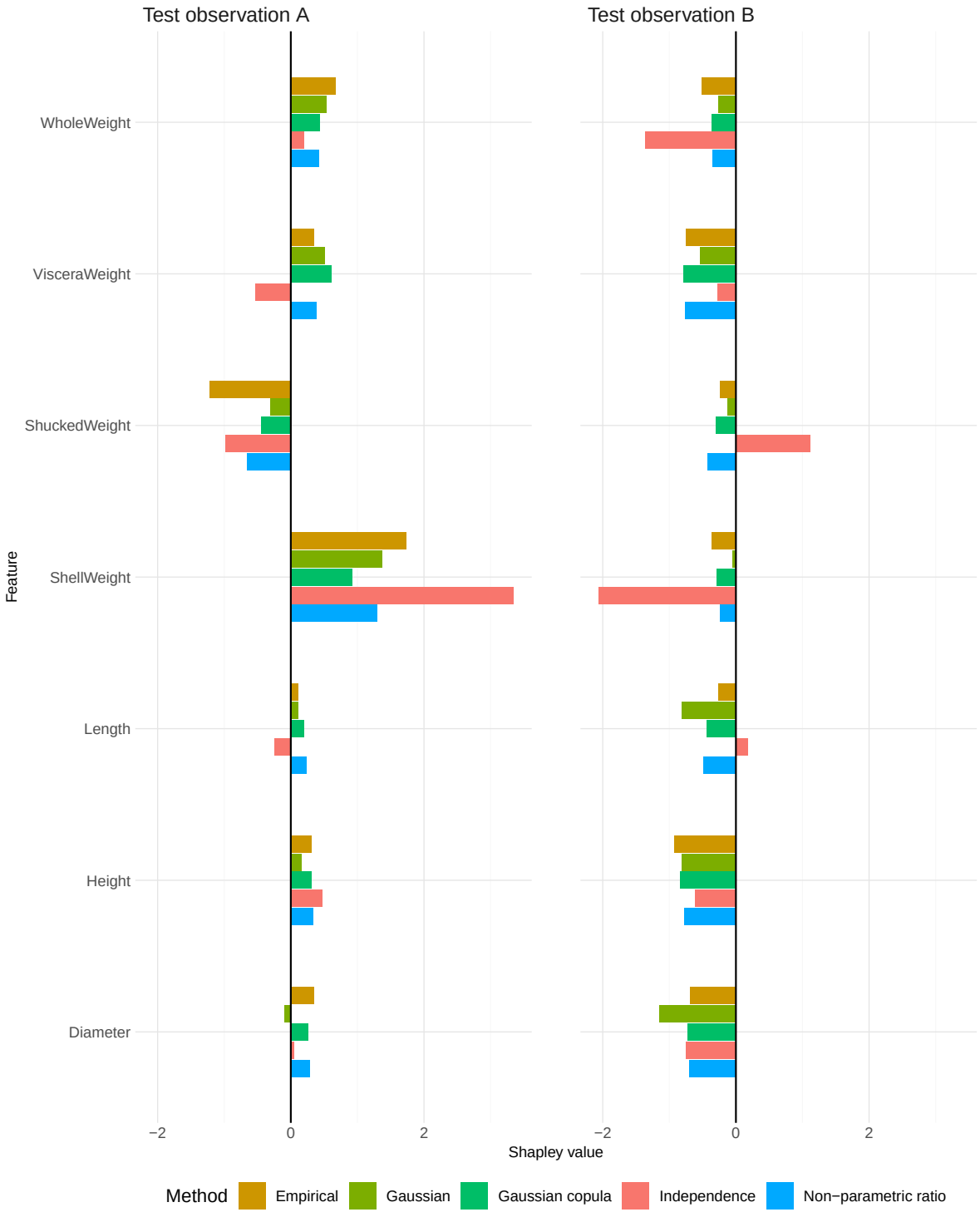

Figure 5: Shapley values for two of the test observations in the real data set computed using the different methods.

ated by the Gaussian copula and non-parametric ratio approaches are more realistic than those generated by the independence approach. Further, for the majority of the test observations/conditional distributions, the Mahalanobis distances corresponding to the non-parametric ratio approach are smaller than those corresponding to the independence and Gaussian copula approaches.

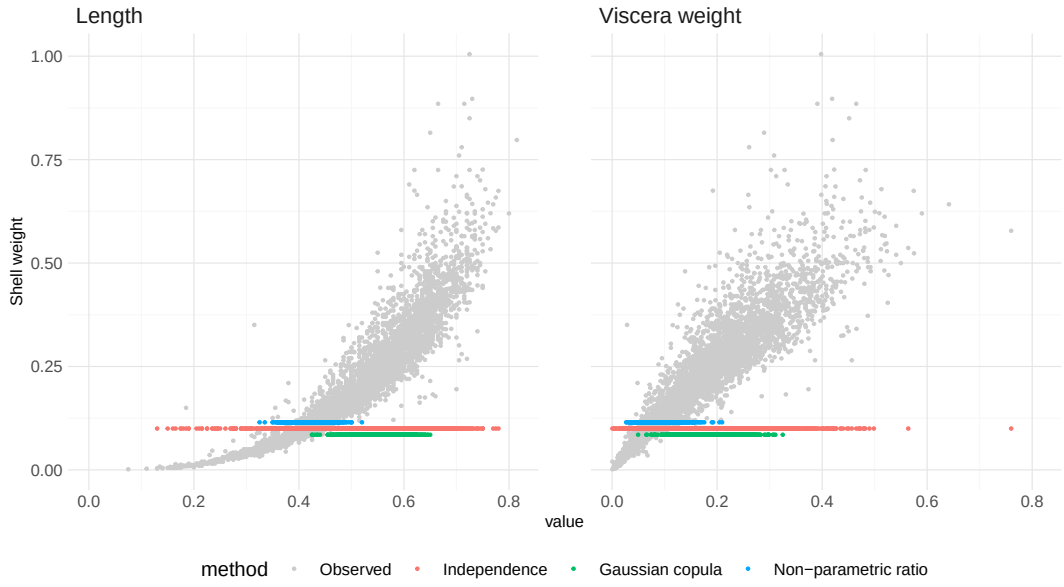

Figure 6: `Length` against `Shell weight` (left) and `Viscera weight` against `Shell weight` (right). The grey dots are the training data. The blue dots are the samples from the conditional distribution of the variable at the x-axis given that `Shell weight` is equal to 0.1 generated using the non-parametric ratio method, the green dots are the corresponding samples generated using the Gaussian copula method, and the red dots are the samples generated using the independence method. Note that the red and green dots have been slightly displaced vertically to improve visibility of the figure.

To summarize, we have illustrated that the Shapley values computed using the non-parametric ratio method and the previously proposed methods are different. We have tried to justify that this is because the non-parametric ratio method gives more correct approximations to the true conditional distributions for this data set.

7 Summary and discussion

Shapley values is a model-agnostic method for explaining individual predictions with a solid theoretical foundation. The original development of Shapley values for prediction explanation relied on the assumption that the features being described were independent. If the features in reality are dependent this may lead to incorrect explanations. Hence, there have recently been attempts of appropriately modelling/estimating the dependence between the features. Although the proposed methods clearly outperform the traditional approach assuming independence, they have their weaknesses. In this paper we have proposed two new approaches for modelling the dependence between the features. Both approaches are based on vine copulas, which are flexible tools for multivariate non-Gaussian distributions able to characterise a wide range of complex dependencies.

We have performed a comprehensive simulation study, showing that our approaches outperform the previously proposed methods. We have also applied the different approaches to a real data set, where the predictions to be explained were produced by a Random forest classifier designed to predict the age of an abalone (sea snail). In this case the true Shapley values are not known, but we provide results which indicate that the vine-based approaches provide more sensible approximations than the previously proposed methods.

The main part of the methodology proposed in this paper may be used for many other applications than computing Shapley values. The need for expressing statistical inference in terms of conditional quantities is ubiquitous in most natural and social sciences [28]. An obvious example is the estimation of the mean of some set of response variables conditioned on sets of explanatory variables taking specified values [6].

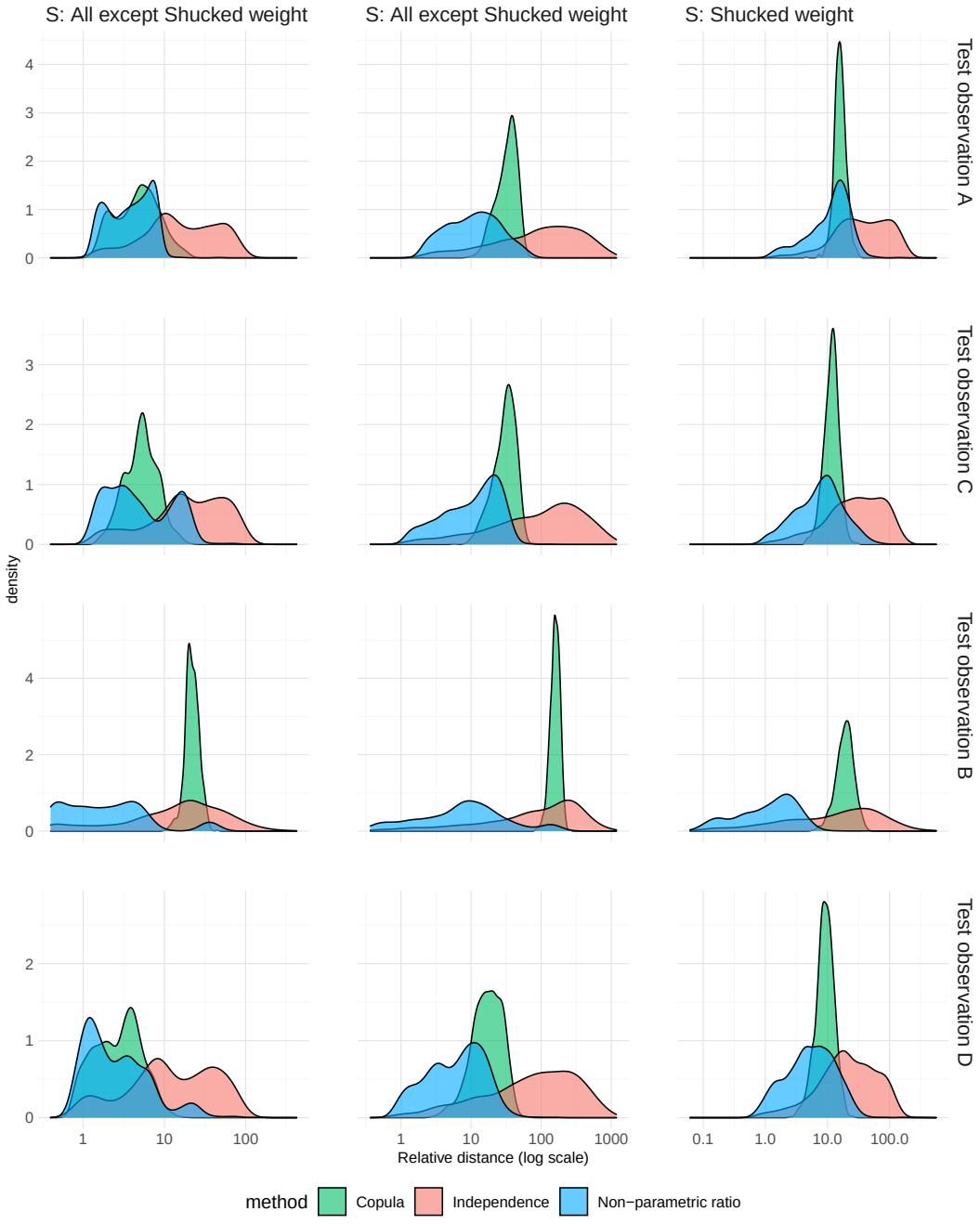

Figure 7: Probability densities of mean Mahalanobis distances for three different conditional distributions and three different individuals. See the text for a further description.

Other common tasks are the forecasting of volatilities or quantiles of financial time series conditioned on past history [34]. Problems of this kind often call for some sort of regression analysis, like the one presented in this paper.

The challenging issue in conditional density estimation is to circumvent the curse of dimensionality. Several methods have been proposed to estimate conditional densities; the classical kernel estimator [32], which has been refined and developed in many directions, see for example [5, 15, 17, 27]; local polynomial estimators [12, 16], and a local Gaussian correlation estimator [28]. However, most of these methods, if not all, are computationally intractable when either $\boldsymbol{x}_S$ or $\boldsymbol{x}_{\bar{S}}$ is not univariate, or both have dimension above

3-4. The suggested vine-based approaches work well when both $\boldsymbol{x}_S$ or $\boldsymbol{x}_{\bar{S}}$ are high dimensional. Hence, the methodology proposed in this paper may be regarded as a contribution to the field of non-parametric conditional density estimation.

Acknowledgements: This work is supported by the Norwegian Research Council grant 237718.

References

[1] Aas, K., C. Czado, A. Frigessi, and H. Bakken (2009). Pair-copula constructions of multiple dependence. *Insurance Math. Econom. 44*(2), 182–198.

[2] Aas, K., M. Jullum, and A. Løland (2021). Explaining individual predictions when features are dependent: More accurate approximations to Shapley values. *Artif. Intell. 298*, Article ID 103502, 24 pages.

[3] Bedford, T. and R. M. Cooke (2001). Probability density decomposition for conditionally dependent random variables modeled by vines. *Ann. Math. Artif. Intell. 32*, 245–268.

[4] Bedford, T. and R. M. Cooke (2002). Vines - a new graphical model for dependent random variables. *Ann. Statist. 30*(4), 1031–1068.

[5] Bertin, K., C. Lacour, and V. Rivoirard (2016). Adaptive pointwise estimation of conditional density function. *Ann. Inst. Henri Poincaré Probab. Stat. 52*(2), 939–980.

[6] Chang, B. and H. Joe (2019). Prediction based on conditional distributions of vine copulas. *Comput. Statist. Data Anal. 139*, 45–63.

[7] Chen, H., J. D. Janizek, S. Lundberg, and S.-I. Lee (2020). True to the model or true to the data? *Proceedings of the 2020 ICML Workshop on Human Interpretability in Machine Learning*, pp. 123–129.

[8] Cook, R. D. and M. E. Johnson (1981). A family of distributions for modelling non-elliptically symmetric multivariate data. *J. R. Stat. Soc. Ser. B. Stat. Methodol. 43*(2), 210–218.

[9] Cooke, R. M., H. Joe, and K. Aas (2010). Vines arise. In D. Kurowicka and H. Joe (Eds.), *Dependence Modeling*, pp. 37–71. World Scientific Publishing, Singapore.

[10] Cooke, R. M., D. Kurowicka, and K. Wilson (2015). Sampling, conditionalizing, counting, merging, searching regular vines. *J. Multivariate Anal. 138*, 4–18.

[11] Czado C. (2019). *Analyzing Dependent Data with Vine Copulas*. Springer, Cham.

[12] Fan, J., Q. Yao, and H. Tong (1996). Estimation of conditional densities and sensitivity measures in nonlinear dynamical systems. *Biometrika 83*(1), 189–206.

[13] Grömping, U. (2015). Variable importance in regression models. *Wiley Interdiscip. Rev. Comput. Stat. 7*(2), 137–152.

[14] Hobæk Haff, I., K. Aas, A. Frigessi, and V. Lacal (2016). Structure learning in Bayesian networks using regular vines. *Comput. Statist. Data Anal. 101*, 186–208.

[15] Holmes, M. P., A. G. Gray, and C. L. Isbell (2010). Fast kernel conditional density estimation: A dual-tree Monte Carlo approach. *Comput. Statist. Data Anal. 54*(7), 1707–1718.

[16] Hyndman, R. J., D. M. Bashtannyk, and G. K. Grunwald (1996). Estimating and visualizing conditional densities. *J. Comput. Graph. Statist. 5*(4), 315–336.

[17] Izbicki, R. and A. B. Lee (2017). Converting high-dimensional regression to high-dimensional conditional density estimation. *Electron. J. Statist. 11*(2), 2800–2831.

[18] Joe, H. (1996). Families of m-variate distributions with given margins and m(m-1)/2 bivariate dependence parameters. In L. Rüschendorf, B. Schweizer, and M. D. Taylor (Eds.), *Distributions with Fixed Marginals and Related Topics*, pp. 120–141. Institute of Mathematical Statistics, Hayward CA.

[19] Johnson, J. W. (2000). A heuristic method for estimating the relative weight of predictor variables in multiple regression. *Multivariate Behav. Res. 35*(1), 1–19.

[20] Kurowicka, D. and R. M. Cooke (2005). Distribution-free continuous Bayesian belief nets. In A. Wilson, N. Limnios, S. Keller-McNulty, and Y. Armijo (Eds.), *Modern Statistical and Mathematical Methods in Reliability*, pp. 309–322. World Scientific Publishing, Singapore.

[21] Kurowicka, D. and R. M. Cooke (2006). *Uncertainty Analysis with High Dimensional Dependence Modelling*. John Wiley & Sons, Chichester.

[22] Lipovetsky, S. and M. Conklin (2001). Analysis of regression in game theory approach. *Appl. Stoch. Models Bus. Ind. 17*(4), 319–330.

[23] Lundberg, S. M. and S.-I. Lee (2017). A unified approach to interpreting model predictions. In I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, pp. 4765–4774. Curran Associates, Red Hook NY.

[24] Nagler, T. and C. Czado (2016). Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *J. Multivariate Anal. 151*, 69–89.

[25] Nagler, T. and T. Vatter (2020). Solving estimating equations with copulas. Available at https://arxiv.org/abs/1801.10576.
[26] Nagler, T. and T. Vatter (2021). `rvinecopulib`*: High Performance Algorithms for Vine Copula Modeling*. R package version 0.5.5.1.1. Available on CRAN.
[27] Nguyen, M.-L. J. (2018). Nonparametric method for space conditional density estimation in moderately large dimensions. Available at https://arxiv.org/abs/1801.06477.
[28] Otneim, H. and D. Tjøstheim (2018). Conditional density estimation using the local Gaussian correlation. *Statist. Comput. 28*, 303–321.
[29] Owen, A. B. and C. Prieur (2017). On Shapley value for measuring importance of dependent inputs. *SIAM/ASA J. Uncertain. Quantif. 5*(1), 986–1002.
[30] Panagiotelis, A., C. Czado, and H. Joe (2012). Pair copula constructions for multivariate discrete data. *J. Amer. Statist. Assoc. 107*(499), 1063–1072.
[31] Rosenblatt, M. (1952). Remarks on a multivariate transformation. *Ann. Math. Statist. 23*(3), 470–472.
[32] Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist. 27*(3), 832–837.
[33] Sahin, E., C. J. Saul, E. Ozsarfati, and A. Yilmaz (2018). Abalone life phase classification with deep learning. In S. Deb, T. Hanne, and K.-C. Wong (Eds.), *Proceedings of the 5th International Conference on Soft Computing & Machine Intelligence*, pp. 163–167. Curran Associates, Red Hook NY.
[34] Schittenkopf C., G. Dorffner, and E. J. Dockner (2000). Forecasting time-dependent conditional densities: a semi-nonparametric neural network approach. *J. Forecast. 19*(4), 355–374.
[35] Sellereite, N. and M. Jullum (2020). shapr: An R-package for explaining machine learning models with dependence-aware Shapley values. *J. Open Source Softw. 5*(46), Article ID 2027, 3 pages.
[36] Shapley, L. S. (1953). A value for n-person games. In H.W. Kuhn and A.W. Tucker (Eds.), *Contributions to the Theory of Games*, pp. 307–317. Princeton University Press.
[37] Smith, J. S., B. Wu, and B. M. Wilamowski (2019). Neural network training with Levenberg-Marquardt and adaptable weight compression. *IEEE Trans. Neural Netw. Learn. Syst. 30*(2), 580–587.
[38] Song, E., B. L. Nelson, and J. Staum (2016). Shapley effects for global sensitivity analysis: Theory and computation. *SIAM/ASA J. Uncertain. Quantif. 4*(1), 1060–1083.
[39] Stöber, J., H. G. Hong, C. Czado, and P. Ghosh (2015). Comorbidity of chronic diseases in the elderly: Patterns identified by a copula design for mixed responses. *Comput. Statist. Data Anal. 88*, 28–39.
[40] Takahasi, K. (1965). Note on the multivariate Burr's distribution. *Ann. Inst. Statist. Math. 17*, 257–260.
[41] Štrumbelj, E. and I. Kononenko (2010). An efficient explanation of individual classifications using game theory. *J. Mach. Learn. Res. 11*(1), 1–18.
[42] Štrumbelj, E. and I. Kononenko (2014). Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst. 41*, 647–665.
[43] Wright, M. N. and A. Ziegler (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *J. Stat. Softw. 77*(1), 1–17.
[44] Yari, G. and A. M. D. Jafari (2006). Information and covariance matrices for multivariate Pareto (IV), Burr, and related distributions. *Int. J. Eng. Sci. 17*(3-4), 61–69.