# Vine Copula-Based Classifiers with Applications

Şahin, Özge; Joe, Harry

**DOI**
10.1007/s00357-024-09494-y

**Publication date**
2024

**Document Version**
Final published version

**Published in**
Journal of Classification

**Citation (APA)**
Şahin, Ö., & Joe, H. (2024). Vine Copula-Based Classifiers with Applications. *Journal of Classification*. https://doi.org/10.1007/s00357-024-09494-y

Journal of Classification
https://doi.org/10.1007/s00357-024-09494-y

ORIGINAL RESEARCH

# Vine Copula-Based Classifiers with Applications

Özge Şahin[1,2] · Harry Joe[3]

Accepted: 22 September 2024

**Abstract**

The vine pair-copula construction can be used to fit flexible non-Gaussian multivariate distributions to a mix of continuous and discrete variables. With multiple classes, fitting univariate distributions and a vine to each class lead to posterior probabilities over classes that can be used for discriminant analysis. This is more flexible than methods with the Gaussian and/or independence assumptions, such as quadratic discriminant analysis and naive Bayes. Some variable selection methods are studied to accompany the vine copula-based classifier because unimportant variables can make discrimination worse. Simple numerical performance metrics cannot give a full picture of how well a classifier is doing. We introduce categorical prediction intervals and other summary measures to assess the difficulty of discriminating classes. Through extensive experiments on real data, we demonstrate the superior performance of our approaches compared to traditional discriminant analysis methods and random forests when features have different dependent structures for different classes.

**Keywords** Classification · Copula · Feature selection · Prediction interval · Statistical learning · Vine

## 1 Introduction

When data consist of observations with feature vectors from different class labels, a classification method fits models to a training set and aims at assigning out-of-sample observations to their correct classes.

Among classification methods, discriminant analysis methods estimate the class densities of a feature vector given class labels and assign an observation to a class where the estimated density is the highest; probabilities for different classes at a given feature vector $\boldsymbol{x}^*$ are based on the relative values of the densities at $\boldsymbol{x}^*$ for the classes. These methods are useful when one does not have a useful way to visualize the data feature vectors in different classes based on projections. They partition the feature space into regions where different classes are more probable and can provide more interpretability than non-discriminant analysis methods.

✉ Özge Şahin
O.Sahin@tudelft.nl

[1] Delft Institute of Applied Mathematics, Delft University of Technology, Delft, The Netherlands

[2] Department of Mathematics, Technical University of Munich, Munich, Germany

[3] Department of Statistics, University of British, Columbia, Canada

Published online: 15 October 2024

Density estimation within classes is based on different assumptions, and linear/quadratic discriminant analysis based on the multivariate Gaussian assumption is one of the earliest methods. Non-discriminant analysis methods form probability prediction equations for a class as a function of the features; these include multinomial logistic regression and random forest.

The most flexible parametric method for the construction of multivariate non-Gaussian distributions with a mix of continuous and discrete variables is the vine pair-copula construction for the dependence structure combined with univariate models for each variable; references are Joe (2014); Panagiotelis et al. (2017); Czado (2019). Software, for example, R package `rvinecopulib` (Nagler & Vatter, 2022b), exists to handle continuous and discrete variables. In this paper, we use this construction to estimate the joint density in each class; we call this the vine copula-based classifier and combine the method with diagnostics to assess the difficulty of the classification problem. This includes categorical prediction intervals for three or more classes and misclassification rates based on these intervals. Summary measures of classification performance include area under the receiver operating characteristic (ROC) curve, out-of-sample negative log-likelihood score, and misclassification rates based on the prediction intervals.

In classification problems with many measured variables or features, there are likely to be some variables that are noisy or redundant. Further, including all measured features might not lead to the best classification performance based on predefined metrics in all classification methods. Hence, it is important to have variable or feature selection methods to avoid overfitting and worsening performance (Tang et al., 2014; Li et al., 2017). In this paper, we propose two feature selection methods. One is a filtering method based on a mutual information-based criterion without fitting densities to classes, while the other is a wrapper method using a sequential forward selection to add features to the vine models.

To illustrate our new methodology, we perform extensive analyses on a data set of music genres provided by Spotify because it is a large data set with ten genres and many features, with some quantitative and non-quantitative.

Some genres are easier to discriminate than others. Using different subsets of genres and features to compare methods, we illustrate the usefulness of diagnostics to show how and when vine classification can provide improved performance.

The rest of the paper is organized as follows: Sect. 2 introduces vine copula constructions, and Sect. 3 explains vine copula-based classifiers. We provide extensive data analyses in Sect. 4 and conclude with a discussion in Sect. 5.

## 2 Vine Copula Constructions

A $d$-dimensional copula is a $d$-dimensional distribution function with $U(0,1)$ univariate margins. Sklar's theorem (Sklar, 1959) allows to express a $d$-dimensional distribution in terms of a $d$-dimensional copula and its univariate margins. Let $\boldsymbol{X} = (X_1, \dots, X_d)^\top \in \mathbb{R}^d$ be a $d$-dimensional random vector with distribution $F$ and univariate marginal distributions $F_1, \dots, F_d$. The copula $C$ associated with $F$ satisfies $F(\boldsymbol{x}) = C(F_1(x_1), \dots, F_d(x_d))$. For absolutely continuous variables, the density $f$ can be written as

$$f(\boldsymbol{x}) = c\big(F_1(x_1), \dots, F_d(x_d)\big) \times f_1(x_1) \times \cdots \times f_d(x_d), \tag{1}$$

where $c$ is the $d$th order mixed partial derivative of $C$.

There are many bivariate parametric copula families that can cover the full range of positive dependence with a variety of tail dependence and asymmetry properties; some of these extend to the full range of negative dependence. However, there is no general approach to obtaining equations corresponding to $d$-variate parametric copula families with flexible dependence and tail properties.

The $d$-dimensional construction is based on sequential mixtures of univariate conditional distributions in Joe (1996), and Sections 3.8 and 3.9 of Joe (2014) use $d(d-1)/2$ bivariate copulas as building blocks. The theory of regular vines in Bedford and Cooke (2001, 2002) provides all possible sequences of mixing conditional distributions.

There is a graph, as explained in Section 3.9.2 of Joe (2014) and Section 5.2 of Czado (2019), associated with the construction known as a vine tree structure $\mathcal{V} = (T_1, \ldots, T_{d-1})$ on $d$ elements if it meets that (i) $T_1$ is a tree with node set $V_1 = \{1, \ldots, d\}$ and edge set $E_1$, (ii) $T_l$ is a tree with node set $V_\ell = E_{\ell-1}$ for $\ell = 2, \ldots, d-1$, and (iii) an edge is allowed to connect two nodes in $T_{\ell+1}$ if their associated edges in $T_\ell$ have a shared node in $T_l$. Using recursions from the construction, the joint density $f(\boldsymbol{x}) = f(x_1, \ldots, x_d)$ from the vine construction for $d$ continuous variables has the form:

$$f(\boldsymbol{x}) = \prod_{l=1}^{d-1} \prod_{e \in E_l} c_{e_a,e_b;D_e}\big(F_{e_a|D_e}(x_{e_a}|\boldsymbol{x}_{D_e}), F_{e_b|D_e}\big(x_{e_b}|\boldsymbol{x}_{D_e}\big)\big|\boldsymbol{x}_{D_e}\big) \prod_{p=1}^{d} f_p(x_p), \qquad (2)$$

where the edge $e = \{e_a, e_b, D_e\}$ has two conditioned variables indexed by $e_a$ and $e_b$ and a conditioning set of variables indexed in the set $D_e$. If the set $\{e_a, e_b\} \cup D_e$ has cardinality $\ell+1$, the edge is in tree $\ell$. The bivariate copula density $c_{e_a,e_b;D_e}$ associated with this edge is used to link the univariate conditional distributions $F_{e_a|D_e}$ and $F_{e_b|D_e}$. The conditional distribution function of $[X_{e_a}|\boldsymbol{X}_{D_e} = \boldsymbol{x}_{D_e}]$, which is $F_{e_a|D_e}(\cdot|\boldsymbol{x}_{D_e})$, is obtained via a recursion. For tree 1, $D_e$ is an empty set, and $F_{e_a|D_e}$ and $F_{e_b|D_e}$ are univariate marginal distributions.

For a pair-copula construction with a mix of discrete and continuous variables, there is a similar decomposition after replacing $c_{e_a,e_b;D_e}$ by $\tilde{c}_{e_a,e_b;D_e}$ with a slightly different form when coupling with one or two discrete variables; more details can be seen in Section 2.2 of Chang and Joe (2019). This has been implemented in, for example, R package `rvinecopulib` (Nagler & Vatter, 2022b).

For models for the $d$-dimensional density $f(\boldsymbol{x})$, parametric or non-parametric families can be used for each $c_{e_a,e_b;D_e}$, $e \in E_1 \cup \cdots \cup E_{d-1}$ and each $f_p$, $p \in \{1, \ldots, d\}$. Diagnostic plots can help decide on models for $f_p$ and $c_{e_a,e_b;D_e} = c_{e_a,e_b}$ for $e \in E_1$ (tree 1). Parametric models are typically used unless non-unimodality is seen in histograms, or unusual cloud shapes are seen in scatterplots.

For the estimation of Eq. 2 with a sample $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ of size $n$, a sequential approach (Joe & Xu, 1996) is used, starting with an estimation of univariate distributions. Then, transforms are made via the estimated univariate cumulative distribution functions: with $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{id})$, $u_{ip} = \widehat{F}_p(x_{ip})$ for a continuous $p$th variable and also $u_{ip}^- = \widehat{F}_p(x_{ip}^-)$ for a discrete $p$th variable, where we define $F_p(x_p^-) = \Pr(X_p < x_p)$. We refer to the $n$ resulting $\boldsymbol{u}$ vectors as copula data. Vine copula algorithms (e.g., Dißmann et al. 2013) decide on choices of copula families sequentially from tree 1, then tree 2, and so on. The estimated pair copulas in tree $\ell$ are used in higher-order trees.

**Example 2.1** *Consider three absolutely continuous random variables $X_1, X_2, X_3$ with respective densities $f_1, f_2, f_3$ and joint density $f_{123}$ associated with the copula density $c_{123}$. By a factorization, we have*

$$f_{123}(x_1, x_2, x_3) = f_{12|3}(x_1, x_2|x_3) f_3(x_3). \qquad (3)$$

*By Sklar's theorem, for* $x_3$ *in the support of* $F_3$*, there is a copula* $C_{12;3}(\cdot;\, F_3(x_3))$ *such that* $F_{12|3}(x_1, x_2|x_3) = C_{12;3}(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3);\, F_3(x_3))$ *because* $F_{12|3}(\cdot|x_3)$ *is a bivariate distribution with margins* $F_{p|3}(\cdot|x_3)$ *for* $p = 1, 2$*. The set of copulas* $\{C_{12;3}(\cdot;\, F_3(x_3))\}$ *summarizes the conditional dependence between the first and second variables given the third one. Differentiation with respect to* $x_1, x_2$ *leads to:*

$$f_{12|3}(x_1, x_2|x_3) = c_{12;3}(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3);\, F_3(x_3))f_{1|3}(x_1|x_3)f_{2|3}(x_2|x_3). \quad (4)$$

*Likewise, from copula* $C_{13}$ *and* $C_{23}$ *associated with* $F_{13}$ *and* $F_{23}$ *respectively, we can express* $f_{1|3}$ *and* $f_{2|3}$ *via* Eq. 1

$$f_{p|3}(x_p|x_3) = f_{p3}(x_p, x_3)/f_3(x_3) = c_{p3}(F_p(x_p), F_3(x_3))f_p(x_p), \quad p = 1, 2.$$

*Next, substitute the above into* Eq. 4 *and combine with* Eq. 3 *to get:*

$$f_{123}(\boldsymbol{x}) = c_{12;3}(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3);\, F_3(x_3)) \prod_{p=1}^{2} c_{p3}(F_p(x_p), F_3(x_3)) \prod_{p=1}^{3} f_p(x_p). \quad (5)$$

$$c_{123}(F_1(x_1), F_2(x_2), F_3(x_3)) = c_{12;3}(F_{1|3}(x_1|x_3), F_{2|3}(x_2|x_3);\, F_3(x_3)) \prod_{p=1}^{2} c_{p3}(F_p(x_p), F_3(x_3)). \quad (6)$$

*One can show that* $F_{1|3}(x_1|x_3) = \frac{\partial C_{1,3}(F_1(x_1), u_3)}{\partial u_3}|_{u_3=F_3(x_3)}$*. Further,* Eqs. 5 and 6 *can be constructed using a different factorization in* Eq. 3 *(e.g., condition on the first or second variable).*

*If these pair copulas are Gaussian, the vine copula corresponds to a Gaussian copula, and the pair copulas do not depend on the value of the conditioning variable (e.g.,* $c_{12;3}$ *in* Eq. 6 *satisfies* $c_{12;3}(v_1, v_2;\, F_3(x_3)) = c_{12;3}(v_1, v_2)$*). When using the vine copula as a parsimonious model and as a generalization of the Gaussian copula, the pair copulas in trees 2 and higher are assumed not to depend on the values of the conditioning variables. This is called the simplifying assumption.*

## 3 Classification Using Vine Copulas

For notation, we assume there are $k$ classes and $d$ variables or features. The number of features $d$ varies when a feature selection method is used. For the vine copula-based classifier, we use a training set from each class $j \in \{1, \ldots, k\}$ to estimate a density $f_j$ of form Eq. 2 with the simplifying assumption described in Sect. 2.

Section 3.1 discusses the importance of the univariate modeling step and introduces notation for the prediction probabilities over classes and the single-class prediction. There are a number of extra considerations and diagnostics when the number of classes $k$ is more than 2. Section 3.2 summarizes methods for feature selection, including our new wrapper method based on a vine copula-based classifier. Section 3.3 introduces categorical prediction intervals as a diagnostic to help understand which classes are harder to discriminate. Section 3.4 summarizes several performance measures that will be used in the data application in Sect. 4.

### 3.1 Prediction Probabilities After Estimation of Models by Class

It is important that good choices of univariate distributions are made for individual variables in each class and checked for adequacy of fit with Q-Q plots (for continuous variables). In previous applications, we have sometimes found the need for 3-parameter and 4-parameter families for variables with unimodal histograms and boxplots that show skewness and a variety of tail weights. These include skew-t (Jones & Faddy, 2003), skew-normal (Azzalini, 1985), and generalized gamma outlined in Section A.3 of Klugman et al. (2010). When no low-dimensional parametric family provides a good fit, kernel density estimation, e.g., R package `kde1d` (Nagler & Vatter, 2022a), can be used after variables have been transformed to avoid extreme skewness.

Assume the training data set consists of $n_j$ observations for class $j$. Let $\boldsymbol{x}_{p,j} = (x_{1,p,j}, \ldots, x_{n_j,p,j})^\top$ denote the $p$th feature vector in the $j$th class for $p = 1, \ldots, d$, and $j = 1, \ldots, k$. For the vine copula-based classifier, after the set of joint density function $\{\widehat{f_j}\}$ for $d$ features are separately estimated by class for a training set, posterior probabilities for a feature vector $\boldsymbol{x}^*$ are based on relative densities using prior class probabilities $\pi_j$:

$$\widehat{\pi}_j(\boldsymbol{x}^*) = \widehat{\Pr}(\text{class} = j|\boldsymbol{x}^*) = \pi_j \widehat{f_j}(\boldsymbol{x}^*) \Big/ \sum_{\ell=1}^{k} \pi_\ell \widehat{f_\ell}(\boldsymbol{x}^*), \quad j = 1, \ldots, k. \tag{7}$$

For a single class prediction, the vector $\boldsymbol{x}^*$ would be assigned to the class as

$$\gamma(\boldsymbol{x}^*) = \text{argmax}_j \widehat{\pi}_j(\boldsymbol{x}^*). \tag{8}$$

In most applications, it is best to use $\pi_j = k^{-1}$ to avoid majority classes in the data have higher prediction probabilities (such as with non-discriminant classification methods).

The theoretical counterpart of Eq. 7 with $\widehat{f_a}$ replaced by $f_a$ for $a \in \{1, \ldots, k\}$ is invariant to a common (over classes) monotone increasing transform of any variable because the Jacobian term for the transform cancels from the numerator and denominator. However, since Eq. 7 is not invariant to monotone transforms, good preprocessing transforms are needed with data to get more reliable density estimates.

Vine copula classifiers have been used in limited settings in some published articles. Chen (2014) restricts to the D-vine structure, which is a boundary class of regular vines and has a limited choice of parametric bivariate copula families; there is no detailed discussion of fitting univariate distributions. Carrera et al. (2016) also use the D-vine structure for a 5-class problem with a limited choice of parametric bivariate copula families and assume univariate Gaussian distributions for individual variables. Nagler and Czado (2016) have a 2-class example to illustrate the use of bivariate copulas based on kernel density estimation when scatterplots suggest that the usual parametric bivariate copula families are inadequate. Also, Schellhase and Spanhel (2018) showed the usage of non-simplified vine copulas in a 2-class classification. Carrera et al. (2019) also allows for all regular vines, but there are no details of exploratory data analysis and feature extraction based on images for their 2-class example of dunes and non-dunes.

The vine-based classification procedure can also be applied to a subset of $d$ available features, or one can start with vine copula constructions based on $3 \le d_1 < d$ features and add extra features if performance measures based on $\{\widehat{\pi}_j(\boldsymbol{x}^*)\}$, over $\boldsymbol{x}^*$ in a validation set, improve with additional features.

In addition to the above discussion of the estimation of vine copula models and univariate densities, there are issues with feature selection approaches and performance evaluations, especially for $k \ge 3$. These are discussed in the following subsections.

## 3.2 Feature Selection: Filtering and Wrapper

Some features might not be needed for classifiers, i.e., excluding them improves the performance and leads to more parsimony and greater computational efficiency. We develop two feature selection approaches for vine copula-based classifiers — we refer to these as a *filtering* method and a *wrapper* method, using the terminology of Guyon and Elisseeff (2003). The former uses the (conditional) mutual information of features given classes and does not depend on the classification method. The latter uses the classifier's accuracy at each iteration and performs sequential forward feature selection.

### Filtering

Filtering methods use the information in features without fitting any classification models. However, their performance varies by models and data sets (Bommert et al., 2020). We assess how well a filtering method does for vine copula-based classifiers for comparisons.

Let $X$ be a discrete or continuous feature with probability mass function or density $f_X$ and let $Y$ be a discrete class label random variable with probability mass function $f_Y$. Let their joint density be $f_{XY}$. The *mutual information* for $X$ and $Y$ is given by

$$\begin{aligned} I(X,Y) &= \int f_{XY}(x,y)\log\left(\frac{f_{XY}(x,y)}{f_X(x)f_Y(y)}\right)\mathrm{d}\nu(x,y) \\ &= \int f_{XY}(x,y)\log\left(\frac{f_{X|Y}(x|y)}{f_X(x)}\right)\mathrm{d}\nu(x,y), \end{aligned} \tag{9}$$

where $\nu$ is a product measure involving counting measure if $X$ is discrete and Lebesgue measure if $X$ is continuous. Equation 9 can be regarded as the amount of information the variable $X$ has about $Y$. For applications, Eq. 9 is simpler to compute as a double summation of the continuous variable $X$ discretized into an ordinal variable; otherwise, (kernel) density estimation is needed for $f_X$ and $f_{X|Y}$. Our experience is that the discretization tends to lead to smaller values but does not change the importance ranking of features. When $X$ and $Y$ are independent, e.g., the feature $X$ is uninformative for classification, Eq. 9 is zero.

Next, let $\mathbf{Z}$ be a random vector of discrete and continuous random variables with density $f_Z$ and suppose $(X, Y, \mathbf{Z})$ has the joint density $f_{XYZ}$ and marginal densities $f_{XZ}$ and $f_{XY}$. The conditional mutual information between $X$ and $Y$ given $\mathbf{Z}$ is

$$I(X,Y|\mathbf{Z}) = \int f_{XYZ}(x,y,z)\log\left(\frac{f_{XY|Z}(x,y|z)}{f_{X|Z}(x|z)f_{Y|Z}(y|z)}\right)\mathrm{d}\nu(x,y,z). \tag{10}$$

In comparison to the mutual information in Eq. 9, the conditional mutual information in Eq. 10 measures the information the variable $X$ provides about the variable $Y$ conditioned on $\mathbf{Z}$. If the set of features $\mathbf{Z}$ has been selected, Eq. 10 summarizes the information the additional feature $X$ provides about the class label $Y$. Similar to Eq. 9, for applications, Eq. 10 is simpler to compute via a multiple sum if the features have been discretized into ordinal variables.

A filtering method might stop adding features to a classifier when it reaches the specified number of features. Alternatively, since conditional mutual information avoids choosing strongly associated features, another approach for stopping can be adding all features $X$ whose conditional mutual information is higher than a specified threshold after $\mathbf{Z}$ reaches the dimension of 3.

We remark that different filtering methods, as analyzed in Bommert et al. (2020), can be applied and compared for vine copula-based classifiers.

### Wrapper

Filtering features for classification might lead to different performances by models since filtering does not account for model fits. To overcome such a drawback and tailor the feature selection to a classifier, wrapper methods are applied; they consider how a classifier performs sequentially and iteratively. In this section, we propose Algorithm 1 as a wrapper for selecting features in vine copula-based classifiers.

We assume that each variable has been transformed to remove extreme skewness (and orders of magnitude variability) before exploratory data analysis and applications of classification methods. For the $p$th variable with values $\boldsymbol{x}_p = (\boldsymbol{x}_{p,1}, \ldots, \boldsymbol{x}_{p,k})^\top$ over $k$ classes, we define

$$\boldsymbol{x}_p^s = (\boldsymbol{x}_{p,1}^s, \ldots, \boldsymbol{x}_{p,k}^s)^\top = [\boldsymbol{x}_p - \hat{\mu}_p]/\hat{\sigma}_p, \tag{11}$$

where $\hat{\mu}_p, \hat{\sigma}_p$ are respectively the sample mean and standard deviation of $\boldsymbol{x}_p$ (values of all $k$ classes). Next, a dissimilarity measure is defined and applied to these scaled values.

The algorithm depends on a dissimilarity measure $m$ for two vectors of data values $\boldsymbol{a}$ and $\boldsymbol{b}$. We will use the simply defined measure based on the sample quartiles:

$$m(\boldsymbol{a}, \boldsymbol{b}) = |\widehat{F}_a^{-1}(0.25) - \widehat{F}_b^{-1}(0.25) + \widehat{F}_a^{-1}(0.5) - \widehat{F}_b^{-1}(0.5) + \widehat{F}_a^{-1}(0.75) - \widehat{F}_b^{-1}(0.75)|/3, \tag{12}$$

where $\widehat{F}_a^{-1}$ and $\widehat{F}_b^{-1}$ are empirical quantile functions of $\boldsymbol{a}$ and $\boldsymbol{b}$, respectively. Thus, it measures the lack of overlap in the central 50% boxes of two side-by-side boxplots. If the central 50% box of one boxplot is nested within the central 50% box of the other, then this dissimilarity measure is small. Further, if there is a location shift from $\boldsymbol{a}$ to $\boldsymbol{b}$ so that quantiles shift by the same amount, $m(\boldsymbol{a}, \boldsymbol{b})$ is the same as the location shift. If the distribution of $\boldsymbol{a}$ is within the distribution of $\boldsymbol{b}$, and the two medians are the same, $m(\boldsymbol{a}, \boldsymbol{b})$ is close to 0.

**Example 3.1** *Suppose a feature from two classes* $\boldsymbol{x}_1 = (\boldsymbol{x}_{1,1}, \boldsymbol{x}_{1,2})^\top$ *has 20 observations with 8 from class 1 and 12 from class 2:* $\boldsymbol{x}_{1,1} = (4.7, 6.2, 9.0, 13.7, 3.8, 13.6, 14.2, 10.3)^\top$ *and* $\boldsymbol{x}_{1,2} = (9.8, 1.9, 3.9, 3.5, 10.6, 6.4, 11.8, 8.0, 11.0, 14.9, 6.3, 11.9)^\top$. *After scaling the feature of 20 observations, its 25%, 50%, and 75% quantiles from the first class are* $-0.737$, *0.219, and 1.212, respectively. The quantiles' average is 0.231. Those from the second class are* $-0.768$, *0.031, and 0.606, respectively. The average at the given quantile levels is* $-0.044$. *Then* Eq. 12 *leads to* $m(\boldsymbol{x}_{1,1}^s, \boldsymbol{x}_{1,2}^s) = 0.275$.

The algorithm initially chooses two features for classification and then adds further features based on a criterion that uses information on misclassified observations based on previously selected features.

Even though a random choice for the initial selection is possible, there is no guarantee that the selected features are useful for classification. Instead, we propose to select the starting features by assessing the dissimilarity of values for the $p$th feature across $k$ classes based on the evaluation of the sum $M_p$ of $m(\cdot, \cdot)$ in Eq. 12 over all pairs of classes, i.e., $M_p = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} m(\boldsymbol{x}_{p,i}^s, \boldsymbol{x}_{p,j}^s)$ where $\boldsymbol{x}_{p,i}^s$ and $\boldsymbol{x}_{p,j}^s$ represent the values of the scaled $p$th feature in classes $i$ and $j$, respectively, as given in Eq. 11.

The first two selected features have the largest two values of $M_p$. Over many classes, this choice matches the visualization of side-by-side boxplots of each feature over all classes. We may initially select more than two features, but starting with two ensures the classifier considers the pairwise dependence structure among features.

Additionally, the initial selection for our wrapper Algorithm 1 considers univariate dissimilarity across classes and might not reveal features that have marginal similarity across classes but still are important for classification, e.g., conditionally given another feature. For such cases, a filtering method can be considered.

However, Algorithm 1 is flexible in the dissimilarity measure. This measure could be defined in Eq. 12 or a modification, incorporating the sum of the absolute values of differences in the three quartiles. We recommend the following: first, examine some univariate and bivariate plots of variables by classes; then, develop a dissimilarity measure that can effectively quantify the class differences observed in these plots.

**Example 3.2** *Continuing from* Example 3.1, *assume that there are two more features with* $\boldsymbol{x}_2 = (\boldsymbol{x}_{2,1}, \boldsymbol{x}_{2,2})^\top$ *and* $\boldsymbol{x}_3 = (\boldsymbol{x}_{3,1}, \boldsymbol{x}_{3,2})^\top$. *The data vectors are:*
$\boldsymbol{x}_{2,1} = (14.1, 4.0, 10.1, 2.8, 4.7, 6.4, 1.2, 6.4)^\top$,
$\boldsymbol{x}_{2,2} = (13.2, 5.8, 7.7, 9.4, 7.9, 3.6, 12.6, 10.4, 12.1, 2.5, 11.1, 6.8)^\top$,
$\boldsymbol{x}_{3,1} = (12.5, 10.1, 12.0, 8.7, 8.4, 12.1, 1.3, 7.7)^\top$, *and*
$\boldsymbol{x}_{3,2} = (11.3, 10.7, 7.7, 13.1, 7.1, 4.4, 2.0, 2.4, 5.4, 8.3, 10.3, 6.7)^\top$. *To decide which two features should be selected initially among three (*$d = 3$*), our algorithm calculates the dissimilarity among classes through scaled features* $\boldsymbol{x}_p^s = (\boldsymbol{x}_{p,1}^s, \boldsymbol{x}_{p,2}^s)^\top$ *for* $p = 1, 2, 3$. *This leads to* $m(\boldsymbol{x}_{2,1}^s, \boldsymbol{x}_{2,2}^s) = 0.862$ *and* $m(\boldsymbol{x}_{3,1}^s, \boldsymbol{x}_{3,2}^s) = 0.626$. *Since* $\boldsymbol{x}_2$ *and* $\boldsymbol{x}_3$ *provide the highest dissimilarity across classes, they are selected as the initial two features.*

Vine copulas are then fitted to get $\widehat{f_j}$ for each class from which predicted class labels via Eq. 8 can be obtained, starting with the two selected features. Then, additional features are added one at a time until a stopping criterion holds.

Let the class label based on Eq. 8 be $y_i^{(t)}$ for the $i$th observation in iteration $t$, which is compared with the true labels $\{y_i\}$ for all observations. We can identify the misclassified observations in class $j$ and let $\boldsymbol{x}_{p,j}^{sMis,(t)}$ be the set of misclassified (scaled) values for the $p$th feature.

A $p$th feature with the large dissimilarity of $\boldsymbol{x}_{p,j}^{sMis,(t)}$ and $\boldsymbol{x}_{p,j'}^s$ for $j \neq j'$, based on $m(\cdot, \cdot)$ in Eq. 12, is considered as an addition.

The $p^{(t)}$th feature which maximizes such sum of dissimilarity over relevant pairs of classes is selected for the next vine copula models with the selected feature set $\mathcal{F}^{(t)}$. Hence, our wrapper algorithm aims to choose a feature through which our classifier may assign the misclassified observations of the previous iteration to their true class while preserving its current classification power.

Information or cross-validation-based criteria involving performance measures can be used to decide if a wrapper method-based classifier stops adding features.

**Example 3.3** *Continuing from* Example 3.2, *after fitting vine copula models using the second and third features, assume that four observations of class 1 are assigned to class 2, whereas others are assigned to their true class; suppose these have values 4.7, 6.2, 9.0, and 13.7 for the first feature. Their corresponding scaled values are denoted by* $\boldsymbol{x}_{1,1}^{sMis,(1)}$. *To decide if we select the first feature or others or stop, we measure the dissimilarity between the misclassified observations of class 1 and the observations of class 2 across the first feature, i.e.,* $m(\boldsymbol{x}_{1,1}^{sMis,(1)}, \boldsymbol{x}_{1,2}^s)$. *If we had the fourth feature that could be a candidate for the selection, we would measure* $m(\boldsymbol{x}_{4,1}^{sMis,(1)}, \boldsymbol{x}_{4,2}^s)$, *where* $\boldsymbol{x}_{4,1}^{sMis,(1)}$ *corresponds to the scaled fourth feature values of the misclassified observations of class 1. Then, if it holds* $m(\boldsymbol{x}_{1,1}^{sMis,(1)}, \boldsymbol{x}_{1,2}^s) > m(\boldsymbol{x}_{4,1}^{sMis,(1)}, \boldsymbol{x}_{4,2}^s)$, *the first feature would be selected and vice versa.*

**Algorithm 1** A wrapper feature selection method for vine-based classifiers.

```
Input: Training set $\boldsymbol{x}_p = (\boldsymbol{x}_{p,1}, \dots, \boldsymbol{x}_{p,k})^\top$ and the corresponding scaled data with the zero mean and unit variance $\boldsymbol{x}^s_p = (\boldsymbol{x}^s_{p,1}, \dots, \boldsymbol{x}^s_{p,k})^\top$ for $p = 1, \dots, d$, the total number of classes $k$, true class labels of observations $y_i$ for $i = 1, \dots, n$, a dissimilarity measure between two (univariate) distributions $m(.,.)$
Output: A vine copula-based classifier with selected features
Obtain the copula data
$\boldsymbol{u}_{p,j} = \widehat{F}_{p,j}(\boldsymbol{x}_{p,j})$ or $\boldsymbol{u}_{p,j} = \widehat{F}_{p,j}(\boldsymbol{x}^-_{p,j})$ for $p = 1, \dots, d$ and $j = 1, \dots, k$
Dissimilarity function for feature selection
$vineclassWrap(\boldsymbol{a}, \boldsymbol{b}, classLabelset_1, classLabelset_2)\{$
    $M \leftarrow 0$
    for $j \in classLabelset_1$ do
        for $j_2 \in classLabelset_2 \setminus j$ do
            $M \leftarrow M + m(\boldsymbol{a}[class == j], \boldsymbol{b}[class == j_2])$
        end for
    end for
    return $M$ }
Initialize selection of two features
$p_1 = \text{argmax}_{p=1,\dots,d}\, vineclassWrap(\boldsymbol{x}^s_p, \boldsymbol{x}^s_p, \{1, \dots, k\}, \{1, \dots, k\})$
$p_2 = \text{argmax}_{p=\{1,\dots,d\}\setminus p_1}\, vineclassWrap(\boldsymbol{x}^s_p, \boldsymbol{x}^s_p, \{1, \dots, k\}, \{1, \dots, k\})$
Set $\mathcal{F}^{(0)} \leftarrow \{p_1, p_2\}$ and $t \leftarrow 0$
Fit vine copula models & selection of other features
while a stopping criterion is not satisfied do
    Fit a vine copula model $\mathcal{V}^{(t)}_j$ to the data $\boldsymbol{u}_{\mathcal{F}^{(t)},j} = \{\boldsymbol{u}_{p,j} : p \in \mathcal{F}^{(t)}\}$ and get estimated density $\widehat{f}_j$ for $j = 1, \dots, k$.
    Estimate the class labels $y^{(t)}_i$ via Eq. 8 for the $i$th observation, $i = 1, \dots, n$.
    Identify misclassified observations in each class $j$, i.e., $y^{(t)}_i \neq y_i$ and $y_i = j$ for $i = 1, \dots, n$, and denote the scaled values for each feature $p$ by $\boldsymbol{x}^{sMis,(t)}_{p,j}$ for $j = 1, \dots, k$. Define $\boldsymbol{x}^{sMis,(t)}_p = (\boldsymbol{x}^{sMis,(t)}_{p,1}, \dots, \boldsymbol{x}^{sMis,(t)}_{p,k})^\top$.
    Identify class labels with the misclassified observations $J^{(t)} = \{j : \text{cardinality}(\boldsymbol{x}^{sMis,(t)}_{p,j}) > 0, \forall p\}$.
    Update $t \leftarrow t + 1$, $p^{(t)} = \text{argmax}_{p=\{1,\dots,d\}\setminus\mathcal{F}^{(t-1)}}\, vineclassWrap(\boldsymbol{x}^{sMis,(t)}_p, \boldsymbol{x}^s_p, J^{(t)}, \{1, \dots, k\})$.
    Check if a stopping criterion is satisfied. If not, update $\mathcal{F}^{(t)} \leftarrow \mathcal{F}^{(t-1)} \cup p^{(t)}$
end while
```

### 3.3 Categorical Prediction Intervals

Suppose there are $k$ classes and the prediction probability vector of a classification method with $d$ features for an out-of-sample case with feature vector $\boldsymbol{x}^*$ is $\widehat{\boldsymbol{\pi}} = \widehat{\boldsymbol{\pi}}(\boldsymbol{x}^*) = (\widehat{\pi}_1, \dots, \widehat{\pi}_k)$ for the $k$ classes based on Eq. 7. The point prediction of the class is based on the modal probability $y = y(\boldsymbol{x}^*) = \text{argmax}_j \widehat{\pi}_j$, as in Eq. 8. This summarization can lose information when comparing different methods because it does not take into account whether the modal probability is near 1 (easy discrimination) or near $1/k$ (difficult discrimination).

Prediction intervals, however, can be used to quantify the *uncertainty* in the classification. Any method that outputs prediction probabilities for different classes can be used to get modal predictions and prediction intervals. This also covers most machine learning methods, such as random forests. A longer categorical prediction interval indicates more difficulty for discrimination, implying more uncertainty.

Let $(j_1, \dots, j_k)$ be a permutation of $(1, \dots, k)$ so that $\widehat{\pi}_{j_1}(\boldsymbol{x}^*) \geq \widehat{\pi}_{j_2}(\boldsymbol{x}^*) \geq \cdots \geq \widehat{\pi}_{j_k}(\boldsymbol{x}^*)$. For $k^{-1} < 1 - \alpha < 1$ and level $1 - \alpha$, a $100(1 - \alpha)\%$ categorical prediction interval with $q$ classes derived from $\widehat{\boldsymbol{\pi}}(\boldsymbol{x}^*)$ is the smallest ordered set $J_q = (j_1, \dots, j_q)$ of classes for which $\sum_{r=1}^{q} \widehat{\pi}_{j_r}(\boldsymbol{x}^*) \geq 1 - \alpha$.

Let $\widehat{\pi}_\gamma$ be the modal probability. Consider 50% and 80% prediction intervals:

- If $\widehat{\pi}_\gamma \geq 0.80$, then the 80% and 50% prediction intervals are both $(\gamma)$.
- If $0.50 \leq \widehat{\pi}_\gamma < 0.80$, then the 50% prediction interval is $(\gamma)$ and the 80% interval is $(\gamma, j_2, \ldots)$, where additional classes are added based on the next fewest largest prediction probabilities in $\widehat{\boldsymbol{\pi}}$ to exceed 0.80.
- If $\widehat{\pi}_\gamma < 0.50$, then the 50% interval is $(\gamma, j_2, \ldots)$, where additional classes are added based on the next fewest largest prediction probabilities in $\widehat{\boldsymbol{\pi}}$ to exceed 0.50.

If the 50% interval has more than one class, it indicates that $\boldsymbol{x}^*$ is in the part of the feature space where more than one class have comparable density values. If the 80% interval has just one class, it indicates that $\boldsymbol{x}^*$ is in part of the feature space where one class has a much larger density than other classes.

### 3.4 Performance Measures

In this section, we summarize some performance measures used to compare different classification methods or the same method with different subsets of features.

One summary of the classification performance for two classes is through a receiver operating characteristic (ROC) curve. The curve illustrates the true positive rate on the $y$-axis versus the false positive rate on the $x$-axis at different thresholds by a classification method. For instance, a random guess corresponds to a point on the diagonal line, whereas the perfect classification is represented by a horizontal line at $y = 1$. The area under the ROC curve, called AUC, is one numeric summary of the classification method's performance. The multiclass AUC, called (m)AUC, is the average of the associated pairwise AUCs of the $k$ classes (Hand & Till, 2001). The hardest pair to discriminate can be identified by the index of the minimum of such AUCs. Nevertheless, the AUC does not reflect the uncertainty in the classification in contrast to categorical prediction intervals defined in Sect. 3.3; see Sect. 4.3.1 for examples.

Hence, we introduce misclassification rates and average lengths of categorical prediction intervals. While the latter represents the uncertainty and difficulty of the discrimination, the former reflects the misclassification rates at different coverage levels. Let $n_{j,test}$ be the number of cases in the test set with true class $j_i$ and $100(1-\alpha)\%$ categorical prediction interval $J_{1-\alpha,i}$ for the $i$th case. Let $n_{test} = \sum_{j=1}^{k} n_{j,test}$. For class $j$ at level $1-\alpha$, the misclassification rate is $[\sum_{i:j_i=j} I(j \notin J_{1-\alpha,i})]/n_{j,test}$, and the average length is $[\sum_{i:j_i=j} \texttt{cardinality}(J_{1-\alpha,i})]/n_{j,test}$. Further, the maximum misclassification rate with the corresponding class highlights which class is hardest to discriminate at different levels.

Another performance measure is the out-of-sample negative log-likelihood score (Czado et al., 2009) given by $-n_{test}^{-1} \sum_{i \in \text{test set}} \log \widehat{\pi}_{j_i}(\boldsymbol{x}_i^*)$ when there are $n_{test}$ cases in the test set, and the $i$th case has feature vector $\boldsymbol{x}_i^*$ and is in class $j_i$.

Note that there is also some loss of information from numerical summaries compared with tables of categorical prediction intervals.

## 4 Application: Spotify Song Genres

In this section, we apply our vine copula-based classifier and feature selection approaches to a real data set and compare their classification performance with competing methods.

The link https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre (visited on August 2022) has a data set that contains information about some songs on Spotify.

The original data set has 50,005 songs (observations), ten genres (classes), and 18 features (variables). The continuous feature, tempo, has around 10% missing observations, and another one, duration, does not change among songs, so we do not consider them in our analyses. After further data cleaning of missing observations and obvious errors, such as the negative popularity, there are 49,306 observations, eight remaining continuous features, and ten classes. The continuous features are *acousticness, dance, energy, liveness, loudness, popularity, speechiness, and valence*, while *instrumentalness* is transformed to be a discrete quantitative variable, and *mode* is a binary variable. Categorical features, like key and non-quantitative features, artist name, song title, song identification number, and data extraction date, are also not used in our analyses. Our interest is to see how well the genres can be classified based on quantitative features.

The ten classes are *Alternative, Anime, Blues, Classical, Country, Electronic, Hip-Hop, Jazz, Rap, and Rock*. More information describing the data set is in Appendix A.

For the detailed analyses, we take 50 random samples with $500 \times k$ observations, where $k$ denotes the number of genres in an analysis, over all continuous features to compare discriminant analysis methods with different number of genres regarding log-likelihood score, misclassification rate, and feature selection. Since the number of observations is similar across classes, our samples can be considered balanced. We work with continuous features for a simpler introduction to vine copula-based classifiers, but the vine method also works for a combination of continuous and discrete variables. While comparing our methods with random forests, we add the binary categorical variable, mode, and instrumentalness as a binned variable in our analyses.

We separate our data into learning, validation, and test sets with corresponding percentages of 60, 20, and 20. We call the combination of learning and validation sets training sets and use 80% of the data for training and 20% for testing.

We run all computations on 20 nodes CPU with Intel Xeon Platinum 8380 H Processor with around 8 GB RAM, running `R` version 4.2.2. A parallelization is applied to estimate classes' density by a discriminant method and select variables for random forests.

We remark that the posterior probabilities in test sets were well defined by the vine method, including the kernel density estimation; thus, the zero density when calculating Eq. 7 was avoided. Otherwise, the bandwidth parameters of the kernel density estimation should be adjusted as proposed by Nagler and Czado (2016).

Next, Sects. 4.1 and 4.2 summarize exploratory data analysis and vine copula-based classifier steps, respectively. For one random sample with a subset of genres, Sect. 4.3 has some comparisons of the vine copula-based classifiers with other discriminant methods and shows applications and interpretations of categorical prediction intervals. Section 4.4 compares vine copula-based classifiers with feature selection to random forests (a non-discriminant analysis method).

### 4.1 Exploratory Data Analysis

The number of observations ranges from 4459 in Classical to 5000 in Hip-Hop and Rap (see Table 7 in Appendix A). The estimated strength of the pairwise dependence measured by Kendall's tau among continuous features varies by a pair of features and genre, as shown in Table 8 in Appendix A.

Figure 2 in Appendix A shows that the features acousticness, liveness, loudness, and speechiness are skewed for Rock, and a similar result applies to other classes. Since the univariate density estimation can be affected by skewness, we transform them using the logit transform for acousticness, liveness, and speechiness and a cube root transform for loudness after converting to all negative combined over all classes. We scale the popularity into [0,1] by dividing the values by 100. The resulting transformed features are referred to as *tacoustic, tlive, tloud, tspeech, and tpopular* (see Fig. 3 in Appendix A).

The binary variable, mode, can help discriminate Country songs from others as seen in Table 9 in Appendix A. Moreover, even though instrumentalness can take values between zero and one, it is zero for 30% of the songs, and some genres, such as Hip-Hop and Rap, have a smaller proportion of zeros. Therefore, we bin as [0, 0.00001], (0.00001, 0.20], (0.20, 1], calling the associated discrete variable with three levels as *tinstrumental*. Table 9 in Appendix A shows that Anime, Blues, Classical, Electronic, and Jazz are different from other genres regarding the distribution of tinstrumental.

Some pairs of classes might be hard to discriminate, making the classification problem hard. For its initial overview, we propose a procedure described in Appendix A. Accordingly, the overlap measure matrix in Table 10 in Appendix A shows that Rap and Hip-Hop are hard to discriminate, whereas Rap and Anime might be well separated. Additionally, Alternative has the highest overlap measure summed over other classes, showing that discriminating it from others is harder. The reverse applies to Classical.

## 4.2 Step-By-Step: Vine Copula-Based Classifiers

### Selection and Estimation of Vine Copula Models

We estimate the univariate densities of continuous features with kernel density estimates using the R package kde1d (Nagler & Vatter, 2022a) for ease of coding with 80 (8 features × 10 genres) univariate distributions. Moreover, we empirically estimate the cumulative distribution functions of discrete variables and retain their left-sided and right-sided limits. Then, we obtain the corresponding copula data using the associated cumulative distribution functions.

We fit vine copula models to the copula data, following the approach (Dißmann et al., 2013) proposed for selecting and estimating vine copula models, as implemented in the R package rvinecopulib (Nagler & Vatter, 2022b). We use parametric pair-copula families available in the package.

### Feature Selection

We apply (conditional) mutual information defined in Eqs. 9 and 10 to filter features. For continuous variables $X$ and $\boldsymbol{Z}$, we compared two implementations. First, we discretize the variables into ordinal variables, where each category out of four has equal frequency for Eqs. 9 and 10, and allow the random vector $\boldsymbol{Z}$ to have one, two or three variables. Second, we have density estimation methods for Eqs. 9 and 10 using 1-dimensional and 2-dimensional numerical integration, respectively, with $\boldsymbol{Z}$ being a scalar in Eqs. 10. The latter is a check on the effect of discretization of continuous variables. The discretization approach, which is numerically simpler, leads to smaller (conditional) mutual information values, but the rankings of the next feature to add are the same in the two approaches. Thus, we continue with only the discretization for the filtering approach. We mainly use the filtering approach to choose three or four features because higher-dimensional tables become sparser.

We stop selecting further features with our wrapper algorithm if the selected feature in a given iteration does not improve the (m)AUC in the validation set. However, the improvement value in the (m)AUC can be taken as a tuning parameter. Alternative stopping criteria can be based on other performance measures in Sect. 3.4. Even though we use a single performance measure for the criterion for feature selection, we make comparisons of different subsets of features using more than one performance measure.

## 4.3 Comparison of Discriminant Analysis Methods

In this section, we compare four discriminant analysis methods for $\widehat{f_j}$ in Eqs. 7 using equal class probabilities $\pi_j$:

1. NB-gauss: naive Bayes with the misspecified assumption of stochastically independent features, with univariate Gaussian densities for each feature in each class as implemented in the `R` package `naivebayes` (Majka, 2019).
2. NB-kde: naive Bayes with the misspecified assumption of stochastically independent features, with univariate densities for each feature estimated via kernel density estimation as implemented in the `R` package `kde1d`.
3. QDA: quadratic discriminant analysis, with the misspecified assumption of multivariate normal density for each class as implemented in the `R` package `mclust` (Scrucca et al., 2016).
4. Vine: vine copula distributions, most flexible in handling non-Gaussian dependence.

### 4.3.1 Summaries for Categorical Prediction Intervals

For illustrations of categorical prediction intervals and comparisons of four discriminant analysis methods, the song genre data set allows us to take different subsets of genres and features. We have mainly considered detailed examples with four genres because summary tables of categorical prediction intervals take much more space with five or more genres.

If the chosen genres and features are quite different in side-by-side boxplots of features by genres, all four methods perform well, and there is not much to distinguish the methods; an example are with genres of Anime, Classical, Electronic, and Rap using features of tpopular, acousticness, and dance. If the selected features are weak in dependence, vine-based classifiers and naive Bayes with density estimation perform similarly, and the other two methods might be a little worse. If the dependence of the features seen in plots is quite different from multivariate Gaussian, then QDA performs worse than vine-based classification, as shown in Fig. 1 in Sect. 4.3.2 and Fig. 4 in the Appendix B.

The performance of the methods can be more greatly differentiated if the Spearman or Kendall correlation matrices have more variability over the selected features. With this latter criterion, we choose a random sample of 2000 observations among the four classes of Anime, Classical, Electronic, and Jazz using features of dance, energy, speechiness, and valence. The number of cases is 1600 for the training set and 400 for the test set. Details below show that the vine method is the best in this case.

#### Summary Tables for Anime, Classical, Electronic, and Jazz

Tables 1 and 2 summarize 50% and 80% categorical prediction intervals for the methods of vine and naive Bayes with density estimation in the classification of Anime, Classical, Electronic, and Jazz using features of dance, energy, tspeech, and valence. Tables for the other two methods are not included to save space. However, Table 3 compares the four discriminant analysis methods from the summary tables of prediction intervals. The tables for prediction

**Table 1** 50% prediction intervals (vine and naive Bayes with kernel density) for a test set of 400 and a training set of 1600 from a random subset with four genres A=Anime, C=Classical, E=Electronic, and J=Jazz, and using the features of dance, energy, speechiness, and valence

Vine: 50% prediction intervals

| Labels | A | AC | AE | AJ | C | CA | CJ | E | EA | EJ | J | JA | JC | JE | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 42 | 2 | 4 | 3 | 17 | 1 | 0 | 4 | 1 | 0 | 7 | 3 | 0 | 1 | 85 |
| C | 9 | 0 | 0 | 2 | 82 | 1 | 3 | 1 | 0 | 0 | 6 | 2 | 0 | 0 | 106 |
| E | 7 | 1 | 5 | 5 | 1 | 1 | 0 | 75 | 5 | 2 | 11 | 2 | 0 | 3 | 118 |
| J | 8 | 1 | 2 | 2 | 10 | 1 | 1 | 9 | 0 | 2 | 4 6 | 4 | 2 | 3 | 91 |
| Total | 66 | 4 | 11 | 12 | 110 | 4 | 4 | 89 | 6 | 4 | 70 | 11 | 2 | 7 | 400 |

Naive Bayes with density estimation: 50% prediction intervals

| Labels | A | AC | AE | AJ | C | CA | CJ | E | EA | EJ | J | JA | JC | JE | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 26 | 1 | 5 | 1 | 21 | 0 | 0 | 11 | 1 | 2 | 7 | 6 | 1 | 3 | 85 |
| C | 8 | 0 | 0 | 2 | 80 | 1 | 2 | 0 | 0 | 0 | 6 | 6 | 1 | 0 | 106 |
| E | 7 | 0 | 8 | 3 | 2 | 0 | 0 | 63 | 6 | 5 | 16 | 5 | 0 | 3 | 118 |
| J | 8 | 0 | 1 | 2 | 13 | 1 | 2 | 13 | 2 | 2 | 39 | 3 | 1 | 4 | 91 |
| Total | 49 | 1 | 14 | 8 | 116 | 2 | 4 | 87 | 9 | 9 | 68 | 20 | 3 | 10 | 400 |

An interval AC means that Anime has a modal probability less than 0.50, and the second largest posterior probability is for Classical, and the total of these posterior probabilities exceeds 0.50

**Table 2** 80% prediction intervals (vine and naive Bayes with kernel density) for the same set-up as in Table 1

Vine: 80% prediction intervals

| Labels | A | AC | ACE | ACJ | AE | AEC | AEJ | AJ | AJE | C | CA | CAE | CAJ | CJ | CJA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 10 | 1 | 2 | 1 | 25 | 2 | 3 | 5 | 2 | 9 | 6 | 0 | 2 | 1 | 0 |
| C | 4 | 1 | 0 | 0 | 2 | 0 | 0 | 1 | 3 | 65 | 5 | 0 | 3 | 11 | 2 |
| E | 0 | 0 | 1 | 0 | 8 | 0 | 3 | 4 | 2 | 0 | 1 | 1 | 0 | 0 | 0 |
| J | 2 | 0 | 0 | 1 | 6 | 0 | 1 | 3 | 0 | 4 | 2 | 0 | 1 | 4 | 1 |
| Total | 16 | 2 | 3 | 2 | 41 | 2 | 7 | 13 | 7 | 78 | 14 | 1 | 6 | 16 | 3 |

| Labels | E | EA | EAJ | ECJ | EJ | EJA | J | JA | JAC | JAE | JC | JCA | JE | JEA | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 2 | 2 | 1 | 0 | 0 | 0 | 0 | 9 | 0 | 1 | 0 | 0 | 0 | 1 | 85 |
| C | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 4 | 1 | 1 | 0 | 0 | 1 | 0 | 106 |
| E | 39 | 22 | 1 | 0 | 18 | 2 | 0 | 3 | 0 | 1 | 0 | 0 | 11 | 1 | 118 |
| J | 4 | 3 | 0 | 0 | 3 | 1 | 9 | 15 | 1 | 1 | 2 | 2 | 23 | 2 | 91 |
| Total | 45 | 27 | 2 | 1 | 21 | 3 | 10 | 31 | 2 | 4 | 2 | 2 | 35 | 4 | 400 |

Naive Bayes with density estimation: 80% prediction intervals

| Labels | A | ACE | AE | AEC | AEJ | AJ | AJC | AJE | C | CA | CAJ | CJ | CJA |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 0 | 1 | 29 | 0 | 1 | 1 | 0 | 1 | 18 | 3 | 0 | 0 | 0 |
| C | 1 | 0 | 5 | 1 | 0 | 0 | 2 | 1 | 69 | 5 | 1 | 5 | 3 |
| E | 0 | 0 | 9 | 1 | 5 | 0 | 1 | 2 | 1 | 1 | 0 | 0 | 0 |
| J | 0 | 0 | 8 | 0 | 0 | 3 | 0 | 0 | 7 | 4 | 2 | 0 | 3 |
| Total | 1 | 1 | 51 | 2 | 6 | 4 | 3 | 4 | 95 | 13 | 3 | 5 | 6 |

**Table 2** continued

| Labels | E | EA | EAJ | EJ | EJA | J | JA | JAC | JAE | JCA | JE | JEA | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 1 | 10 | 0 | 2 | 1 | 0 | 6 | 1 | 6 | 1 | 1 | 2 | 85 |
| C | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 1 | 4 | 1 | 1 | 0 | 106 |
| E | 17 | 25 | 3 | 25 | 4 | 0 | 4 | 0 | 5 | 0 | 14 | 1 | 118 |
| J | 1 | 2 | 1 | 11 | 2 | 3 | 12 | 1 | 1 | 1 | 26 | 3 | 91 |
| Total | 19 | 37 | 4 | 38 | 7 | 3 | 28 | 3 | 16 | 3 | 42 | 6 | 400 |

An interval ACJ means that Anime has a modal probability of less than 0.80, and the second and third largest posterior probability is for Classical and Jazz respectively and the sum of these three probabilities is needed to exceed 0.80

intervals can provide information about the ease or difficulty of class discrimination, as well as performance by class, and they also provide the uncertainty in discrimination by methods. This type of information is lost in overall performance measures, such as overall misclassification rate and average AUC. In other words, prediction intervals highlight which classes have densities that overlap more. Also, a summary of pairwise AUC and multiAUC is in Table 4 for comparison.

## Comparison of Methods Regarding Categorical Prediction Intervals

The summaries in Tables 1 to 3 show, for the case of these four genres and four features, that vine-based classifier is the best method, followed by QDA, while the two naive Bayes methods perform much worse. Likewise, in other cases, such as with genres of Anime, Hip-Hop, Rap, Rock, and features with moderate non-Gaussian dependence, the vine method is overall the best, and QDA is worse than that based on misclassification rates. The pair (Hip-Hop, Rap) turns out to be the pair of genres that is most difficult to distinguish. When this pair is included in a subset of four genres, it can lead to some cases of 80% intervals that have all four genres (that is, all class prediction probabilities near 0.25). More details of such comparison are given in Sect. 4.3.2. Nonetheless, for the genres of Anime, Classical, Electronic, and Jazz with the selected features, the 80% categorical prediction intervals have at most three genres.

It is rare for one method to dominate another for performance measures in all classes. In Tables 1 to 3, the vine method has a shorter average prediction interval length and smaller

**Table 3** Summary statistics based on tables like in Tables 1 and 2 for all four methods

| | 50% Misclass | | | | 50% Avglen | | | | 80% Misclass | | | | 80% Avglen | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | vine | nbke | nbga | QDA | vine | nbke | nbga | QDA | vine | nbke | nbga | QDA | vine | nbke | nbga | QDA |
| A | 0.341 | 0.529 | 0.553 | 0.271 | 1.18 | 1.24 | 1.37 | 1.22 | 0.141 | 0.259 | 0.282 | 0.176 | 1.93 | 1.94 | 1.98 | 1.79 |
| C | 0.189 | 0.208 | 0.179 | 0.226 | 1.08 | 1.11 | 1.06 | 1.13 | 0.160 | 0.170 | 0.160 | 0.160 | 1.44 | 1.47 | 1.31 | 1.43 |
| E | 0.237 | 0.280 | 0.237 | 0.280 | 1.20 | 1.25 | 1.30 | 1.13 | 0.068 | 0.059 | 0.076 | 0.102 | 1.77 | 2.03 | 2.03 | 1.71 |
| J | 0.341 | 0.418 | 0.495 | 0.352 | 1.20 | 1.20 | 1.22 | 1.19 | 0.231 | 0.242 | 0.264 | 0.209 | 1.91 | 2.03 | 2.04 | 1.88 |
| avg | 0.277 | 0.359 | 0.366 | 0.282 | 1.16 | 1.20 | 1.23 | 1.17 | 0.150 | 0.182 | 0.196 | 0.162 | 1.76 | 1.87 | 1.84 | 1.70 |

The abbreviations are Misclass = misclassification rates for 50% and 80% intervals and Avglen = average length of 50% and 80% intervals. Smaller misclassification rates are better. The last row has averages over the first four rows. Abbreviations for the methods are: nbke for NB-kde; nbga for NB-gauss

**Table 4** Pairwise and multi AUC, (m)AUC, for a random subset with 1600 in training set and 400 in a test set with four genres: A=Anime, C=Classical, E=Electronic, and J=Jazz

| Method | Pairwise AUC | | | | | | mAUC |
|---|---|---|---|---|---|---|---|
| | AC | AE | AJ | CE | CJ | EJ | multi |
| vine | 0.862 | 0.856 | 0.811 | 0.984 | 0.892 | 0.817 | 0.873 |
| nbke | 0.833 | 0.746 | 0.745 | 0.981 | 0.893 | 0.754 | 0.828 |
| nbga | 0.806 | 0.689 | 0.684 | 0.967 | 0.881 | 0.735 | 0.797 |
| QDA | 0.835 | 0.838 | 0.812 | 0.973 | 0.885 | 0.819 | 0.862 |

misclassification rates compared with naive Bayes methods for at least three of the four genres. For vine versus QDA, there is some trade-off that one method is better than the other, depending on the genre. For performance measures of average misclassification rates and prediction interval lengths, the vine-based classifier performs better than QDA, except for an average length of 80% prediction intervals. Table 4 has condensed information from pairwise AUC values. It suggests that the vine-based classifier is the best regarding the (m)AUC and discriminates pairs of genres better than the other methods with pairwise AUCs, except for the pairs (Classical, Jazz), (Electronic, Jazz), and (Anime, Jazz), where the difference of pairwise AUC values for the vine and the best methods are very small.

The summaries in the tables show that there are cases where Anime, Electronic, and Jazz are difficult to distinguish. For example, in the 80% intervals, the set $\{A, E, J\}$ as a triplet in one of the six orders occurs in a fraction of $27/400 = 0.0675$ in the test set with the vine method, and there is a fraction of $168/400 = 0.420$ with a subset of two of these three genres. Hence, Classical is easier to distinguish among the four genres.

On inspection, all methods can predict over 80% on an incorrect class, indicating that some parts of the feature space are difficult for classification. The vine method and QDA are best based on tendency or shorter intervals with fewer misclassifications.

Further analyses with other subsets of genres and features show that including more features in discriminant analysis is sometimes better for all performance measures. This motivates our algorithms in Sect. 3.2; see Sects. 4.3.2 and 4.4 for more illustrations of feature selection. Also note that the NB-kde, NB-gauss, and QDA performance would be worse if transforms were not made for the features with skewed and bounded distributions.

### 4.3.2 Comparison of Methods Regarding Log-Likelihood Score, Misclassification Rate, and Feature Selection

#### Four Genres' Discrimination

In classifying Anime, Hip-Hop, Rap, and Rock, which include non-Gaussian dependence among some pairs of features and pairs of classes being easy, moderate, and hard to discriminate as given in Table 10 in Appendix A, Fig. 1 illustrates that the vine method with the four filtered features estimates higher posterior probabilities in the correct classes than the others as reflected in its lower negative log-likelihood scores. The reverse applies to QDA using all features. Moreover, applying feature selection, on average, results in more certainty toward the true class assignment of the observations, i.e., higher estimated posterior probabilities, for all methods. The average negative log-likelihood score is 267.61, with a standard deviation of 20.38, using the vine method and all features.

The vine method with feature selection provides lower misclassification rates for the 50% prediction interval than using all features, at least in 75% of the replicates. Even though

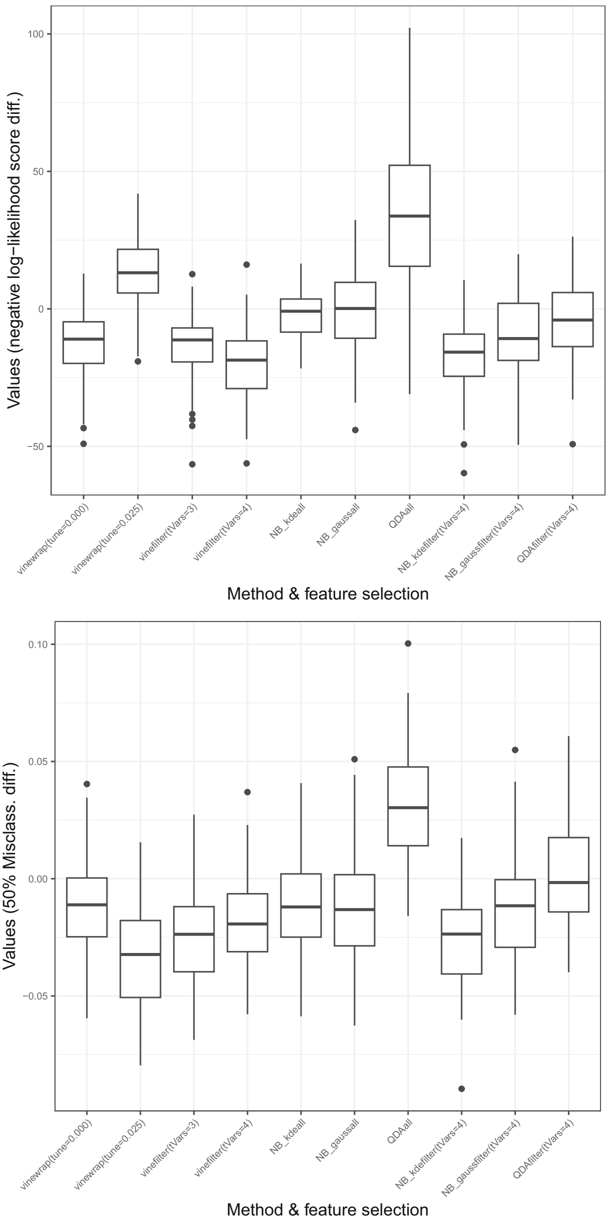

**Fig. 1** Comparison of the classification performance measures of the methods on test sets in classifying Anime, Hip-Hop, Rap, and Rock with the vine method using all continuous features out of 50 replicates. The value per replicate in the $y$-axis is calculated by the difference of the given method&feature selection from that of the vine method using all features. vinefilter(tVars=3/tVars=4): vine method using filtered three/four features, vinewrap(tune=0.000/tune=0.025): vine method using wrapper feature selection with 0.000/0.025 tuning parameter (improvement in mAUC), NB_kdefilter(tVars=4)/NB_gaussfilter(tVars=4)/QDAfilter(tVars=4): naive Bayes with univariate Gaussian densities/kernel density estimates/quadratic discriminant analysis using filtered four features, NB_kdeall/NB_gaussall/QDAall = naive Bayes with univariate Gaussian densities/kernel density estimates/quadratic discriminant analysis using all features. Smaller values of the negative log-likelihood score and misclassification rate are better

our wrapper algorithm with the non-zero tuning parameter, which checks the improvement in mAUC in a validation set to stop adding features, usually has the worst log-likelihood among the vine methods, it has better misclassification rates for the 50% prediction interval. Accordingly, the average length for the 50% prediction interval is, on average, the highest for our wrapper algorithm with the non-zero tuning parameter.

It shows that its estimated posterior probabilities for the true classes are lower than the other methods, but it better identifies the feature space where the classes have similar density values. Using all features with the vine method leads to the average misclassification rate (for the 50% prediction interval) of 0.28 with a standard deviation of 0.02.

NB-kde has similar log-likelihood scores and misclassification rates for the 50% prediction interval as NB-gauss using all features. However, the feature selection provides better measures for NB-kde than NB-gauss. Also, modeling a Gaussian dependence for a set of features, i.e., QDA, provides worse results than assuming independence, i.e., NB-gauss. Since pairwise plots of features suggest that the Gaussian dependence assumption does not hold, mostly with Anime or Hip-Hop, such a result might be expected.

As another classification performance measure, the maximum misclassification rate for the 50% prediction interval is the lowest for our wrapper algorithm with the non-zero tuning parameter, discriminating the classes better. Out of 50 replicates, the wrapper finds the class with the maximum misclassification rate as Hip-Hop and Rap in 27 and 23 replicates, respectively. However, it has a mean of 0.48 and a standard deviation of 0.06, showing that a random guess may be better sometimes for the hardest class discrimination. Still, it is the best method regarding this measure, among others considered.

Table 11 in Appendix B lists the selected feature indices by our filtering and wrapper methods. The first two selected features, which provide, on average, more univariate dissimilarity across classes, are tpopular and tspeech in all replicates. Figure 3 in Appendix A supports that Anime songs are less popular than Rock and Rap songs. Moreover, Hip-Hop and Rap songs contain fewer words than Rock songs. Hence, Anime and Rock are marginally well separable from the others and each other through the initial two features. Hip-Hop and Rap are the hardest to discriminate from each other, but the two features provide more dissimilarity than others.

Then, our methods always choose the feature dance, which has the highest conditional mutual information with the class label given tpopular and tspeech. In addition, the misclassified observations by the vine method using tpopular and tspeech are classified better by adding the third feature, dance, into the method than by adding the others. Next, the most selected fourth features by the filtering and wrapper methods are valence (32 replicates) and energy (23 replicates), respectively.

### Ten Genres' Discrimination

Figure 4 in Appendix B shows that vine copula-based classifiers using all features or selected features by the wrapper method of the zero tuning parameter, which selects features until mAUC does not improve in a validation set, discriminate ten genres better than the others regarding the negative log-likelihood score. Similar results about the misclassification rates of the 50% prediction intervals hold for the ten genres' discrimination, except that vine methods using all features have smaller rates than the Naive Bayes methods using all features. Thus, as the number of genres to classify increases, the vine methods' advantage over Naive Bayes increases.

### 4.4 Comparison of a Non-Discriminant Method: Random Forests

Fernández-Delgado et al. (2014) conclude that random forest-based approaches outperform hundreds of classifiers regarding classification accuracy, including neural networks, and Speiser et al. (2019) point out that the feature selection method proposed by Genuer et al. (2010) for random forests, called VSURF, usually performs better than other methods. Thus, we compare our vine copula-based classifier with random forests implemented in the `R` packages `randomForest` (Liaw & Wiener, 2002) and `VSURF` (Genuer et al., 2019). We work with the packages' default specifications.

For ten genres and eight continuous features, a vine-based classifier and random forest (using all features) discriminate Classical better than the other genres as shown in Table 5, consistent with our explanatory data analysis. The methods' misclassification rate for the 50% prediction interval for Classical and Rock is, on average, less than 17% and 24% percent, respectively. However, the associated average length for Classical is less than 1.26 for both methods, that for Rock is 2.19 for vines and 2.64 for random forest. Thus, the methods are more uncertain in the discrimination of Rock than that of Classical. In over half of the replicates, the random forest assigns the Rock's observations into three genres in the 50% prediction intervals. We remark that there is a trade-off of shorter prediction intervals and slightly higher misclassification rates in the methods' comparison.

Using eight continuous features, the average (m)AUC performances of random forests and vine copula-based classifiers are similar in the discrimination of four and ten genres in Table 6. Nevertheless, the vine method using the wrapper with a tuning parameter of zero is the best performer regarding the (m)AUC in both analyses. That method adds features in vine models until no further improvement in mAUC in a validation set is present. The same result applies while considering the minimum pairwise AUC. Further, all vine methods have a better minimum pairwise AUC than random forests. Thus, it shows that the vine methods discriminate the two classes {Hip-Hop, Rap} having similar density values better than random forests. Further, the feature of tlive was selected neither by our wrapper with zero tuning parameter nor by the random forest-specific variable selection, VSURF, in our replicates for the discrimination of ten genres. Thus, the tlive feature need not be recorded

**Table 5** Average summary statistics using the eight continuous features in ten genres' discrimination out of 50 replicates

| Genre | 50% Misclass | | 50% Avglen | |
|---|---|---|---|---|
| | vine | RandomForest | vine | RandomForest |
| Alternative | 0.33 (0.04) | 0.25 (0.05) | 1.75 (0.08) | 2.31 (0.08) |
| Anime | 0.29 (0.04) | 0.22 (0.05) | 1.15 (0.03) | 1.38 (0.07) |
| Blues | 0.47 (0.05) | 0.34 (0.04) | 1.43 (0.06) | 1.86 (0.05) |
| Classical | 0.16 (0.04) | 0.13 (0.04) | 1.09 (0.03) | 1.25 (0.05) |
| Country | 0.40 (0.06) | 0.35 (0.06) | 1.66 (0.07) | 2.15 (0.07) |
| Electronic | 0.40 (0.05) | 0.32 (0.05) | 1.40 (0.05) | 1.87 (0.08) |
| Hip-Hop | 0.40 (0.05) | 0.27 (0.06) | 1.43 (0.06) | 1.83 (0.07) |
| Jazz | 0.47 (0.05) | 0.36 (0.05) | 1.52 (0.05) | 2.00 (0.06) |
| Rap | 0.51 (0.05) | 0.34 (0.07) | 1.52 (0.07) | 1.89 (0.08) |
| Rock | 0.23 (0.04) | 0.16 (0.04) | 2.19 (0.07) | 2.64 (0.08) |

The numbers in parentheses are the corresponding empirical standard errors. The abbreviations are Misclass = misclassification rates and Avglen = average length of 50% intervals. Smaller misclassification rates are better

**Table 6** Comparison of the average multiclass-area under the curve, (m)AUC, the minimum pairwise AUC measures, and the number of selected features (#SelFeat.) of the random forest and the vine methods on test sets in classifying four (Anime, Hip-Hop, Rap, Rock) and ten classes out of 50 replicates

| Feature set | Feature selection | #class | #SelFeat | (m)AUC | | min. pairwise AUC | | neg.log-lik. score | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | RandomForest | vine | RandomForest | vine | RandomForest | vine |
| All | – | 10 | 8.00 (0.00) | **0.90 (0.01)** | **0.90 (0.01)** | 0.50 (0.03) | **0.53 (0.03)** | 1280.0 (29.9) | 1339.0 (44.8) |
| eight | Filter(tVars=3) | | 3.00 (0.00) | – | 0.88 (0.01) | – | 0.52 (0.03) | – | 1427.4 (33.8) |
| continuous | Filter(tVars=4) | | 4.00 (0.00) | 0.87 (0.01) | 0.89 (0.01) | 0.50 (0.03) | **0.53 (0.03)** | 1467.4 (43.8) | 1361.2 (36.7) |
| features | wrap(tune=0.000) | | 6.72 (0.78) | – | **0.90 (0.01)** | – | **0.53 (0.03)** | – | 1332.3 (44.5) |
| | wrap(tune=0.025) | | 2.16 (0.37) | – | 0.85 (0.01) | – | 0.52 (0.03) | – | 1536.0 (54.0) |
| | VSURF | | 7.00 (0.35) | **0.90 (0.01)** | – | 0.50 (0.03) | – | **1278.7 (32.7)** | – |
| All | – | 4 | 8.00 (0.00) | 0.89 (0.01) | 0.89 (0.01) | 0.51 (0.03) | **0.55 (0.03)** | 252.9 (12.0) | 267.6 (20.3) |
| eight | Filter(tVars=3) | | 3.00 (0.00) | – | 0.89 (0.01) | – | **0.55 (0.03)** | – | 253.7 (17.2) |
| continuous | Filter(tVars=4) | | 4.00 (0.00) | 0.88 (0.01) | **0.90 (0.01)** | 0.51 (0.03) | **0.55 (0.03)** | 267.7 (17.6) | **248.0 (17.9)** |
| features | wrap(tune=0.000) | | 4.46 (1.22) | – | **0.90 (0.01)** | – | **0.55 (0.03)** | – | 255.2 (18.6) |
| | wrap(tune=0.025) | | 2.04 (0.20) | – | 0.88 (0.01) | – | **0.55 (0.03)** | – | 279.8 (16.7) |
| | VSURF | | 5.44 (1.37) | 0.89 (0.01) | – | 0.51 (0.04) | – | 259.8 (19.0) | – |
| tacoustic& energy& tloud | – | 4 | 3.00 (0.00) | 0.66 (0.02) | **0.70 (0.02)** | 0.49 (0.03) | **0.50 (0.02)** | 537.8 (18.3) | **492.0 (12.4)** |
| dance& tspeech& valence | – | 4 | 3.00 (0.00) | 0.78 (0.01) | **0.81 (0.01)** | 0.48 (0.03) | **0.50 (0.02)** | 431.9 (15.8) | **397.1 (14.3)** |

The best performance for each feature set and measure is highlighted. The numbers in parentheses are the corresponding empirical standard errors. (–) shows nonapplicability for song genre discrimination, and music experts should consider replacing it with another feature that can better discriminate between Hip-Hop and Rap.

VSURF has the best negative log-likelihood score in the discrimination of ten genres using eight continuous features. Thus, the VSURF discriminates the observations in their true classes with higher probabilities than the other methods. The same result holds for the vine method using the wrapper with a tuning parameter of zero among the vine-based classifiers for ten genre discrimination. However, the vine method with filtering four features is the best performer of the negative log-likelihood score in classifying four genres using eight continuous features. Moreover, VSURF performs better than filtering features with (conditional) mutual information for random forests in all cases regarding the three measures analyzed in Table 6.

The estimated dependence strength among pairs of features is mostly low in classes in Table 8 in Appendix A. However, as the strength increases, vine copula-based classifiers are more likely to outperform random forests similar to the prediction tasks, as shown in Sahin and Czado (2024), where the main interest is to predict a continuous response using vine copulas and selecting the relevant features. Thus, to analyze the impact of dependent features on random forests and the vine method, we run them using the three dependent features, tacoustic, energy, and tloud, in the discrimination of four genres. Then, the vine method, on average, discriminates classes better than random forests regarding the (m)AUC, minimum pairwise AUC, and negative log-likelihood scores as seen in Table 6. Even though three weakly dependent features, dance, tspeech, and valence, are considered in the discrimination, a vine-based classifier is better than a random forest. A reason might be that random forests need more features to discriminate the classes better.

While our wrapper with the zero tuning parameter selects 4.46 features, on average, in the discrimination of four classes, the number increases to 6.72 in the discrimination of ten classes out of eight features. Likewise, VSURF selects more features to discriminate ten classes than four classes. Hence, models in our data example need more features to discriminate more classes. Still, eliminating features improves the discrimination accuracy compared to using all features, as reflected in the negative log-likelihood score and minimum pairwise AUC by our wrapper algorithm with the zero tuning parameter.

On average, the computation time of vine and random forests is 4.19 and 1.90 s, respectively, using eight continuous features to classify ten genres. Moreover, in the same analysis, while the filtering on the vine takes less than one second, the wrapper runs around 9.52 s. Since the latter fits the vine models at each step, such computation time can be expected. For this case, the VSURF gives the highest average computation time of 38.20 s. Like our wrapper algorithm for vine-based classifiers, tailoring feature selection to a specific method comes with a trade-off of higher computational cost and better performances.

When considering two discrete variables, tinstrumental and mode, with eight continuous features, for the ten genres' discrimination in vine-based classifiers, the multi AUC given in Table 6 increases by 0.01 through the selected features by our wrapper but not through filtering and using all features. The former is because (conditional) mutual information does not identify tinstrumental and mode within the most important four features in most replicates. While the average number of selected variables by our wrapper slightly increases with the inclusion of tinstrumental and mode, the wrapper method does not select mode at all and replaces tacoustic with tinstrumental in most replicates. Since tacoustic differs for Classical, Jazz, and Electronic but tinstrumental is different for five genres as shown in Appendix A, such a result can be expected. Further, our wrapper methods slightly improve the negative log-likelihood scores by adding tinstrumental. But, compared to using only eight continuous features, modeling ten features together decreases the vine method's performance regarding the (m)AUC and negative log-likelihood score, on average. This shows the importance of selecting features for vine-based classifiers. Moreover, Appendix A shows that tinstrumental and mode have a similar distribution for Hip-Hop and Rap. Thus, adding tinstrumental and/or mode into vine-based classifiers does not impact their minimum pairwise AUC. Moreover, the misclassification rates still vary across genres, highlighting the distinct impact of tinstrumental on Blues, Jazz, and Electronic and that of mode on Country.

Discriminant analysis via the vine copula-based classifier is more interpretable than random forests. The vine copula-based classifier can lead to smooth functions of $\{\widehat{\pi}_j(\boldsymbol{x}) : \boldsymbol{x} \in$ feature space, $j \in \{1, \ldots, k\}\}$, and one could (maybe through projections) interpret different parts of the feature space that favors different classes. Random forests have good classification performance, but the conditional probability vector $(\widehat{\pi}_{RF}(j|\boldsymbol{x}) : j = 1, \ldots, k)$ is piecewise constant with a finite number of trees. Thus, if one attempts to study the behavior as a function of $\boldsymbol{x}$ by choosing out-of-sample vectors in a grid, one can see that the function can have many up-and-down jumps locally.

## 5 Conclusion

We propose a vine copula-based classifier, which is a flexible method for estimating a non-Gaussian multivariate density with a combination of continuous and discrete variables and selects relevant features for classification tasks with proposed filtering and wrapper methods. With estimated posterior probabilities, we define categorical prediction intervals for classification. Then, we extract additional summary statistics, such as the misclassification rate and the average length of 50% prediction intervals by class. Such statistics provide further insights into the classification problem, such as the pair of classes that are most difficult to discriminate.

We show that naive Bayes (NB) and quadratic discriminant analyses (QDA) methods perform worse when assumptions of independent features and/or Gaussian distributions are far from holding. Further, unlike the vine method, NB-gauss and QDA cannot work with a combination of continuous and discrete variables. On the other hand, NB-kde can be extended to having discrete variables by using estimated probability mass functions (or frequency table in the training set) to replace kernel density estimation.

In addition, in our Spotify data analysis, vine copula classifiers require additional features to discriminate between a larger number of classes effectively. However, eliminating certain features improves the overall classification performance compared to using all features. Moreover, we show that the pair of genres (Hip-Hop, Rap) is the most challenging to discriminate. Hence, music experts should consider incorporating an alternative feature to differentiate between Hip-Hop and Rap more effectively. The songs' popularity score (tpopular) and the number of spoken words (tspeech) are selected as important by vine methods in discriminating song genres: Anime, Hip-Hop, Rap, and Rock.

Further, in our data example, the vine method and random forests have comparable classification accuracy, but the discriminant analysis through vines leads to smoother prediction probabilities over the feature space. Moreover, random forests need more computational memory than the vine copula-based classifier since each fitted tree needs to be saved for the forests for out-of-sample use. Moreover, over- and under-sampling can be a problem for random forests, whereas vine copula-based classifiers can perform well with unbalanced data sets if the number of observations per class is adequate for their fits, e.g.,$\geq 300$, and the number of features $\geq 2$.

Through data analysis, our wrapper algorithm shows how tailoring feature selection to a specific classifier can improve classification performance despite the increased computational cost. Therefore, an embedded method for the vines, which integrates feature and model selection steps, or a hybrid feature selection for the vine copula-based classifiers, which combines the filtering and wrapper methods, may be proposed and compared with our methods. Also, different stopping criteria can be investigated from the wrapper method. Future research can focus on extending our concept of dissimilarity to bivariate distributions by defining a central 50% region based on multivariate depth measures.

As a limitation of our method, we remark that we have mainly tested our methodology for monotonically related variables. Further research is on the agenda for data set analyses with non-monotone relationships between variables.

## Appendix A: Data Description

Acousticness is a confidence measure of whether a song is acoustic. The higher the value, the higher the confidence that the song is acoustic. Dance shows how suitable a song is for dancing based on a combination of musical elements, including tempo, rhythm stability, beat strength, and overall regularity. The high dance values indicate a good danceable song. Energy represents a perceptual measure of intensity and activity, with high values associated with fast and loud songs. Instrumentalness shows whether a track has vocals, with low values corresponding to more words. Liveness describes the presence of an audience in the recording, where high values indicate more audiences. Loudness gives the loudness of a track in decibels. How many streams within a certain period a song has is reflected in its popularity. Mode represents the modality of a track, either as major with the value of one or as minor with the value of zero. Speechiness describes the presence of spoken words in a song. The values in [0.00, 0.33] likely represent songs with non-speech; [0.33, 0.66] describes songs that may contain music and speech; [0.66, 1.00] are songs with only spoken words. Valence is a musical positiveness conveyed by a song. The songs with high valence sound happier, while songs with low valence sound more negative. More details about the features are given by Spotify at the link https://developer.spotify.com/documentation/web-api/reference/get-audio-features (visited on August 2022). The range of continuous features, except for loudness and popularity, is [0,1], whereas loudness and popularity have values in [-48,4] and [1, 100], respectively.

**Table 7** The number of observations classified by genre in the data set

| Genre | Number of observations | Genre | Number of observations |
|---|---|---|---|
| Alternative | 4995 | Electronic | 4973 |
| Anime | 4979 | Hip-Hop | 5000 |
| Blues | 4969 | Jazz | 4973 |
| Classical | 4459 | Rap | 5000 |
| Country | 4959 | Rock | 4999 |

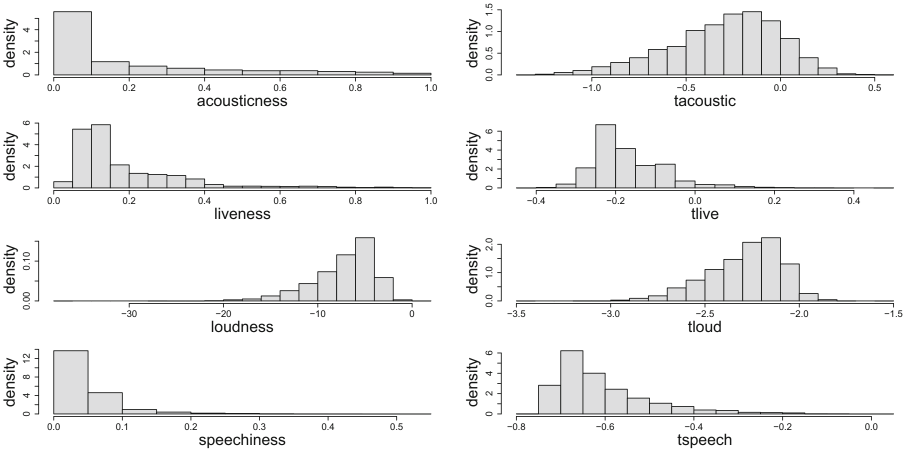

**Fig. 2** Histogram of acousticness (left), liveness (left), speechiness (left), loudness (left), and transformed acousticness (right), transformed liveness (right), transformed speechiness (right) using the logit transform and transformed loudness (right) using the cube root transform in Rock

**Table 8** Estimated Kendall's $\tau$ matrix of continuous features for Anime and Hip-Hop

| | Popularity | Acoust | Dance | Energy | Liveness | Loudness | Speech | Valence |
|---|---|---|---|---|---|---|---|---|
| Anime | | | | | | | | |
| Popularity | 1.000 | −0.078 | 0.036 | 0.110 | 0.021 | 0.153 | 0.029 | 0.069 |
| Acoust | −0.078 | 1.000 | −0.035 | −0.558 | −0.149 | −0.454 | −0.237 | −0.246 |
| Dance | 0.036 | −0.035 | 1.000 | 0.115 | −0.028 | 0.133 | −0.053 | 0.376 |
| Energy | 0.110 | −0.558 | 0.115 | 1.000 | 0.197 | 0.675 | 0.394 | 0.342 |
| Liveness | 0.021 | −0.149 | −0.028 | 0.197 | 1.000 | 0.171 | 0.102 | 0.090 |
| Loudness | 0.153 | −0.454 | 0.133 | 0.675 | 0.171 | 1.000 | 0.269 | 0.347 |
| Speech | 0.029 | −0.237 | −0.053 | 0.394 | 0.102 | 0.269 | 1.000 | 0.107 |
| Valence | 0.069 | −0.246 | 0.376 | 0.342 | 0.090 | 0.347 | 0.107 | 1.000 |
| Hip-Hop | | | | | | | | |
| Popularity | 1.000 | 0.022 | 0.037 | −0.023 | −0.035 | 0.053 | −0.064 | 0.016 |
| Acoust | 0.022 | 1.000 | −0.083 | −0.109 | −0.007 | −0.094 | 0.069 | 0.060 |
| Dance | 0.037 | −0.083 | 1.000 | −0.112 | −0.128 | −0.003 | −0.024 | 0.103 |
| Energy | −0.023 | −0.109 | −0.112 | 1.000 | 0.114 | 0.474 | 0.014 | 0.217 |
| Liveness | −0.035 | −0.007 | −0.128 | 0.114 | 1.000 | 0.045 | 0.074 | 0.020 |
| Loudness | 0.053 | −0.094 | −0.003 | 0.474 | 0.045 | 1.000 | −0.064 | 0.117 |
| Speech | −0.064 | 0.069 | −0.024 | 0.014 | 0.074 | −0.064 | 1.000 | 0.074 |
| Valence | 0.016 | 0.060 | 0.103 | 0.217 | 0.020 | 0.117 | 0.074 | 1.000 |

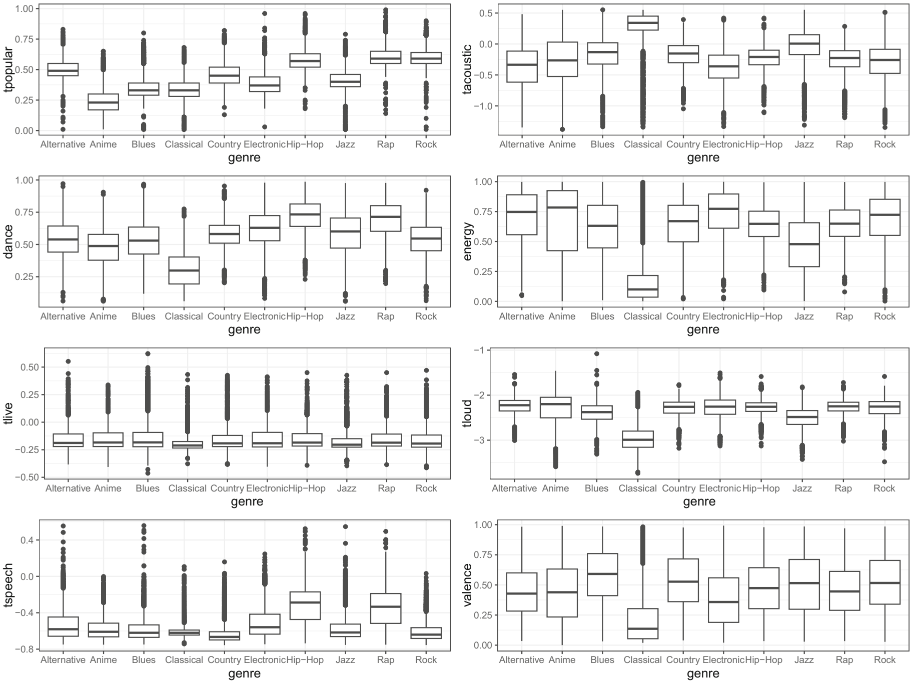

**Fig. 3** Boxplots of transformed continuous features in each class

**Table 9** Distribution of the discrete features, mode and tinstrumental, across genres; the genre names in Table 7 are shortened

| | Value | Alt | Ani | Blu | Cla | Cou | Ele | HHo | Jaz | Rap | Roc |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Mode | 0 | 3163 | 3087 | 3537 | 2955 | 4326 | 2677 | 2777 | 2633 | 2763 | 3697 |
| | 1 | 1832 | 1892 | 1432 | 1504 | 633 | 2296 | 2223 | 2340 | 2237 | 1302 |
| Tinstrumental | 1 | 2019 | 1883 | 1249 | 210 | 3273 | 416 | 3737 | 821 | 3745 | 2114 |
| | 2 | 2506 | 1410 | 3007 | 864 | 1652 | 2129 | 1181 | 1821 | 1193 | 2470 |
| | 3 | 470 | 1686 | 713 | 3385 | 34 | 2428 | 82 | 2331 | 62 | 415 |

### A.1 Simple Procedure to Evaluate if a Pair of Classes Is Hard to Discriminate

For a pair of classes $A$ and $B$, fit a logistic regression with response $I(\text{class} = B)$ and feature vector $\boldsymbol{x}$ for a training set to get logit $\left(\widehat{\Pr}(\text{class} = B)\right) = \hat{\beta}_{AB}^{\top}\boldsymbol{x}$. Consider two sets of linear predictions $\{\hat{\beta}_{AB}^{\top}\boldsymbol{x}_i \; : \; i \; \in \; \text{class } A\}$ and $\{\hat{\beta}_{AB}^{\top}\boldsymbol{x}_i \; : \; i \; \in \; \text{class } B\}$. With density estimate $g_A, g_B$ for these sets of projections to $(-\infty, \infty)$, the overlap of these two densities is $\int_{-\infty}^{\infty} \min\{g_A(z), g_B(z)\}\mathrm{d}z$. If such predictions are close to each other for two classes, classification is a hard task or the best separation of the classes in nonlinear.

**Table 10** Overlap measure matrix by genres whose names in Table 7 are shortened

| | Alt | Ani | Blu | Cla | Cou | Ele | HHo | Jaz | Rap | Roc |
|---|---|---|---|---|---|---|---|---|---|---|
| Alt | 0.000 | 0.106 | 0.240 | 0.092 | 0.522 | 0.370 | 0.308 | 0.333 | 0.325 | 0.426 |
| Ani | 0.106 | 0.000 | 0.451 | 0.259 | 0.159 | 0.290 | 0.022 | 0.242 | 0.022 | 0.031 |
| Blu | 0.240 | 0.451 | 0.000 | 0.200 | 0.401 | 0.360 | 0.067 | 0.574 | 0.062 | 0.114 |
| Cla | 0.092 | 0.259 | 0.200 | 0.000 | 0.100 | 0.142 | 0.018 | 0.263 | 0.017 | 0.045 |
| Cou | 0.522 | 0.159 | 0.401 | 0.100 | 0.000 | 0.343 | 0.160 | 0.477 | 0.168 | 0.343 |
| Ele | 0.370 | 0.290 | 0.360 | 0.142 | 0.343 | 0.000 | 0.154 | 0.393 | 0.133 | 0.133 |
| HHo | 0.308 | 0.022 | 0.067 | 0.018 | 0.160 | 0.154 | 0.000 | 0.129 | 0.862 | 0.195 |
| Jaz | 0.333 | 0.242 | 0.574 | 0.263 | 0.477 | 0.393 | 0.129 | 0.000 | 0.107 | 0.155 |
| Rap | 0.325 | 0.022 | 0.062 | 0.017 | 0.168 | 0.133 | 0.862 | 0.107 | 0.000 | 0.262 |
| Roc | 0.426 | 0.031 | 0.114 | 0.045 | 0.343 | 0.133 | 0.195 | 0.155 | 0.262 | 0.000 |

## Appendix B: Results

**Table 11** Selected continuous feature indices by filtering (top) and wrapper with the tuning parameter of zero (bottom) with the corresponding number of selections out of 50 replicates in classifying Anime, Hip-Hop, Rap, and Rock

| Feature indices | # sel |
|---|---|
| 1, 7, 3 | 50 |
| 1, 7, 3, 8 | 32 |
| 1, 7, 3, 4 | 14 |
| 1, 7, 3, 2 | 2 |
| 1, 7, 3, 6 | 2 |
| | |
| 1, 7, 3 | 13 |
| 1, 7, 3, 4, 8 | 11 |
| 1, 7, 3, 8 | 8 |
| 1, 7, 3, 4 | 5 |
| 1, 7, 3, 8, 4 | 3 |
| (1, 7, 3, 4, 6, 8), (1, 7, 3, 8, 2, 4, 6), (1, 7, 3, 2, 4) | 1 |
| (1, 7, 3, 4, 5), (1, 7, 3, 4, 5, 6), (1, 7, 3, 4, 6, 2) | 1 |
| (1, 7, 3, 4, 8, 2, 6, 5), (1, 7, 3, 8, 6, 2) | 1 |
| (1, 7, 3, 4, 8, 5), (1, 7, 3, 4, 8, 6, 5) | 1 |

Feature indices are (1) tpopular, (2) tacoustic, (3) dance, (4) energy, (5) tlive, (6) tloud, (7) tspeech, and (8) valence

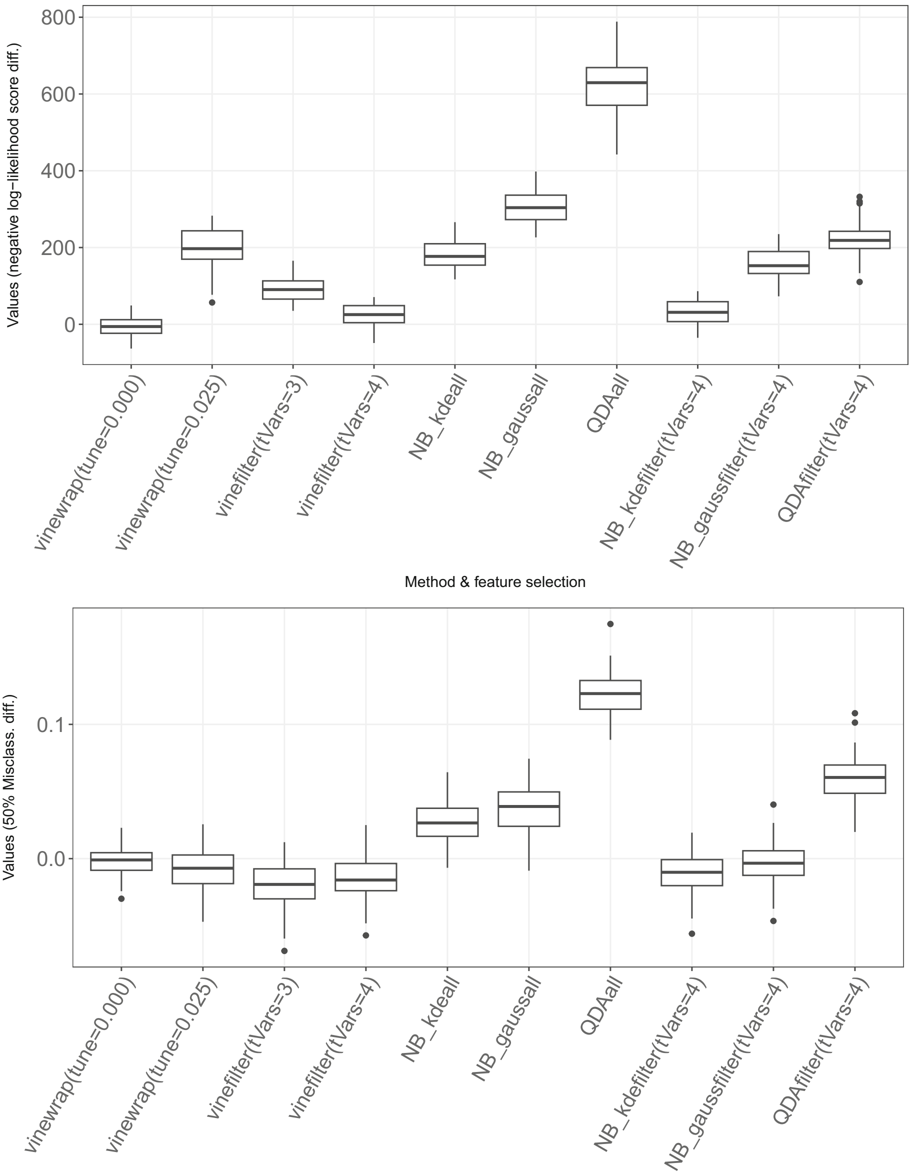

**Fig. 4** Comparison of the classification performance measures of the methods on test sets in classifying ten genres with the vine method using continuous features out of 50 replicates. The value per replicate in the *y*-axis is calculated by the difference of the given method&feature selection from that of the vine method using all features. vinefilter(tVars=3/tVars=4): vine method using filtered three/four features, vinewrap(tune=0.000/tune=0.025): vine method using wrapper feature selection with 0.000/0.025 tuning parameter, NB_kdefilter(tVars=4)/NB_gaussfilter(tVars=4)/QDAfilter(tVars=4): naive Bayes with univariate Gaussian densities/kernel density estimates/quadratic discriminant analysis using filtered four features, NB_kdeall/NB_gaussall/QDAall = naive Bayes with univariate Gaussian densities/kernel density estimates/quadratic discriminant analysis using all features. Smaller values of negative log-likelihood score and misclassification rate are better

**Acknowledgements** The project is supported by the German Research Foundation and a Mercator Fellowship (DFG grant CZ 86/6-1). Ozge Sahin acknowledges the support of the Global Challenges for Women in Math program by the Technical University of Munich. We are grateful for the discussions with Claudia Czado and Xinyao Fan. Thanks to the referees for their constructive comments leading to improvements.

**Funding** The project is supported by the German Research Foundation and a Mercator Fellowship (DFG grant CZ 86/6-1).

**Data Availability** The data used in the application is available at the website: https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre.

## Declarations

**Ethical Approval** This research contains no studies with human participation or animals performed by any authors.

**Conflict of Interest** The authors declare no competing interests.

## References

Azzalini, A. (1985). A class of distributions which includes the normal ones. *Scandinavian Journal of Statistics, 12*(2), 171–178.

Bedford, T., & Cooke, R. M. (2001). Probability density decomposition for conditionally dependent random variables modeled by vines. *Annals of Mathematics and Artificial Intelligence, 32*, 245–268. https://doi.org/10.1023/A:1016725902970

Bedford, T., & Cooke, R. M. (2002). Vines - A new graphical model for dependent random variables. *Annals of Statistics, 30*(4), 1031–1068. https://doi.org/10.1214/aos/1031689016

Bommert, A., Sun, X., Bischl, B., Rahnenführer, J., & Lang, M. (2020). Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis, 143*,. https://doi.org/10.1016/j.csda.2019.106839

Carrera, D., Bandeira, L., Santana, R., & Lozano, J. A. (2019). Detection of sand dunes on mars using a regular vine-based classification approach. *Knowledge-Based Systems, 163*, 858–874.

Carrera, D., Santana, R., & Lozano, J. A. (2016). Vine copula classifiers for the mind reading problem. *Progress in Artificial Intelligence, 5*, 289–305.

Chang, B., & Joe, H. (2019). Prediction based on conditional distributions of vine copulas. *Computational Statistics & Data Analysis, 139*, 45–63.

Chen, Y. (2014). A copula-based supervised learning classification for continuous and discrete data. *Journal of Data Science, 14*(4), 769–782.

Czado, C. (2019). *Analyzing dependent data with vine copulas. a practical guide with r*. Cham, Switzerland: Springer.

Czado, C., Gneiting, T., & Held, L. (2009). Predictive model assessment for count data. *Biometrics, 65*(4), 1254–1261.

Dißmann, J., Brechmann, E. C., Czado, C., & Kurowicka, D. (2013). Selecting and estimating regular vine copulae and application to financial returns. *Computational Statistics and Data Analysis, 59*, 52–69. https://doi.org/10.1016/j.csda.2012.08.010

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research, 15*(1), 3133–3181.

Genuer, R., Poggi, J.-M., Tuleau-Malot, C. (2010). Variable selection using random forests. *Pattern Recognition Letters*, *31*(14), 2225–2236. https://doi.org/10.1016/j.patrec.2010.03.014

Genuer, R., Poggi, J.- M., Tuleau-Malot, C. (2019). (R package version 1.1.0).

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1157–1182.

Hand, D. J., & Till, R. J. (2001). A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning, 45*, 171–186.

Joe, H. (1996). Families of $m$-variate distributions with given margins and $m(m-1)/2$ bivariate dependence parameters. L. Rüschendorf, B. Schweizer, and M.D. Taylor (Eds.), *Distributions with fixed marginals and related topics* (Vol. 28, p.120-141). Hayward, CA: Institute of Mathematical Statistics.

Joe, H. (2014). *Dependence modeling with copulas*. Boca Raton, FL: Chapman & Hall/CRC.

Joe, H., & Xu, J.J. (1996). The estimation method of inference functions for margins for multivariate models. *Technical Report no. 166, Department of Statistics, University of British Columbia*, 1–21. https://doi.org/10.14288/1.0225985

Jones, M.C., & Faddy, M.J. (2003). A skew extension of the t-distribution, with applications. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, *65*(1), 159–174.

Klugman, S. A., Panjer, H. H., & Willmot, G. E. (2010). *Loss models: From data to decisions* (3rd ed.). New York: Wiley.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature selection: A data perspective. *ACM Computing Surveys (CSUR), 50*(6), 1–45.

Liaw, A., & Wiener, M. (2002). Classification and regression by random forest. *R News, 2*(3), 18–22.

Majka, M. (2019). naivebayes: High performance implementation of the Naive Bayes algorithm in R. R package version 0.9.7.

Nagler, T., & Czado, C. (2016). Evading the curse of dimensionality in nonparametric density estimation with simplified vine copulas. *Journal of Multivariate Analysis, 151*, 69–89. https://doi.org/10.1016/j.jmva.2016.07.003

Nagler, T., & Vatter, T. (2022a)kde1d: Univariate Kernel Density Estimation. R package version 1.0.4.

Nagler, T., & Vatter, T. (2022). rvinecopulib: High performance algorithms for vine copula modeling. R package version 0.6.1.1.3.

Panagiotelis, A., Czado, C., Joe, H., & Stöber, J. (2017). Model selection for discrete regular vine copulas. *Computational Statistics & Data Analysis, 106*, 138–152. https://doi.org/10.1016/j.csda.2016.09.007

Sahin, Ö., & Czado, C. (2024). High-dimensional sparse vine copula regression with application to genomic prediction. *Biometrics*, *80*(1). https://doi.org/10.1093/biomtc/ujad042

Schellhase, C., & Spanhel, F. (2018). Estimating non-simplified vine copulas using penalized splines. *Statistics and Computing, 28*, 387–409.

Scrucca, L., Fop, M., Murphy, T.B., Raftery, A.E. (2016). Mclust 5: Clustering, classification and density estimation using Gaussian finite mixture models. *R Journal*, *8*(1), 289–317. https://doi.org/10.32614/rj-2016-021

Sklar, A. (1959). Fonctions de répartition à $n$ dimensions et leurs marges. *Publications de L'Institut de Statistique de L'Université de Paris, 8*, 229–231.

Speiser, J.L., Miller, M.E., Tooze, J., Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. *Expert Systems with Applications*, *134*, 93–101. https://doi.org/10.1016/j.eswa.2019.05.028

Tang, J., Alelyani, S., Liu, H. (2014). Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, 37.