On the consistency of supervised learning with missing values

Julie Josse, Jacob M. Chen, Nicolas Prost, Gaël Varoquaux, Erwan Scornet

Noname manuscript No.
(will be inserted by the editor)

On the consistency of supervised learning with missing values

Julie Josse · Jacob M Chen · Nicolas Prost · Gaël Varoquaux · Erwan Scornet

March 4, 2024

Abstract In many application settings, data have missing entries, which makes subsequent analyses challenging. An abundant literature addresses missing values in an inferential framework, aiming at estimating parameters and their variance from incomplete tables. Here, we consider supervised-learning settings: predicting a target when missing values appear in both training and test data. We first rewrite classic missing values results for this setting. We then show the consistency of two approaches, test-time multiple imputation and single imputation in prediction. A striking result is that the widely-used method of imputing with a constant prior to learning is consistent when missing values are not informative. This contrasts with inferential settings where mean imputation is frowned upon as it distorts the distribution of the data. The consistency of such a popular simple approach is

J. Josse
Centre de Mathématiques Appliquées, Ecole Polytechnique E-mail: julie.josse@inria.fr
Present address: INRIA-INSERM University of Montpellier

J.M. Chen
Williams College, Massachusetts

N. Prost
Centre de Mathématiques Appliquées, Ecole Polytechnique
Parietal project-team, INRIA Saclay

E. Scornet
Sorbonne Université and Université Paris Cité, CNRS,
Laboratoire de Probabilités, Statistique et Modélisation, F-75005 Paris, France

G. Varoquaux
Parietal project-team, INRIA Saclay

important in practice. Finally, to contrast procedures based on imputation prior to learning with procedures that optimize the missing-value handling for prediction, we consider decision trees. Indeed, decision trees are among the few methods that can tackle empirical risk minimization with missing values, due to their ability to handle the half-discrete nature of incomplete variables. After comparing empirically different missing values strategies in trees, we recommend using the "missing incorporated in attribute" method as it can handle both non-informative and informative missing values.

Keywords Bayes consistency, empirical risk minimization, decision trees, missing values, imputation, missing incorporated in attribute

1 Introduction

As volumes of data increase, they are harder to curate and clean. They may come from the aggregation of various sources (*e.g.* merging multiple databases) and contain variables of different natures (*e.g.* different sensors). Such heterogeneous data collections can lead to many missing values: samples only come with a fraction of the features observed. Though there is a vast literature on treating missing values, it focuses on estimating parameters and their variance in the presence of missing values in a single data set. In contrast, there are few studies of supervised-learning settings where the aim is to predict a target variable given input variables with missing entries. These settings only require to use *discriminative* (or conditional) models, compared to the first inferential frameworks, which often assume parametric data distributions (generative modelling). Besides, a predictive model is applied on a test set, different from the training set, a separation seldom considered in inferential settings. Therefore, inference and prediction in the presence of missing values are intrinsically two very different problems.

Beyond the aggregation of multiple sources, missing values can appear for a variety of reasons. For sensor data, missing values can arise from device failure. Informative missing values can be found for instance in poll data where participants may not

answer sensitive questions related to unpopular opinions. In medical studies, some measurements may be impractical on patients in a critical state, in which case the presence of missing values can be related to the variable of interest, target of the prediction (*e.g.* patient status). These various scenarios correspond to different missing-value mechanisms.

The classical literature on missing values, led by Rubin [1976], defines missing-values mechanisms based on the relationship between missingness and observed values: if they are independent, the mechanism is said to be Missing Completely At Random (MCAR); if the missingness only depends on the observed values, then it is Missing At Random (MAR); otherwise it is Missing Not At Random (MNAR). However, this nomenclature has seldom been discussed in the context of supervised learning, accounting for the target variable of the prediction.

Many statistical methods tackle missing values [Josse and Reiter, 2018, Mayer et al., 2022]. Listwise deletion, *i.e.* removing incomplete observations, may allow to train the model on complete data. However, it may lead to the deletion of almost all data especially in high dimension and may result in biased sample depending on the missing values mechanism. In addition, it does not suffice for supervised learning, as the test set may also contain incomplete data. Hence the prediction procedure should handle missing values. A popular solution is to impute missing values, that is to replace them with plausible values to produce a completed data set. The benefit of imputation is that it adapts existing pipelines and software to the presence of missing values. The widespread practice of imputing with the mean of the variable on the observed entries has serious drawbacks in inferential settings, as it distorts the joint and marginal distributions of the data which induces biased estimators [Little and Rubin, 2019]. Yet, few studies focus on the impact of constant imputation in a predictive setting. Imputation itself must be revisited for out-of-sample prediction settings: users resort to different strategies such as imputing separately the train and test sets or imputing them jointly. More elaborate strategies rely on using maximum likelihood with expectation maximization (EM)

to fit a model on incomplete data [Dempster et al., 1977, Little, 1992, Jiang et al., 2019]. However, such techniques often rely on strong parametric assumptions. Alternatively, some learning algorithms, such as decision trees, can readily handle missing values, accounting for their discrete nature.

In this paper, we study the classic tools of missing values in the context of supervised learning. We start in Section 2 by setting the notations and briefly summarizing the missing-value literature. Our first contribution, detailed in Section 3, is to adapt the formalism for missing values to supervised learning: we show how to use standard missing-values techniques to make predictions on a test set with missing values. Section 4 presents our main contribution: studying the consistency of two approaches to estimate the prediction function with missing values. The first theorem states that, given an optimal predictor for the completely-observed data, a consistent procedure can be built by predicting on a test set where missing entries are replaced by multiple imputation. The second theorem, which is the most striking and has important consequences in practice, shows that constant imputation prior to learning is consistent for supervised learning. This is, as far as we know, the first result justifying this very convenient practice of handling missing values. In Section 5, we compare imputation to learning directly with missing values via decision trees. Indeed, their greedy and discrete natures allow adapting them to handle missing values directly. We compare the different tree methods (together with classic machine learning approaches such as SVM or nearest neighbours) on simulated data with missing values and recommend to use the "Missing incorporated in attributes" (MIA, Twala et al. 2008) approach, whose good predictive performances have been highlighted by Kapelner and Bleich [2015], one of the few studies of trees with missing values for supervised learning. Other experimental works have shown that tree-based methods are competitive in terms of predictive performances [see Jäger et al., 2021]. We also show the benefits for prediction of an approach often used in practice, which consists in "adding the mask", *i.e.* adding binary variables that code for the missingness of each variables as new co-

variates, even though this method is not recommended for parameter estimation [Jones, 1996].

2 Definitions, problem setting, prior art

Notation. Throughout the paper, **bold** letters refer to vectors; CAPITAL letters refer to random variables, while lower-case letters are realisations. In addition, as usual, for any two variables A and B of joint density g,

$$g(b) \coloneqq g_B(b) \coloneqq \int g(\alpha, b)\, \mathrm{d}\mu(\alpha), \qquad g(a|b) \coloneqq g_{A|B=b}(a) \coloneqq \frac{g(a,b)}{g(b)}.$$

2.1 Supervised learning

Supervised learning is typically focused on learning to predict a *target* $Y \in \mathcal{Y}$ from inputs $\mathbf{X} \in \mathcal{X} = \bigotimes_{j=1}^{d} \mathcal{X}_j$, where the pair $(\mathbf{X}, Y)$ is considered as random, drawn from a distribution P. Formally, the goal is to find a function $f : \mathcal{X} \to \mathcal{Y}$, that minimizes $\mathbb{E}[\ell(f(\mathbf{X}), Y)]$ given a cost function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$, called the *loss* [Vapnik, 1999]. The best possible prediction function is known as the *Bayes predictor*, given by

$$f^\star \in \operatorname*{argmin}_{f\colon \mathcal{X} \to \mathcal{Y}} \mathbb{E}\left[\ell(f(\mathbf{X}), Y)\right], \tag{1}$$

and its expected loss is the *Bayes loss* [Devroye et al., 2013]. A *learning* procedure is used to create a function f based on a set of *training* pairs $\mathcal{D}_{n,\text{train}} = \{(\mathbf{X}_i, Y_i), i = 1, \ldots, n\}$. The function f is therefore itself a function of $\mathcal{D}_{n,\text{train}}$, and can be written $\hat{f}_{\mathcal{D}_{n,\text{train}}}$ or simply $\hat{f}_n$. There are many different learning procedures, including random forests [Breiman, 2001] or support vector machines [SVM, see Cortes and Vapnik, 1995]. A learning procedure that, given an infinite amount of data, yields a function that achieves the Bayes loss is said to be *Bayes consistent*. In other words, $\hat{f}_n$ is Bayes consistent if

$$\lim_{n\to\infty} \mathbb{E}[\ell(\hat{f}_n(\mathbf{X}), Y)] = \mathbb{E}[\ell(f^\star(\mathbf{X}), Y)].$$

A learning procedure that is Bayes consistent for every distribution $(\mathbf{X}, Y)$ is said to be (Bayes) universally consistent. In a classification setting, Y is drawn from a finite set of discrete values, and the cost ℓ is typically the zero-one loss: $\ell(Y_1, Y_2) = \mathbb{1}_{Y_1 \neq Y_2}$. In a regression setting, Y is drawn from continuous values in $\mathbb{R}$ and is assumed to satisfy $\mathbb{E}[Y^2] < \infty$. A common cost is then the square loss, $\ell(Y_1, Y_2) = (Y_1 - Y_2)^2$. Considering the zero-one loss [Rosasco et al., 2004] or the square loss (see *e.g.* sec 1.5.5 of Bishop [2006]), the Bayes predictor $f^\star$, that minimizes the expected loss, satisfies $f^\star(\mathbf{X}) = \mathbb{E}[Y|\mathbf{X}]$.

Note that the learning procedure has access to a finite sample $\mathcal{D}_{n,\text{train}}$, and not to the distribution P, hence it can only use the *empirical* risk, $\sum_{i=1\dots n} \ell(f(\mathbf{X}_i), Y)$, rather than the expected risk. A typical learning procedure is therefore the *empirical risk minimization* defined as the following optimization problem

$$\hat{f}_n \in \underset{f:\mathcal{X}\to\mathcal{Y}}{\operatorname{argmin}} \left(\frac{1}{n} \sum_{i=1}^{n} \ell\left(f(\mathbf{X}_i), Y_i\right) \right).$$

A new data set $\mathcal{D}_{n,\text{test}}$ is then needed to estimate the generalization error of the resulting function $\hat{f}_n$.

2.2 Background on missing values

In this section, we introduce the classic work on missing values, including the different missing-value mechanisms. We then summarize the main methods to handle missing values: imputation and likelihood-based methods. Most of this prior art to deal with missing values is based on a single data set with no distinction between training and test set. One challenge in formalizing statistical learning with missing values is adapting notations to describe precisely incomplete feature vectors: notations not explicit enough have led to confusions [Seaman et al., 2013]; the literature is in flux [Little and Rubin, 2019, preface].

Notations for missing values In presence of missing values, we do not observe a complete vector $\mathbf{X}$. To define precisely the observed quantity, we introduce the

missing indicator vector $\mathbf{M} \in \{0,1\}^d$ which satisfies, for all $1 \leq j \leq d$, $M_j = 1$ if and only if X_j is not observed. The random vector $\mathbf{M}$ acts as a mask on $\mathbf{X}$. To formalize incomplete observations, we use the incomplete feature vector $\widetilde{\mathbf{X}}$ (see Rubin [1976], Rosenbaum and Rubin 1984, appendix B; Mohan and Pearl 2018, Yoon et al. 2018) defined as $\widetilde{X}_j = \texttt{NA}$ if $M_j = 1$, and $\widetilde{X}_j = X_j$ otherwise. As $\mathcal{X}$ is a cartesian product, $\widetilde{\mathbf{X}}$ belongs to the space $\widetilde{\mathcal{X}} = \bigotimes_{j=1}^{d} (\mathcal{X}_j \cup \{\texttt{NA}\})$. We have

$$\widetilde{\mathbf{X}} = \mathbf{X} \odot (\mathbf{1} - \mathbf{M}) + \texttt{NA} \odot \mathbf{M},$$

where $\odot$ is the term-by-term product, with the convention that, for all one-dimensional $x \neq 0$, $\texttt{NA} \cdot x = \texttt{NA}$ and $\texttt{NA} \cdot 0 = 0$. As such, when the data are real, $\widetilde{\mathbf{X}}$ can be seen as a mixed categorical and continuous variable, taking values in $\mathbb{R} \cup \{\texttt{NA}\}$. Here is an example of realizations (lower-case letters) of previous random variables: for a given vector $\mathbf{x} = (1.1, 2.3, -3.1, 8, 5.27)$ with the missing pattern $\mathbf{m} = (0, 1, 0, 0, 1)$, we have

$$\widetilde{\mathbf{x}} = (1.1, \quad \texttt{NA}, \quad -3.1, \quad 8, \quad \texttt{NA}).$$

To write likelihoods (see Section 2.2.1), we must also introduce notations $\mathbf{x}_{obs(m)}$ and $\mathbf{x}_{mis(m)}$, classic in the missing value literature. For any vector $\mathbf{x}$, and any set $J \subset \{1, \ldots, d\}$, we let $\mathbf{x}_J$ be the subvector of $\mathbf{x}$ composed of the components of $\mathbf{x}$ indexed by J. We also let $|J|$ be the cardinal of J. For any vector $\mathbf{m} \in \{0,1\}^d$, we let $obs(\mathbf{m}) = \{j \in \{1, \ldots, d\}, \mathbf{m}_j = 0\}$ and $mis(\mathbf{m}) = \{j \in \{1, \ldots, d\}, \mathbf{m}_j = 1\}$. Hence $\mathbf{x}_{obs(\mathbf{m})} \in \bigotimes_{j \in obs(\mathbf{m})} \mathcal{X}_j$ is composed of observed entries in $\mathbf{x}$ and $\mathbf{x}_{mis(\mathbf{m})} \in \bigotimes_{j \in mis(\mathbf{m})} \mathcal{X}_j$ contains the missing components in $\mathbf{x}$. To shorten notations, we will sometimes write $\mathbf{x}_o$ (resp. $\mathbf{x}_m$) instead of $\mathbf{x}_{obs(\mathbf{m})}$ (resp. $\mathbf{x}_{mis(\mathbf{m})}$). To continue the above example, we have

$$\mathbf{x}_{obs(\mathbf{m})} = \mathbf{x}_{(1,3,4)} = (1.1, \ -3.1, \ \ 8), \qquad \mathbf{x}_{mis(\mathbf{m})} = \mathbf{x}_{(2,5)} = (2.3, \ \ 5.27).$$

Given a vector $\mathbf{x} \in \bigotimes_{j=1}^{d} \mathcal{X}_j$ and a missingness pattern $\mathbf{m}$, one can recompose $\mathbf{x}$ based on $\mathbf{x}_{obs(\mathbf{m})}$, $\mathbf{x}_{mis(\mathbf{m})}$ and $\mathbf{m}$ by the ordering operator $\mathbf{x} = o(\mathbf{x}_{obs(\mathbf{m})}, \mathbf{x}_{mis(\mathbf{m})}; \mathbf{m})$. Finally, we use the generic notation $\hat{\mathbf{x}}$ to denote $\widetilde{\mathbf{x}}$ in which missing values have been imputed, based on any imputation procedure to be specified. For example, if we impute the missing values by 0 in the previous example, we have

$$\hat{\mathbf{x}} = (1.1, \quad 0, -3.1, \quad 8, \quad 0).$$

Notations related to missing data are summarized in Table 1.

Notation	**Description**
$\mathbf{x} \in \mathcal{X}$	Complete input vector
$y \in \mathcal{Y}$	Complete output (always observed)
$\mathbf{m} \in \{0,1\}^d$	Missingness indicator vector
$\widetilde{\mathbf{x}} \in \bigotimes_{j=1}^{d}(\mathcal{X}_j \cup \{\texttt{NA}\})$	Observed vector $\mathbf{x}$ where missing entries are written as `NA`
$\mathbf{x}_{obs(\mathbf{m})} \in \bigotimes_{j \in obs(\mathbf{m})} \mathcal{X}_j$	Subvector of $\mathbf{x}$ containing its observed components
$\mathbf{x}_{mis(\mathbf{m})} \in \bigotimes_{j \in mis(\mathbf{m})} \mathcal{X}_j$	Subvector of $\mathbf{x}$ containing its missing components
$\mathbf{x} = o(\mathbf{x}_{obs(\mathbf{m})}, \mathbf{x}_{mis(\mathbf{m})}; \mathbf{m})$	Complete vector $\mathbf{x}$ recreated by merging $\mathbf{x}_{obs(\mathbf{m})}$ and $\mathbf{x}_{mis(\mathbf{m})}$ according to the missing pattern $\mathbf{m}$
$\hat{\mathbf{x}} \in \mathcal{X}$	Vector $\mathbf{x}$ for which missing entries have been imputed

Table 1: Notations and definitions used throughout the paper. Bold variables are vectors

2.2.1 Missing data mechanisms

To follow the historical definitions which do not give to the response Y a particular role, we temporarily consider Y as part of the input vector $\mathbf{X}$, though we assume that Y has no missing values. Rubin [1976] defines three missing data mechanisms and fundamental results for working with likelihood models in the presence of missing data. Let us consider that realizations $(\mathbf{x}_i, \mathbf{m}_i)$ are sampled i.i.d. from a distribution in $\mathscr{P} = \{g_\theta(\mathbf{x}) g_\phi(\mathbf{m}|\mathbf{x}) : (\theta, \phi) \in \Omega_{\theta,\phi}\}$ where $\Omega_{\theta,\phi} \subset \Theta \times \Phi$ is the joint parameter space (marginally, $\theta \in \Theta$ and $\phi \in \Phi$). The goal in statistical inference

is to estimate the parameter θ. This is usually done by maximizing the likelihood $\mathcal{L}(\theta) = \prod_{i=1}^{n} g_\theta(\mathbf{x}_i)$, which is well defined when the $\mathbf{x}_i$ are fully observed. Recall that each $\mathbf{x}_i$ can be decomposed into an observed vector $\mathbf{x}_{i,o}$ and an unobserved vector $\mathbf{x}_{i,m}$. Here, the likelihood is integrated over the missing values, resulting in

$$\text{(full likelihood)} \quad \mathcal{L}_1(\theta, \phi) = \prod_{i=1}^{n} \int g_\theta(o(\mathbf{x}_{i,o}, \mathbf{x}_{i,m}; \mathbf{m})) g_\phi(\mathbf{m}_i | o(\mathbf{x}_{i,o}, \mathbf{x}_{i,m}; \mathbf{m})) \, \mathrm{d}\mathbf{x}_{i,m},$$

where the integration is taken on the components of $\mathbf{x}_{i,m}$ only. The parameter ϕ is generally not considered as of interest. In addition, modelling the missing values mechanism may require strong parametric assumptions. An easier quantity would be

$$\text{(likelihood of observed data)} \qquad \mathcal{L}_2(\theta) = \prod_{i=1}^{n} \int g_\theta(o(\mathbf{x}_{i,o}, \mathbf{x}_{i,m}; \mathbf{m})) \, \mathrm{d}\mathbf{x}_{i,m}$$

ignoring the missing data mechanism. To leave the difficult term, *i.e.* the missing values mechanism, out of the expectation, Rubin [1976] introduces an *ad hoc* assumption, called *Missing At Random (MAR)*, which is that for all $\phi \in \Phi$, for all $i \in [\![1, n]\!]$, for all $\mathbf{x}' \in \mathcal{X}$,

$$\mathbf{x}'_{obs(\mathbf{m}_i)} = \mathbf{x}_{i,o}, \;\; \Rightarrow \;\; g_\phi(\mathbf{m}_i | \mathbf{x}') = g_\phi(\mathbf{m}_i | \mathbf{x}_i),$$

for instance, for all $a, b \in \mathbb{R}$, $\quad g_\phi((0,1,0,0)|(1,a,3,10)) = g_\phi((0,1,0,0)|(1,b,3,10))$.

Using this assumption, he states the following result.

Theorem 1 (Theorem 7.1 in Rubin [1976]) *Let ϕ such that for all $1 \leq i \leq n$, $g_\phi(\mathbf{m}_i|\mathbf{x}_i) > 0$. Assume (a) MAR, (b) $\Omega_{\theta,\phi} = \Theta \times \Phi$, then $\mathcal{L}_2(\theta)$ is proportional to $\mathcal{L}_1(\theta, \phi)$ with respect to θ, so that the inference for θ can be obtained by maximizing the likelihood $\mathcal{L}_2$, which ignores the missing mechanism.*

MAR has a stronger version, more intuitive: *Missing Completely At Random (MCAR)*. In its simplest and strongest form, it states that $\mathbf{M} \perp\!\!\!\perp \mathbf{X}$ (the model density is

$g_\theta(\mathbf{x})g_\phi(\mathbf{m})$). At the other end of the spectrum, if it is not possible to ignore the mechanism, the corresponding model is called *Missing Not At Random (MNAR)*. These definitions are often subject to debates [Seaman et al., 2013] but can be understood using the following example: let us consider two variables, income and age with missing values on income. MCAR means that the missing values are independent of any values; MAR is satisfied if missing values on income depend on the values of age (older people are less incline to reveal their income) whereas MNAR holds if rich people are less prone to reveal their income

There is little literature on missing data mechanism for supervised learning or discriminative models. Kapelner and Bleich [2015] formalise the problem by separating the role of the response y, factorising the likelihood as $g_\theta(\mathbf{x})g_\phi(\mathbf{m}|\mathbf{x})g_\chi(y|\mathbf{x},\mathbf{m})$. Note that they do not write $g_\phi(\mathbf{m}|\mathbf{x},y)$. They justify this factorisation with the – somewhat causal – consideration that the missing values are part of the features, which precede the response. The need to represent the response variable in the factorization show that it may be useful to extend the traditional mechanisms for a supervised learning setting: the link between the mechanism and the output variable can have a significant impact on the results. Davidian [2017] and Arel-Bundock and Pelc [2018] noticed that as long as $\mathbf{M}$ does not depend on Y, it is possible to estimate regression coefficients without bias even with listwise deletion and MNAR values. Ding and Simonoff [2010] generalise the MAR assumption with the following nomenclature MXY: the missing mechanism can marginally depend on the target (**Y), on the features that are always observed (*X*) or on the features that can be missing (M**).

2.2.2 Imputation prior to analysis

Most statistical models and machine-learning procedures are not designed for incomplete data. To use existing pipelines in the presence on missing values, *imputing* the data is commonly used to form a completed data set. To (single) impute data, *joint modeling* (JM) approaches capture the joint distribution across features [Lit-

tle and Rubin, 2019]. A simple example of joint modeling imputation is to assume a Gaussian distribution of the data and to estimate the mean vector and covariance matrix from the incomplete data (using an EM algorithm, see Section 2.2.3). Missing entries can then be imputed with their conditional expectation knowing the observed data and the estimated parameters. More powerful methods can be based on low-rank models [Hastie et al., 2015, Josse et al., 2016]. While KNN imputation is not intrinsically suited for high-dimensional data, such methods often yield good predictive performances [see, e.g., Batista and Monard, 2003, Poulos and Valle, 2018]. Recently, several deep learning (DL) architectures have been proposed to impute datasets, based on variational autoencoder [Mattei and Frellsen, 2019], GAN such as GAIN [Yoon et al., 2018] or MisGAN [Li et al., 2019], or on denoising autoencoders as MIDA [Vincent et al., 2008, Gondara and Wang, 2018] just to name a few. The quality of an imputation strategy is usually assessed via its RMSE performance: several observed values are first hidden, then the imputed values are compared to the true ones in terms of RMSE. While such a protocol is easy to implement, it has been recently criticized [see, e.g., Näf et al., 2023] as the optimal imputation is then the conditional mean, therefore reducing the variability of the distribution of imputed data, compared to the actual true distribution. While DL architectures show promising performances for imputation in terms of RMSE, DL architectures fails to provide a correct imputation in terms of distributional statistics [mean, variance, correlation, see Wang et al., 2022]. A new line of research aims at designing imputation scores able to reflect how the distribution of imputed value is close to the actual distribution [see, e.g., Näf et al., 2023]. In this paper, we are interested in the predictive performance of any methods (including imputation followed by a learning algorithm) capable of handling missing data. Therefore, the quality of imputation is not assessed on its own but through the predictive performance of subsequent learning algorithm applied on the imputed data set.

Another class of popular approaches to impute data defines the joint distribution implicitly by the conditional distributions of each variable. These approaches are called fully conditional specification (FCS) or imputation with conditional equation (ICE) [van Buuren, 2018]. This formulation is very flexible and can easily handle variables of a different nature such as ordinal, categorical, numerical, etc, via a separate model for each, *e.g.* using supervised learning. Well-known examples of such approach are `missForest`, using iterative imputation of each variable by random forests [Stekhoven and Bühlmann, 2011], or `MICE` [Buuren and Groothuis-Oudshoorn, 2010]. Their computational scalability however prohibits their use on large dataset. As they fit one model per feature, their cost is at least $O(d^2)$: using random forests as a base model leads to a complexity $O(d^2 n \log n)$ and using a ridge model amounts to a complexity $O(d^2 n \min(n, d))$.

The role of the dependent variable Y and whether or not to include it in the imputation model has been a rather controversial point. Indeed, it is quite counter-intuitive to include it when the aim is to apply a conditional model on the imputed data set to predict the outcome Y. Nevertheless, it is recommended as it can provide information for imputing covariates [Allison, 2001, p.57]. Sterne et al. [2009] illustrated the point for the simple case of a bivariate Gaussian data (X, Y) with a positive structure of correlation and missing values on X. Imputing using only X is not appropriate when the aim is to estimate the parameters of the linear regression model of Y given X.

One important issue with "single" imputation, *i.e.* predicting only one value for each missing entries, is that it forgets that some values were missing and considers imputed values and observed values in the same way. It leads to underestimation of the variance of the parameters [Little and Rubin, 2019] estimated on the completed data. One solution, to incorporate the uncertainty of the imputed values is to use multiple imputation (MI, Rubin 1987) where many plausible values are generated for each missing entries, leading to many imputed data sets. Then, MI consists in applying an analysis on each imputed data sets and combining the results.

Although many procedures to generate multiple imputed data sets are available [Murray, 2018], here again, the case of discriminative models is rarely considered, with the exception of Wood et al. [2008] who use a variable selection procedure on each imputed data set and propose to keep the variables selected in all imputed data sets to construct the final model [see also Liu et al., 2016].

2.2.3 EM algorithm

Imputation leads to two-step methods that are generic in the sense that any analysis can be performed from the same imputed data set. On the contrary, the expectation maximization (EM) algorithm [Dempster et al., 1977] proceeds directly in one step. It can thus be better suited to a specific problem but requires the development of a dedicated algorithm.

The EM algorithm can be used in missing data settings to compute maximum likelihood estimates from an incomplete data set. Indeed, with the assumptions of Theorem 1 (MAR settings), maximizing the observed likelihood $\mathcal{L}_2$ gives principle estimation of parameters θ. The log-likelihood of the observed data is

$$\log \mathcal{L}_2(\theta) = \sum_{i=1}^{n} \log \int g_\theta(o(\mathbf{x}_{i,o}, \mathbf{x}_{i,m}; \mathbf{m}))\, \mathrm{d}\mathbf{x}_{i,m}.$$

Starting from an initial parameter $\theta^{(0)}$, the algorithm alternates the two following steps,

$$\textbf{(E-step)} \quad Q(\theta|\theta^{(t)}) = \sum_{i=1}^{n} \int (\log g_\theta(o(\mathbf{x}_{i,o}, \mathbf{x}_{i,m}; \mathbf{m}))) g_{\theta^{(t)}}(o(\mathbf{x}_{i,o}, \mathbf{x}_{i,m}; \mathbf{m}))\, \mathrm{d}\mathbf{x}_{i,m}.$$

$$\textbf{(M-step)} \qquad \theta^{(t+1)} \in \operatorname*{argmax}_{\theta \in \Theta} Q(\theta|\theta^{(t)}).$$

The well-known property of the EM algorithm states that at each step t, the observed log-likelihood increases, although there is no guarantee to find the global maximum. In Appendix C.2 we give an example of an EM algorithm to estimate the parameters of a bivariate Gaussian distribution from incomplete data. The in-

terested reader can refer to Roche [2011] and the references therein for a theoretical review on the EM algorithm.

3 Supervised learning procedures with missing data on train and test set

Supervised learning typically assumes that the data are i.i.d. In particular, an out-of-sample observation (test set) is supposed to be drawn from the same distribution as the original sample (train set). Hence, it must possess the same missing-values mechanism. An appropriate method should then be able to predict on new data with missing values. Here we discuss how to adapt classic missing-values techniques to machine-learning settings, and vice versa.

3.1 Out-of-sample imputation

Using missing-value imputation in a supervised learning setting is not straightforward as it requires to impute new, out-of-sample, test data, where the target Y is unavailable.

A simple strategy is to fit an imputation model on the training set; let us consider a parametric imputation model governed by a parameter α . Based on the training set, we estimate a value $\hat{\alpha}$ and use this value to impute the training set, which is then denoted by $\hat{\mathbf{X}}_{\text{train}}$. Then a supervised-learning model is learned using $\hat{\mathbf{X}}_{\text{train}}$ and Y_{train}. If the supervised-learning procedure is indexed by a parameter β, it yields the estimated parameter $\hat{\beta}$. Finally, on the test set, the covariates must be imputed with the same imputation model (using $\hat{\alpha}$) and the prediction is built using the imputed test set and the estimated learning model (using $\hat{\beta}$).

This approach is easy to implement for *univariate imputation* methods that consider each feature separately, for instance with mean imputation: parameters $\hat{\alpha}$ correspond to the mean $\hat{\mu}_j$ of each column which is learned on the training set, and new observations on the test set are imputed by $(\hat{\mu}_1, \ldots, \hat{\mu}_d)$. This approach can also be implemented with a joint Gaussian model on $(\mathbf{X}, Y)$, learning param-

eters of the Gaussian with the EM algorithm on the training set. Indeed, one can then impute the test set using the conditional expectations of the missing features given the observed features (without Y) and the estimated parameters.

For more general imputation methods, two issues hinder out-of-sample imputation. First, many available imputation methods are "black-boxes" that take as input an incomplete data set and output a completed data set: they do not separate the estimation of model parameters from their use to complete the data. This is the case for many implementations of iterative conditional imputation such as missForest [Stekhoven and Bühlmann, 2011], through scikit-learn [Pedregosa et al., 2011] and recent versions of MICE [van Buuren, 2018] (using an argument "ignore") provides out-of-sample iterative conditional imputation. It is also difficult for powerful imputers presented in Section 2.2.2 such as low-rank matrix completion, which cannot be easily marginalised on $\mathbf{X}$ alone.

As most existing implementations cannot easily impute a new data set with the same imputation model, some analysts resort to performing separate imputation of the training set and the test set. But the smaller the test set, the more suboptimal this strategy is, and it completely fails in the case where only one observation has to be predicted. Another option is to consider semi-supervised settings, when the test set is available at train time: grouped imputation can then simultaneously impute the train and the test set [Kapelner and Bleich, 2015], while the predictive model is subsequently learned on the training set only.

3.2 EM and out-of-sample prediction

The likelihood framework (Section 2.2.1) enables predicting new observation, though it has not been much discussed. Jiang et al. [2019] consider a special case of this approach for a logistic regression where covariates $\mathbf{X}$ are assumed to be Gaussian. Let the assumptions of Theorem 1 be verified (MAR settings). The true parameters of the model can then be estimated by maximizing the observed log-likelihood $\log \mathcal{L}_2$ with an EM algorithm on the train data (Section 2.2.3). The corresponding

estimates $\hat{\theta}_n$ can be used for out-of-sample prediction with missing values. The probability distribution of y as a function of the observed values $\mathbf{x}_o = \mathbf{x}_{obs(\mathbf{m})}$ only, can be related to that on a fully-observed data set:

$$\begin{aligned} g_{\hat{\theta}_n}(y|\mathbf{X}_o = \mathbf{x}_o) &= \frac{g_{\hat{\theta}_n}(y, \mathbf{x}_o)}{g_{\hat{\theta}_n}(\mathbf{x}_o)} \\ &= \frac{1}{g_{\hat{\theta}_n}(\mathbf{x}_o)} \int g_{\hat{\theta}_n}(y, o(\mathbf{x}_o, \mathbf{x}_m; \mathbf{m}))\, \mathrm{d}\mathbf{x}_m \\ &= \int g_{\hat{\theta}_n}(y|o(\mathbf{x}_o, \mathbf{x}_m; \mathbf{m})) \frac{g_{\hat{\theta}_n}(o(\mathbf{x}_o, \mathbf{x}_m; \mathbf{m}))}{g_{\hat{\theta}_n}(\mathbf{x}_o)}\, \mathrm{d}\mathbf{x}_m \\ &= \mathbb{E}_{\mathbf{X}_m|\mathbf{X}_o=\mathbf{x}_o}\left[g_{\hat{\theta}_n}(y|o(\mathbf{x}_o, \mathbf{X}_m; \mathbf{m}))\right] \end{aligned} \tag{2}$$

It is then possible to approximate the expectation with Monte Carlo sampling from the distribution $g_{\hat{\theta}_n}(\mathbf{X}_\mathrm{m}|\mathbf{X}_\mathrm{o} = \mathbf{x}_\mathrm{o})$. Such a sampling is easy in simple models, *e.g.* using Schur's complements for Gaussian distributions in linear regression settings. But in more complex settings, such as logistic regression, there is no explicit solution and one option is to use Metropolis-Hasting algorithms [Hastings, 1970].

3.3 Empirical risk minimization with missing data

The two approaches discussed above are specifically designed to fix the missing-values issue: imputing or specifying a parametric model and computing the probability of the response given the observed values. However, in supervised-learning settings, the goal is rather to build a prediction function that minimizes an expected risk. Empirical risk minimization, the workhorse of machine learning, can be adapted to deal with missing data.

Recall that in missing-values settings, we do not have access to $\mathbf{X}$ but rather to the incomplete vector $\widetilde{\mathbf{X}}$. Therefore, we aim at minimizing the empirical risk over the set of measurable functions from $\widetilde{\mathcal{X}}$ to $\mathcal{Y}$, that is

$$\widehat{f}_n \in \underset{f:\widetilde{\mathcal{X}}\to\mathcal{Y}}{\operatorname{argmin}} \frac{1}{n}\sum_{i=1}^{n} \ell\left(f(\widetilde{\mathbf{X}}_i), Y_i\right). \tag{3}$$

Unfortunately, the half-discrete nature of $\widetilde{\mathcal{X}} = \bigotimes_{j=1}^{d} (\mathcal{X}_j \cup \{\texttt{NA}\})$ makes the problem difficult. Indeed, many learning algorithms do not work with mixed data types, such as $\mathbb{R} \cup \{\texttt{NA}\}$, but rather require a vector space. This is true in particular for gradient-based algorithms. As a result, the optimization problem (3) is hard to solve with typical learning tools.

Another point of view can be adopted for losses which leads to Bayes-optimal solutions defined as $f^\star_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}) = \mathbb{E}[Y|\widetilde{\mathbf{X}}]$. As there are at most 2^d admissible missing patterns, we can rewrite the Bayes estimate as

$$f^\star_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}) = \sum_{\mathbf{m} \in \{0,1\}^d} \mathbb{E}\left[Y \middle| \mathbf{X}_{obs(\mathbf{m})}, \mathbf{M} = \mathbf{m}\right] \mathbb{1}_{\mathbf{M}=\mathbf{m}}, \tag{4}$$

This formulation highlights the combinatorial issues: solving (3) may require, as suggested by Rosenbaum and Rubin [1984, Appendix B], to estimate 2^d different submodels, that is $\mathbb{E}\left[Y \middle| \mathbf{X}_{obs(\mathbf{m})}, \mathbf{M} = \mathbf{m}\right]$ appearing in (4) for each $\mathbf{m} \in \{0,1\}^d$, which grows exponentially with the number of variables.

Modifying existing algorithms or creating new ones to deal with the optimization problem (3) is in general a difficult task, due to the numerous possible missing data patterns. Nevertheless, we will see in Section 5 and Appendix B that decision trees are particularly well suited to address this problem.

Remark 1 Note that, in practice, not all patterns may be possible in the training and test sets. For instance, if there are only complete data in the train set, the only submodel of interest is $\mathbb{E}\left[Y \middle| \mathbf{X}_{obs(\mathbf{m})}, \mathbf{M} = \mathbf{m}\right]$ for $\mathbf{m} = (0, \ldots, 0)$, which boils down to the regular supervised-learning scenario on a complete data set. However, the train and test sets are assumed to be drawn from the same data distribution. Hence, we expect to observe similar patterns of missingness in train and test sets. If this is not the case, we are in presence of a distributional shift, which should be tackled with dedicated methods [see, e.g., Sugiyama et al., 2017]. This may happen for instance, when a study conducted on past data leads to operational

recommendations, advising practitioners to focus on certain variables of interest. In that case, they will more likely measure them systematically.

4 Consistency of imputation procedures

In this section, we show theoretically that, without assuming any parametric distribution for the data, imputation procedures can lead to a Bayes-optimal predictor in the presence of missing data on covariates (in both train and test sets), *i.e.* it asymptotically targets the function $f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}) = \mathbb{E}[Y|\widetilde{\mathbf{X}}]$.

In Section 4.1, we assume that we are given the true regression function $f^{\star}$ on the fully-observed data and study the performance of applying this regression function to a test set with missing values, using several imputation strategies: unconditional mean, conditional mean and multiple imputation. Note that, for MCAR data, the function $f^{\star}$ can be estimated using the complete observations only, i.e., by deleting observations with missing values in the train set and applying a supervised procedure on the remaining observations. Such a strategy is relevant for very large training sets and MCAR missing values.

In Section 4.2, we consider the full problem of tackling missing values in the train and the test set, which is of particular interest when the training set is of reasonable size as it can leverage the information contained in incomplete observations. We study a classical approach, described in Section 3.1, which consists in imputing the training set, fitting a learning algorithm on the imputed data, and predicting on a test set which has been imputed with the same method. Although mean imputation of variables is one of the most widely used approaches, it is highly criticised in the classic literature for missing data [Little and Rubin, 2019]. Indeed, it leads to a distortion of the data distribution and, consequently, statistics calculated on the imputed data table are biased. A simple example is the correlation coefficient between two variables, which is biased towards zero if the missing data are imputed by the mean. However, in a supervised-learning setting, the aim is not to compute statistics representative of the data set, but to minimize a prediction risk

by estimating a regression function. For this purpose, we show in Section 4.2 that constant imputation may be completely appropriate and leads to consistent estimation of the prediction function. This result is remarkable and extremely useful in practice.

4.1 Test-time imputation

Here we consider that we have access to the optimal (Bayes) predictor $f^\star$ for the complete data, *i.e.* $f^\star(\mathbf{X}) = \mathbb{E}[Y|\mathbf{X}]$, and we show that, in MAR settings, when missing data appears in the test set, the optimal predictor for incomplete data can be computed by using multiple imputation with $f^\star$.

4.1.1 Test-time conditional multiple imputation is consistent

Let us first make explicit the multiple imputation procedure for prediction. Recall that we observe $\mathbf{x}_{obs(\mathbf{m})}$. We then draw the missing values $\mathbf{X}_m$ from the conditional distribution $\mathbf{X}_m|\mathbf{X}_\text{o} = \mathbf{x}_\text{o}$ and compute the regression function on these completed observations. The resulting multiple imputation function is given by:

$$f^\star_{\text{MI}}(\widetilde{\mathbf{x}}) = \mathbb{E}_{\mathbf{X}_m|\mathbf{X}_o=\mathbf{x}_o}[f^\star(o(\mathbf{x}_o, \mathbf{X}_m; \mathbf{m}))]. \tag{5}$$

Note that this expression is similar to the expression (2) given for EM, but assuming that we know the true nonparametric distribution of the data.

Assumption 1 (Regression model) *The regression model satisfies* $Y = f^\star(\mathbf{X}) + \varepsilon$, *where* $\mathbf{X}$ *takes values in* $\mathbb{R}^d$ *and* ε *satisfies a.s.* $\mathbb{E}[\varepsilon|\mathbf{X}_{obs(\mathbf{M})}] = 0$.

Assumption 2 (Missingness pattern - MAR-Y) *We have* $Y \perp\!\!\!\perp \mathbf{M}|X_{obs(\mathbf{M})}$

The missingness pattern in Assumption 2 is more generic than MAR, since it is notably verified if the missingness pattern is MAR, that is if $\mathbb{P}[\mathbf{M} = \mathbf{m}|\mathbf{X}] = \mathbb{P}[\mathbf{M} = \mathbf{m}|\mathbf{X}_{obs(\mathbf{m})}]$ (classic definition recalled in Section 2.2.1), *i.e.* if the probability to observe a given pattern depends only on the observed values.

Theorem 2 *Grant Assumption 1 and 2. Then the multiple imputation procedure, defined in (5) is Bayes optimal, that is, for all* $\widetilde{\mathbf{x}} \in (\mathbb{R} \cup \{NA\})^d$,

$$f^{\star}_{MI}(\widetilde{\mathbf{x}}) = f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{x}}).$$

The proof is given in Appendix A. Theorem 2 justifies the use of multiple imputation when missing pattern is MAR and when we have access to an estimate of $f^{\star}$ and of the conditional distribution $\mathbf{X}_m|\mathbf{X}_o = \mathbf{x}_o$.

4.1.2 Single mean imputation is not consistent

Given the success of multiple imputation, it is worth checking that single imputation is not sufficient. We show with two simple examples that indeed, single imputation on the test set is not consistent even in MAR setting.

We first show, that (unconditional) mean imputation is not consistent, if the learning algorithm has been trained on the complete cases only.

Example 1 In one dimension, consider the following simple example,

$$X_1 \sim U(0,1), \quad Y = X_1^2 + \varepsilon, \quad M_1 \sim \mathcal{B}(1/2) \perp\!\!\!\perp (X_1, Y),$$

with ε an independent centered Gaussian noise. Here, $\mathbb{E}[Y|X_1] = X_1^2$, and the regression function $f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}) = \mathbb{E}[Y|\widetilde{\mathbf{X}}]$ satisfies

$$\begin{aligned} f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}) &= X_1^2 \cdot \mathbb{1}_{M_1=0} + \mathbb{E}[Y|\widetilde{X} = \texttt{NA}] \cdot \mathbb{1}_{M_1=1} \\ &= X_1^2 \cdot \mathbb{1}_{M_1=0} + \mathbb{E}[X_1^2] \cdot \mathbb{1}_{M_1=1} \\ &= X_1^2 \cdot \mathbb{1}_{M_1=0} + (1/3) \cdot \mathbb{1}_{M_1=1}. \end{aligned} \tag{6}$$

In the oracle setting where the distribution of (X_1, Y, M_1) is known, "plugging in" the mean imputation of X_1 yields the Plug-in Imputation function prediction

$$\begin{aligned} f_{PI}(\widetilde{\mathbf{X}}) &= X_1^2 \cdot \mathbb{1}_{M_1=0} + (\mathbb{E}[X_1])^2 \cdot \mathbb{1}_{M_1=1} \\ &= X_1^2 \cdot \mathbb{1}_{M_1=0} + (1/4) \cdot \mathbb{1}_{M_1=1}. \end{aligned} \tag{7}$$

In this example, mean imputation is not optimal: when X_1 is missing, the prediction obtained by mean imputation is $1/4$, whereas the optimal prediction (the one which minimizes the square loss) is $1/3$ as seen in (6).

Inspecting (6) and (7) reveals that the poor performance of mean imputation is due to the fact that $\mathbb{E}[X_1^2] \neq (\mathbb{E}[X_1])^2$. The non-linear relation between Y and X_1 breaks mean imputation. This highlights the fact that the imputation method should be chosen in accordance with the learning algorithm that will be applied later on. This is related to the concept of congeniality [Meng, 1994] defined in multiple imputation.

4.1.3 Conditional mean imputation is consistent if there are deterministic relations between input variables

We now consider conditional mean imputation, using information of other observed variables to impute. Conditional mean imputation may work in situations where there is redundancy between variables, as highlighted in Example 2. However, we give a simple example below stressing that using it to impute the test may not be Bayes optimal.

Example 2 Consider the following regression problem with two identical input variables:

$$X_1 = X_2 \sim \mathcal{U}([0,1]), \quad Y = X_1 + X_2^2 + \varepsilon, \quad M_2 \sim \mathcal{B}(1/2) \perp\!\!\!\perp (X_1, X_2, Y)$$

The Bayes predictor is then given by

$$f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}) = \begin{cases} X_1 + X_2^2 & \text{if } \widetilde{X}_2 \neq \texttt{NA} \\ X_1 + \mathbb{E}[X_2^2 | X_1, \widetilde{X}_2 = \texttt{NA}] & \text{if } \widetilde{X}_2 = \texttt{NA} \end{cases}$$
$$= \begin{cases} X_1 + X_2^2 \text{ if } \widetilde{X}_2 \neq \texttt{NA} \\ X_1 + X_1^2 \text{ if } \widetilde{X}_2 = \texttt{NA} \end{cases}$$

Plugging in the previous expression the mean of X_2 when missing leads to the Plug-in Imputation function

$$f_{PI}(\widetilde{\mathbf{X}}) = \begin{cases} X_1 + X_2^2 & \text{if } \widetilde{X}_2 \neq \texttt{NA} \\ X_1 + (1/4) & \text{if } \widetilde{X}_2 = \texttt{NA}, \end{cases}$$

whereas plugging in the Mean Imputation of X_2 conditional on X_1 leads to

$$f_{PMI}(\widetilde{\mathbf{X}}) = \begin{cases} X_1 + X_2^2 \text{ if } \widetilde{X}_2 \neq \texttt{NA} \\ X_1 + X_1^2 \text{ if } \widetilde{X}_2 = \texttt{NA} \end{cases},$$

as $(\mathbb{E}[X_2|X_1])^2 = X_1^2$.

If there is no deterministic link between variables, conditional mean imputation fails to recover the regression function, in the case where the regression function is not linear (see Example 2, where $X_1 = X_2$ is replaced by $X_1 = X_2 + \varepsilon$).

4.1.4 Pathological case: missingness is a covariate

Example 3 below shows a situation in which any imputation method, single or multiple, fails, because the missingness contains information about the response variable Y. In this univariate setting, there is no distinction between conditional and unconditional mean.

Example 3 Consider the following regression model,

$$X_1 \sim \mathcal{U}(0,1) \qquad M_1 \sim \mathcal{B}(1/2) \perp\!\!\!\perp X_1 \qquad Y = X_1 \cdot \mathbb{1}_{M_1=0} + 3X_1 \cdot \mathbb{1}_{M_1=1} + \varepsilon.$$

Here, $\mathbb{E}[Y|X_1] = X_1 \cdot \mathbb{P}(M_1 = 0) + 3X_1 \cdot \mathbb{P}(M_1 = 1) = 2X_1\,.$

Plugging in the mean imputation of X_1 leads to

$$\begin{aligned} f_{PI}(\widetilde{X}) &= X_1 \cdot \mathbb{1}_{M_1=0} + \mathbb{E}[X_1] \cdot \mathbb{1}_{M_1=1} \\ &= X_1 \cdot \mathbb{1}_{M_1=0} + (1/2) \cdot \mathbb{1}_{M_1=1}, \end{aligned}$$

whereas the regression function satisfies

$$\begin{aligned} f^\star_{\widetilde{\mathbf{X}}}(\widetilde{X}) &= X_1 \cdot \mathbb{1}_{M_1=0} + 3\mathbb{E}[X_1|\widetilde{X} = \texttt{NA}] \cdot \mathbb{1}_{M_1=1} \\ &= X_1 \cdot \mathbb{1}_{M_1=0} + (3/2) \cdot \mathbb{1}_{M_1=1}. \end{aligned}$$

In this case, the presence of missing values is informative in itself, and having access to the complete data set (all values of X_1) does not provide enough information. Such a scenario advocates for considering the missingness as an additional input variable. Indeed, in such situations, single and multiple imputation fail to recover the targeted regression function, without adding a missingness indicator to the input variables.

4.2 Constant imputation at train and test time is consistent

We now show that the same single imputation used in both train and test sets leads to consistent procedures. More precisely, we allow missing data on X_1 only, and replace its value by some constant $\alpha \in \mathbb{R}$ if X_1 is missing. More precisely, for each observed $\tilde{\mathbf{x}} \in (\mathbb{R} \cup \{\texttt{NA}\}) \times \mathbb{R}^{d-1}$, the imputed entry is defined as $\mathbf{x}' = (x_1', x_2, \ldots, x_d)$ where

(constant imputation) $$x_1' = x_1 \mathbb{1}_{M_1=0} + \alpha \mathbb{1}_{M_1=1}.$$

We consider the following procedure: (i) impute the missing values on X_1 in the training set by α (ii) use a universally consistent algorithm (see the definition

in Section 2) on the training set to approach the regression function $f^{\star}_{\mathrm{SI}}(\mathbf{x}') = \mathbb{E}[Y|\mathbf{X}' = \mathbf{x}']$. Theorem 3 shows that this procedure is consistent under the following assumptions.

Assumption 3 (Regression model) *Let $Y = f^{\star}(\mathbf{X}) + \varepsilon$ where $\mathbf{X}$ has a continuous density $g > 0$ on $[0,1]^d$, $f^{\star}$ is continuous, and ε is a centred noise independent of $(\mathbf{X}, M_1)$.*

Assumption 4 (Missingness pattern - MAR) *The variables $X_2, \ldots, X_d$ are fully observed and the missingness pattern M_1 on variable X_1 satisfies $M_1 \perp\!\!\!\perp X_1 | X_2, \ldots, X_d$ and is such that the function $(x_2, \ldots, x_d) \mapsto \mathbb{P}[M_1 = 1 | X_2 = x_2, \ldots, X_d = x_d]$ is continuous.*

As for Assumption 2, Assumption 4 states that the missingness pattern is a specific MAR process since only X_1 can be missing with a probability that depends only on the other variables, which are always observed. The conditional distribution of M is also assumed to be continuous to avoid technical complexities in the proof of Theorem 3.

Theorem 3 *Grant Assumption 3 and 4. The single imputation procedure described above satisfies, for all imputed entries $\mathbf{x}' \in \mathbb{R}^d$,*

$$\begin{aligned} f^{\star}_{SI}(\mathbf{x}') = {} & \mathbb{E}[Y | X_2 = x_2, \ldots, X_d = x_d, M_1 = 1] \mathbb{1}_{x'_1 = \alpha} \mathbb{1}_{\mathbb{P}[M_1 = 1 | X_2 = x_2, \ldots, X_d = x_d] > 0} \\ & + \mathbb{E}[Y | \mathbf{X} = \mathbf{x}'] \mathbb{1}_{x'_1 = \alpha} \mathbb{1}_{\mathbb{P}[M_1 = 1 | X_2 = x_2, \ldots, X_d = x_d] = 0} \\ & + \mathbb{E}[Y | \mathbf{X} = \mathbf{x}', M_1 = 0] \mathbb{1}_{x'_1 \neq \alpha}. \end{aligned}$$

Consequently, letting

$$\widetilde{\mathbf{X}} = \begin{cases} \mathbf{X}' & \text{if } X'_1 \neq \alpha \\ (\texttt{NA}, X_2, \ldots, X_d) & \text{if } X'_1 = \alpha \end{cases},$$

the single imputation procedure is equal to the Bayes function almost everywhere, that is

$$f^{\star}_{SI}(\mathbf{X}') = f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}}). \tag{8}$$

The proof is given in Appendix A. Theorem 3 confirms that it is preferable to use the same imputation for the training and test sets. Indeed, the learning algorithm can learn the imputed value and use that information to detect that the entry was initially missing. If the imputed value changes from train set to test set (for example, if instead of imputing the test set with the mean of the variables of the train set, one imputes by the mean of the variables on the test set), the learning algorithm may fail, since the imputed data distribution would differ between train and test sets.

Multivariate missingness. Interestingly, Theorem 3 remains valid when missing values occur for variables $X_1, \ldots, X_j$ under the assumption that $(M_1, \ldots, M_j) \perp\!\!\!\perp (X_1, \ldots, X_j)$ conditional on $(X_{j+1}, \ldots, X_d)$ and if for every pattern $\mathbf{m} \in \{0,1\}^j \times \{0\}^{d-j}$, the functions $(x_{j+1}, \ldots, x_d) \mapsto \mathbb{P}[\mathbf{M} = \mathbf{m} | X_{j+1} = x_{j+1}, \ldots, X_d = x_d]$ are continuous.

Note that the precise imputed value α does not matter if the learning algorithm is universally consistent. By default, the mean is not a bad choice, as it preserves the first order statistic (mean) of the sample. However, our analysis does not stress out any particular role of mean imputation. In fact, any imputation in the support of X_1 is equivalent and leads to a consistent procedure. The comment below stresses out the benefit of choosing α outside of the support of X_1.

Almost everywhere consistency. The equality between the constant imputation learner $f^\star_{\mathrm{SI}}$ and the Bayes function $f^\star_{\tilde{\mathbf{X}}}$ holds almost surely but not for every $\tilde{\mathbf{x}}$. Indeed, under the setting of Theorem 3, let $\tilde{\mathbf{x}} = (\alpha, x_2, \ldots, x_d)$, for any $x_2, \ldots, x_d \in [0,1]$ such that $\mathbb{P}[M_1 = 1 | X_2 = x_2, \ldots, X_d = x_d] > 0$. In this case, $\mathbf{x}' = (\alpha, x_2, \ldots, x_d)$ and

$$f^\star_{\mathrm{SI}}(\mathbf{x}') = \mathbb{E}[Y | X_2 = x_2, \ldots, X_d = x_d, M_1 = 1],$$

which is different, in general, from

$$f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{x}}) = \mathbb{E}[Y|X_1 = \alpha, X_2 = x_2, \ldots, X_d = x_d].$$

Therefore, on the event $A_1 = \{\tilde{\mathbf{X}}, \tilde{X}_1 = \alpha\}$, the two functions $f^{\star}_{\mathrm{SI}}$ and $f^{\star}_{\widetilde{\mathbf{X}}}$ differ, and thus the equality between these functions does not hold pointwise. However, since A_1 is a zero probability event, the equality $f^{\star}_{\mathrm{SI}}(\mathbf{X}') = f^{\star}_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{X}})$ hold almost everywhere, as stated in Theorem 3. A simple way to obtain the pointwise equality in equation (8) is to impute missing entries by values that are out of the range of the true distribution, which echoes the "separate class" method advocated by Ding and Simonoff [2010].

Discrete/categorical variables. According to Assumption 3, the variables $X_1, \ldots, X_d$ admit a density. Therefore, the proof of Theorem 3 is not valid for discrete variables. However, Theorem 3 can be extended to handle discrete variables, if missing entries in each variable are imputed by an extra category "missing" for each variable. In this framework, consistency boils down to estimating the expectation of Y given a category which directly results from the universal consistency of the selected algorithm.

Classification. Theorem 3 is established for a regression problem. However, in a binary classification setting where $Y \in \{0, 1\}$, the plug-in classifier $\hat{g}_n(\mathbf{x}) = \mathbb{1}_{\hat{\eta}_n(\mathbf{x}) \geq 1/2}$ is universally consistent if the estimate $\hat{\eta}_n$ (an estimator of $\mathbb{E}[Y|\mathbf{X}]$) is universally consistent, according to Corollary 6.2 in Devroye et al. [2013]. Theorem 3 can be applied to consistently estimate $\mathbb{E}[Y|\mathbf{X} = \mathbf{x}]$ via $\hat{\eta}_n(\mathbf{x})$, which leads to a universally consistent plug-in classifier $\hat{g}_n$. Consequently, Theorem 3 is also valid for classification frameworks.

Universal consistency. In Theorem 3, we assume to be given a universally consistent algorithm which may appear as a strong restriction on the choice of the algorithm. However many estimators exhibit this property as, for example, local averaging

estimate [kernel methods, nearest neighbors and decision trees, see Devroye et al., 2013]. The key point of Theorem 3 is to state that the universal consistency of a procedure with missing values results directly from the universal consistency of an algorithm applied to an imputed data set, therefore providing guarantees that consistent algorithm and imputation are useful tools to handle missing values.

Optimizing the imputed values. Bertsimas et al. [2018][1] propose a general framework to optimize the imputation values, and instantiate this framework for different learning algorithm. Extended simulations show that their proposal is competitive in terms of imputation quality (measured via MAE/RMSE) or predictive performances.

Consistency for some specific distributions. In Theorem 3, we assume to be given a universally consistent algorithm. One can legitimately ask if the result still holds if an algorithm which is consistent only for some specific data distribution was used instead. For example, assume that data are generated via a linear model and a linear estimator is used after missing values have been imputed. One can show that the 2^d submodels are not linear in general and consequently, using a single linear model on imputed data does not yield the Bayes loss [Le Morvan et al., 2020]. In a nutshell, Theorem 3 is not valid for procedures that are consistent for some specific data distributions only. The interested reader can refer to Le Morvan et al. [2020] for further details on missing values in linear generative models.

5 Simulations

In Section 3, we provided theoretical guarantees justifying the common practice of imputing by a constant, in a specific MAR setting and in an asymptotic sample regime. More precisely, Theorem 3 gives us a positive response in principle regarding the usage of constant imputation for prediction purpose but does not establish

[1] Their work were submitted after our contribution.

the predictive performances of such methods in a finite-sample regime. To go beyond the framework of Theorem 3, we now evaluate the predictive performances of learning algorithms trained on imputed data sets (following the imputation prior-to-learning strategy of Theorem 3) and, to give a broader perspective, we compare it to that of procedures intrinsically able to handle missing values. In general such procedures are hard to design, but decision trees offer natural approaches for empirical risk minimization with missing values [Saar-Tsechansky and Provost, 2007, Twala et al., 2008]. This is due to their ability to handle the half-discrete nature of $\tilde{\mathbf{X}}$, as they rely on greedy decisions rather than smooth optimization.

We first present the different approaches available to handle missing values in tree-based methods in Section 5.1. We then compare numerically the two different strategies: imputation prior to learning and learning directly with missing data.

5.1 Dealing with missing values using decision trees

There exist several approach to build decision trees based on data sets containing missing data. These approaches can be divided into two parts: (i) those for which the splitting mechanism omits missing data (namely Probabilistic splits, Block propagation and surrogate splits) and (ii) those for which missing data are taken into account to build the splits (Missing incorporated in attributes, MIA). These approaches are detailed below.

For the first set of approaches, in each cell, a splitting criterion (see equation 21 in Appendix B) is computed for each variable $j \in \{1, \ldots, d\}$, based on observations that have non-missing entries X_j. The best split $(j^\star, z^\star)$ (corresponding to a split at position $z^\star$ along variable $j^\star$), is then chosen as the one optimizing this criterion (see Algorithm 1 in Appendix B.2 for details). Since such an approach omits the missing data for building the best split, one need to specify a method to send partially-observed data down the tree. The following approaches propose different strategies.

Surrogate splits Surrogate splits search for a split on another variable that induces a data partition close to the original one. More precisely, for a selected split $(j_0^\star, z_0^\star)$, to send down the tree observations with no $j_0^\star$th variable, a new stump, *i.e.*, a tree with one cut, is fitted to the response $\mathbb{1}_{X_{j_0^\star} \leq z_0^\star}$, using variables $(X_j)_{j \neq j_0^\star}$. The split $(j_1^\star, z_1^\star)$ which minimizes the misclassification error is selected, and observations are split accordingly. Those that lack both variables $j_0^\star$ and $j_1^\star$ are split with the second best, $j_2^\star$, and so on until the proposed split has a worse misclassification error than the blind rule of sending all remaining missing values to the same daughter, the most populated one. To predict, the training surrogates are kept (see Algorithm 2 in Appendix B.2 for details). This construction is the default method in `rpart` [Therneau et al., 1997]. Surrogate method is expected to be appropriate when there are relationships between covariates.

Probabilistic splits Another option is to propagate missing observations according to a Bernoulli distribution $\mathcal{B}(\frac{\#L}{\#L+\#R})$, where $\#L$ (resp. $\#R$) is the number of points already on the left (resp. right) (see Algorithm 3 in Appendix B.5). This is the default method in C4.5 [Quinlan, 2014].

Block propagation A third option is to choose the split on the observed values, and then send all incomplete observations as a block, to a side chosen by minimizing the error (see Algorithm 4 in Appendix B.5). This is the method used in LightGBM [Ke et al., 2017].

Missing incorporated in attribute (MIA, Twala et al. 2008) A second class of methods uses missing values to compute the splitting criterion itself. Consequently, the splitting location depends on the missing values, contrary to all methods presented above. Its most common instance is "missing incorporated in attribute" (MIA), which considers the following splits, for all splits (j, z):

– $\{\widetilde{X}_j \leq z \text{ or } \widetilde{X}_j = \texttt{NA}\}$ vs $\{\widetilde{X}_j > z\}$,

– $\{\widetilde{X}_j \leq z\}$ vs $\{\widetilde{X}_j > z \text{ or } \widetilde{X}_j = \texttt{NA}\}$,
– $\{\widetilde{X}_j \neq \texttt{NA}\}$ vs $\{\widetilde{X}_j = \texttt{NA}\}$.

In a nutshell, MIA tries to send all missing values to the left or to the right for each possible split, or to separate observed values from missing ones. For each option, the prediction error is computed and the selected option is the one minimizing the prediction error (see Algorithm 5 in Appendix B.5 for details).

Missing values are treated as a category by MIA, which is thus nothing but a greedy algorithm minimizing the square loss between Y and a function of $\widetilde{\mathbf{X}}$ and consequently targets the quantity (4) which separate $\mathbb{E}[Y|\widetilde{\mathbf{X}}]$ into 2^d terms. However, it is not exhaustive: at each step, the tree can cut for each variable according to missing or non missing and selects this cut when it is relevant, *i.e.* when it minimizes the prediction error. The final leaves can correspond to a cluster of missing values patterns (observations with missing values on the two first variables for instance and any missing patterns for the other variables).

MIA is thought to be a good method to apply when missing pattern is informative, as this procedure allows to cut with respect to missing/non missing and uses missing data to compute the best splits. Note this latter property implies that the MIA approach does not require a different method to propagate missing data down the tree. Notably, MIA is implemented in the R packages `partykit` [Hothorn and Zeileis, 2015] and `grf` [Tibshirani et al., 2020], as well as in XGBoost [Chen and Guestrin, 2016] and for the `HistGradientBoosting` models in scikit-learn [Pedregosa et al., 2011].

In Appendix B.4, we conduct a theoretical analysis on a very simple regression models to highlight differences between the above strategies. In particular, we compare in Proposition 2 the risk of the different splitting strategies (probabilistic split, block propagation, surrogate split, and MIA) and prove that MIA and surrogate splits are the two best strategies, one of which may be better than the other depending on the dependence structure of covariates. Note that block propagation can be seen as a greedy way of successively optimizing the choices in the two first

options in MIA. However, as we show in Proposition 1, these successive choices are sub-optimal.

5.2 Simulation settings

We consider three regression models with covariates $(X_1, \dots, X_d)$ distributed as $\mathcal{N}(\mu, \Sigma)$ with $\mu = \mathbf{1}_d$ and $\Sigma = \rho \mathbf{1}_d \mathbf{1}_d^T + (1-\rho) I_d$, where $\mathbf{1}_d$ is the d-dimensional vector composed of ones and I_d the $d \times d$ identity matrix. By default, ρ is set to 0.5 in our experiments. The first model is quadratic, the second one is linear, and the third one has been used as a benchmark for tree methods by several authors, including Friedman [1991] and Breiman [1996]. We also consider a last regression model where the relationship between covariates are nonlinear. In all four models, ε is a centered Gaussian noise with standard deviation 0.1.

Model 1 (Quadratic) $Y = X_1^2 + X_2^2 + X_3^2 + \varepsilon$

Model 2 (Linear) $Y = X\beta + \varepsilon$ *with* $\beta = (1, 2, -1, 3, -0.5, -1, 0.3, 1.7, 0.4, -0.3)$.

Model 3 (Friedman) $Y = 10 \sin(\pi X_1 X_2) + 20(X_3 - 0.5)^2 + 10 X_4 + 5 X_5 + \varepsilon$

Model 4 (Friedman, Nonlinear) $Y = \sin(\pi X_1 X_2) + 2(X_3 - 0.5)^2 + X_4 + .5 X_5 + \varepsilon$ *where* X *is a hidden uniform variable on* $[-3, 0]$ *and the covariates* $(X_1, \dots, X_d)$ *are distributed as*

$$\begin{cases} X_1 = X^2 + \varepsilon_1 \\ X_2 = \sin(X) + \varepsilon_2 \\ X_3 = \tanh(X)\exp(X)\sin(X) + \varepsilon_3 \\ X_4 = \sin(X-1) + \cos(X-3)^3 + \varepsilon_4 \\ X_5 = (1-X)^3 + \varepsilon_5 \end{cases} \quad \begin{cases} X_6 = \sqrt{\sin(X^2) + 2} + \varepsilon_6 \\ X_7 = X - 3 + \varepsilon_7 \\ X_8 = (1-X)\sin(X)\cosh(X) + \varepsilon_8 \\ X_9 = \frac{1}{\sin(2X) - 2} + \varepsilon_9 \\ X_{10} = X^4 + \varepsilon_{10} \end{cases},$$

where ε_i *are independent centered Gaussian with standard deviation 0.05.*

In our experiments, we generate missing data as follows.

Missing Pattern 1 (MCAR) *For $p \in [0,1]$, the missingness on variable j is generated according to a Bernoulli distribution*

$$\forall i \in [\![1,n]\!], M_{i,j} \sim \mathcal{B}(p).$$

Missing Pattern 2 (Censoring MNAR) *A direct way to select a proportion p of missing values on a variable X_j, that depends on the underlying value, is to crop them above the $1-p$-th quantile*

$$\forall i \in [\![1,n]\!], M_{i,j} = \mathbb{1}_{X_{i,j} > [X_j]_{(1-p)n}}.$$

Missing Pattern 3 (Predictive missingness) *Last, we consider a pattern mixture model, letting M_j be part of the regression function, with $M_j \perp\!\!\!\perp \mathbf{X}$,*

$$\forall i \in [\![1,n]\!], M_{i,j} \sim \mathcal{B}(p), \quad \textit{and} \quad Y = \sum_{j=1}^{3} \left(X_j^2 + 2\, M_j\right) + \varepsilon.$$

MCAR is the most simple instance of missing patterns. In particular, it falls in the framework of Theorem 3, for which constant imputation leads to consistent predictive methods. MNAR is a more complex pattern where the missingness indicator depends on the true value of the variable. For such a pattern, adding the mask can help the prediction, but using the dependencies between the missing covariate and the existing ones can help to build an accurate prediction. The third missing pattern is, in some sense, pathological (see Example 3) as the output depends directly on the missingness indicator. In such extreme settings, good performances can only be obtained if the mask is added as input variable.

We compare the following methods using implementation in R [R Core Team, 2018] and default values for the tuning parameters. We compare tree-based methods (decision trees, random forests, gradient boosting) able to handle directly missing values, with more classic machine learning methods like Support Vector Machines [SVM, see Cortes and Vapnik, 1995] or K nearest neighbours [KNN, see Cover and Hart, 1967]. Unless stated otherwise, we use the following packages: rpart

[Therneau and Atkinson, 2018] for decision trees, ranger [Wright and Ziegler, 2015] for random forests, XGBoost [Chen and Guestrin, 2016] for gradient boosting, e1071 for SVM and caret for KNN. Note that we have used surrogate splits only with decision trees. More precisely, we compare for decision trees the following strategies:

- **MIA**: missing in attributes (see Remark 2 in Appendix B.3 for implementation details)
- **rpart+mask/ rpart**: CART with surrogate splits, with or without the indicator $\mathbf{M}$ in the covariates
- **ctree+mask/ ctree**: conditional trees, implemented in package partykit [Hothorn and Zeileis, 2015] with or without the indicator $\mathbf{M}$ in the covariates
- **impute mean+mask/ impute mean**: missing values are imputed by unconditional mean with or without the indicator $\mathbf{M}$ added in the covariates
- **impute OOR+mask / impute OOR**: missing values are imputed by a constant value, chosen out of range (OOR) from the values of the corresponding covariate in the training set, with or without the indicator $\mathbf{M}$ added in the covariates
- **impute Gaussian+mask /impute Gaussian**: missing values are imputed by conditional expectation when data are assumed to follow a Gaussian multivariate distribution. More precisely, the parameters of the Gaussian distribution are estimated with an EM algorithm (R package norm [Fox, 2013]). Note that for numerical reasons, we shrink the estimated covariance matrix (replacing $\hat{\Sigma}$ by $0.99 \times \hat{\Sigma} + 0.01 \times \mathrm{tr}(\hat{\Sigma})\mathrm{I_d}$) before imputing. The method can also be applied with the indicator $\mathbf{M}$ added in the imputed values.

For KNN and SVM, we compare the last three methods, namely mean imputation, Out-Of-Range (OOR) imputation, Gaussian imputation (as other methods do not apply), with and without adding the mask, whereas for random forests and gradient boosting methods, we also use MIA.

To evaluate the performance of the methods, we repeatedly draw a training set and a testing set of the same size 1000 times. We choose to display the percentage

of explained variance, *i.e.* the R^2 statistic computed on the test set, defined as

$$R^2 = 1 - \frac{\frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} (\hat{y}_i - y_i)^2}{\mathbb{V}[Y]}, \tag{9}$$

The numerator corresponds to the Mean Square Error (MSE) computed on the test set, and the denominator is the true variance of Y, available since we know the generative model. Using the R^2 metric allows us to quickly assess the quality of a regression model: values close to one indicate a very good predictive model. Note that, depending on the regression model and the missing mechanism at hand, some predictive tasks are easier than other, thus explaining the differences in the average predictive performance of all methods. ~~For visual purposes, we display the *relative* explained variance: for each of the 1000 repetitions separately, we center the scores of all the methods by substracting the mean. This is also done separately for trees, forests, boosting, SVM and KNN.~~ The code for these experiments is available online[2].

Experiment 1 In the first experiment, we use Model 1 with $d = 9$ and introduce missing values on X_1, X_2, and X_3 according to the mechanisms described hereafter. Results are depicted in Figure 1.

Experiment 2 In the second experiment, the other three models are used with $d = 10$, with a MCAR mechanism on all variables. Results are shown in Figure 3.

5.3 Results comparing strategies for fixed sample sizes

Figure 1 presents the results for one choice of correlation ($\rho = 0.5$) between covariates and percentage of missing entries (20%), as others give similar interpretation (see Appendix D for different values of the correlation ρ and the missing rate).
In the MCAR case, all decision-tree methods perform similarly aside from out-of-range imputation which significantly under-performs. Performing a "good" condi-

[2] https://github.com/jacobmchen/supervised_missing/

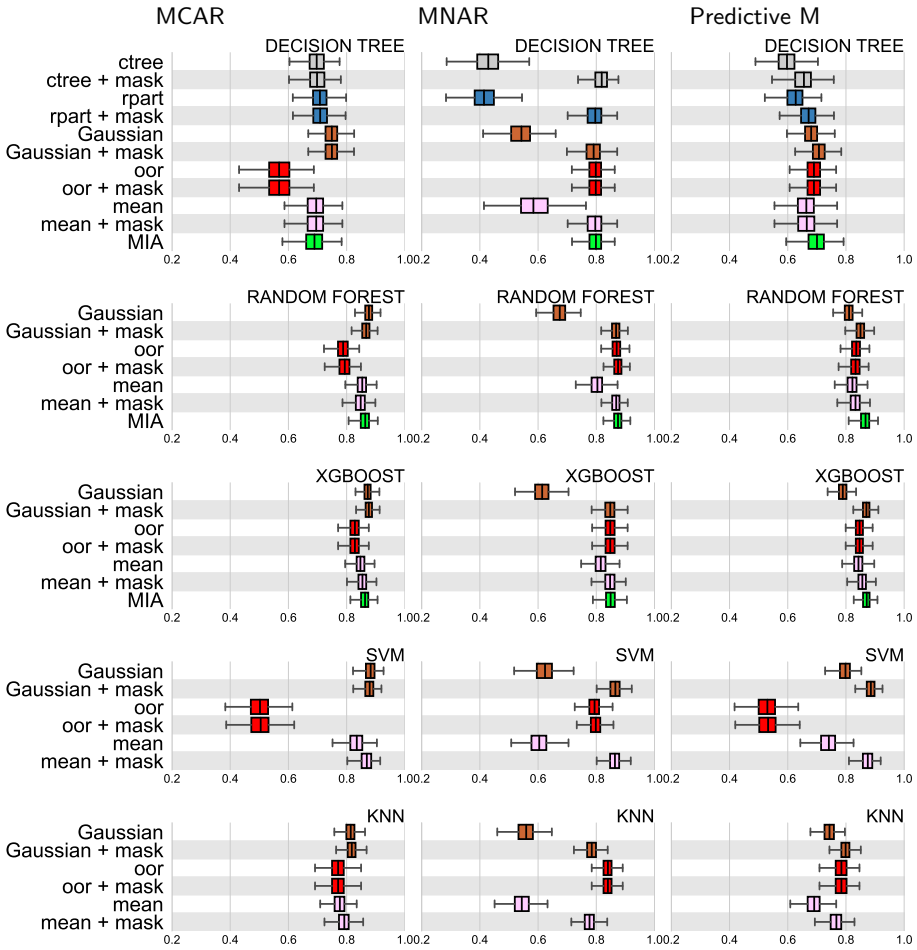


Fig. 1: R^2 **scores on model 1** • Normalized explained variance for different mechanisms with 20% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.5$.

tional imputation, *i.e.* one that captures the relationships between variables such as *impute Gaussian*, slightly helps prediction. This is all the most true as the correlation between variables increases, which we have not displayed. Adding the missing-value mask is not important. For powerful models, random forests and gradient boosting, the benefit of conditional imputation compared to MIA is much reduced. These models give significantly better prediction accuracy than tree-based models as expected.

More complex patterns (MNAR or predictive missingness) reveal the importance of adding the missing mask in most imputation strategies, except for the OOR imputation. MIA achieves excellent performance even for these more complex missing-values mechanisms. Remarkably, mean imputation also achieves good performances with random forests and xgboost methods though adding a mask helps. This is not the case for KNN and SVM. Note that, whatever the missing data mechanism, combining OOR imputation with SVM learners leads to sub-optimal predictive performances.

Statistical tests In order to assess whether the differences observed in Figure 1 are significant, we employ statistical tests. More precisely, for each learning algorithm and each pair of imputation strategy, we consider the R^2 of the 1000 repetitions. As these values are computed on similar individuals (the generated data are the same for all imputation methods and a given repetition), we employ paired t-tests. A p-value lower than 0.05 indicates that the two considered imputation method exhibits different predictive performances. Such results help us to assess the significance of the difference observed via the boxplots in Figure 1. P-values are displayed in Appendix D.2. The vast majority of differences observed in Figure 1 appear to be significant. For example, in the MCAR model, only the pair Gaussian/Gaussian+mask and Mean/Mean+mask are not statistically different for decision trees and random forests at the level 0.05.

Computational complexity The computational complexity together with the predictive performances are two main criteria to assess the usefulness of a method handling missing values. We display in Figure 2 the training time (taking into account the imputation time and the training time on imputed data) of each method. As expected, the computational time increases when adding the mask except for the forests. In this low-dimensional setting, KNN is the quickest method, followed by decision trees and SVMs. The most computationally intensive methods, as expected, are random forests and gradient boosting. Nevertheless, the computational

time for data of this size is not substantial. The computation time is primarily attributable to the learners, and imputation methods have minimal impact.

Figure 3 compares methods for datasets with a non-linear generative model and values missing completely at random. The figure focuses on methods without adding the mask, as it makes little difference in MCAR settings. Even with non linearities, Gaussian imputation often performs as well or better than mean imputation or MIA, likely because the non-linear supervised model compensates for the non linearity. All in all, MIA proves to be a strong option in all the scenarios that we have experimented when using tree-based methods, although Gaussian imputation with the mask can outperform it in these MCAR settings.

5.4 Consistency and learning curves

In the third experiment, we compare the methods of Section 5.2 varying sample size to assess their asymptotic performances, on models 2, 3 and 4. We wish to compare the tree performance with respect to the Bayes loss. For each sample size (between 300 and 10^4), we summarize 200 repetitions by their median and quartiles (as in the boxplots). Assuming MCAR, the Bayes estimator is the expectation of Y conditionally to the observed values,

$$\mathbb{E}[Y|\widetilde{\mathbf{X}}] = \mathbb{E}[f(\mathbf{X})|\widetilde{\mathbf{X}}] = \mathbb{E}[f(\mathbf{X})|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}, \mathbf{M} = \mathbf{m}].$$

It has an simple expression only if the joint distribution of $(\mathbf{X}, Y)$ is Gaussian. To compute an approximate Bayes loss for a nonlinear regression with Gaussian features, we apply joint Gaussian multiple imputation, as justified in Section 4.1.1, on a very large sample. For the third scenario with non-Gaussian features, we have not computed the Bayes loss.

In linear settings, Figure 4 (left) shows that *impute Gaussian* benefits from correlations between features and is the best-performing method; For decision trees and forests, mean imputation, MIA and surrogate splits are also consistent but

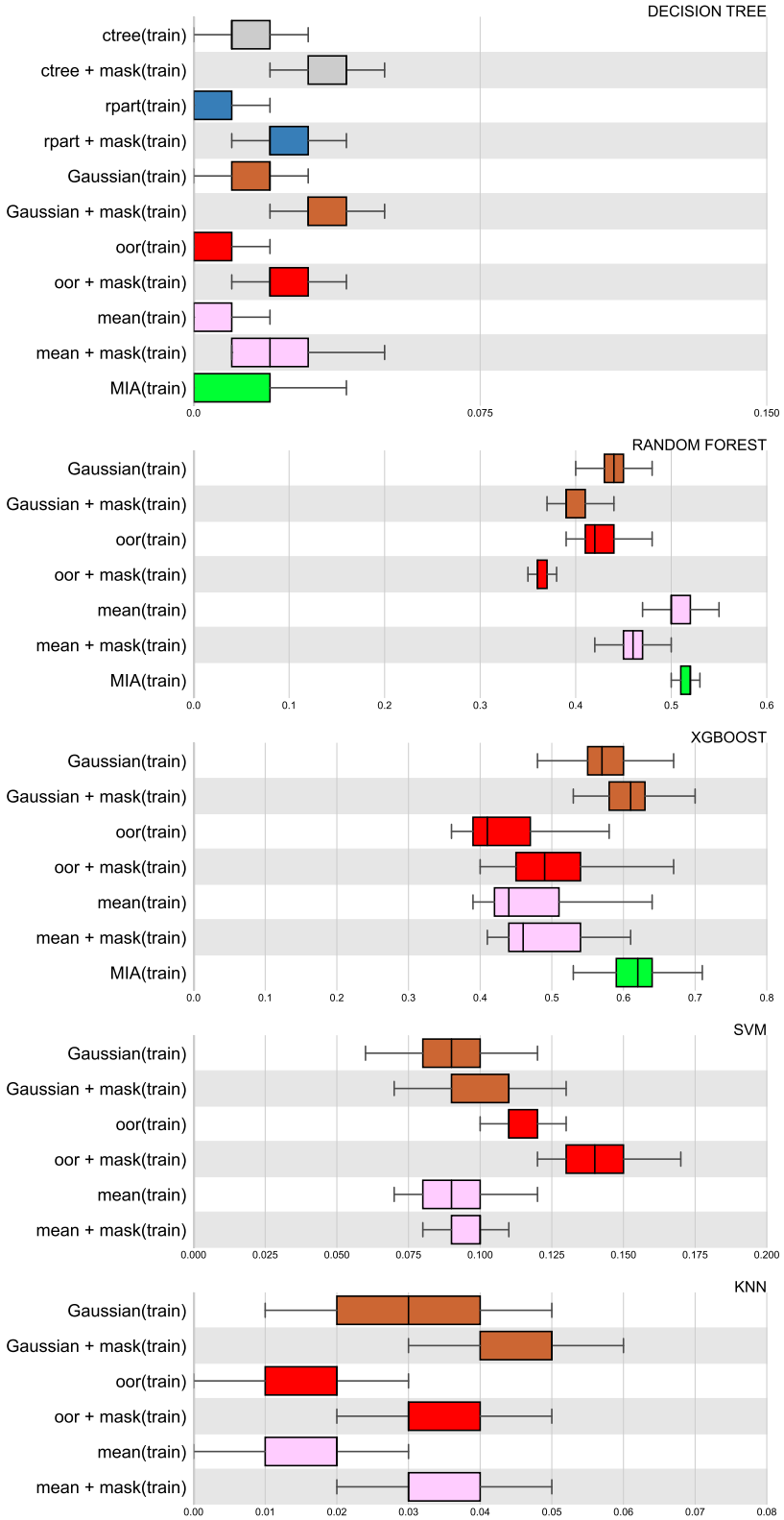


Fig. 2: **Computation time (in seconds)** of the different imputation methods/learning procedures for the MCAR generative mechanism with 20% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.5$.

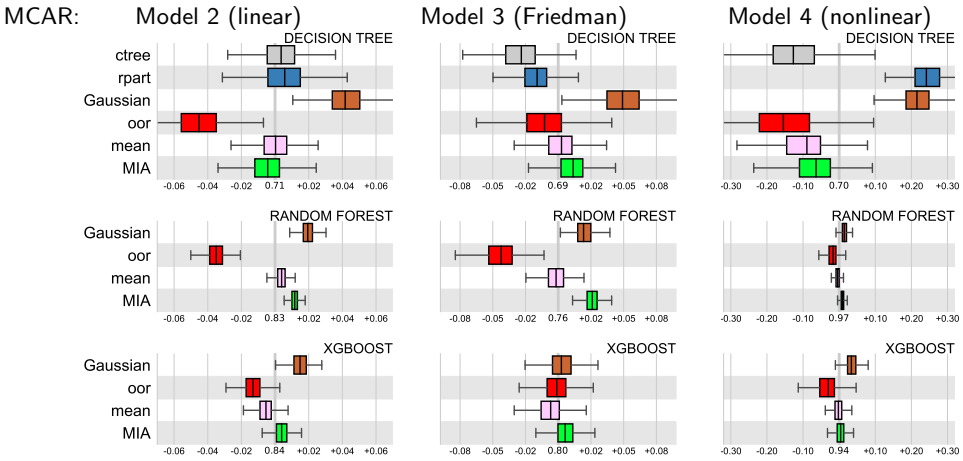


Fig. 3: **Relative scores on different models in MCAR** • Relative explained variance for models 2, 3, 4, MCAR with 20% of missing values, $n = 1000$, $d = 10$ and $\rho = 0.5$.

with a slower convergence rate (we have not displayed conditional trees as they exhibit the same behaviour as rpart with surrogate splits). Adding the indicator matrix in the data changes almost nothing here, so we have not displayed the corresponding curves. For non-linear associations (Figure 4, middle and right), the benefit brought by Gaussian imputation over the others methods seems to carry over though it is less pronounced for random forests and boosting. For low-noise settings (Figure 4, right) MIA and mean imputation seem equivalent. Gaussian imputation is also the best method compared to mean imputation when using KNN or SVM. Note that SVM is not a local averaging method (in contrast to tree-based methods and nearest neighbors) and thus poor imputation may damage the predictive performances in the whole input space. On the contrary, imputed data points have only a local influence in tree-based methods and nearest neighbors, which may make them less sensitive to a bad imputation technique.

For boosting, the difference between methods vanishes with large n, as can be expected from boosting's ability to turn weak learners into strong ones [Schapire, 1990]. Gaussian imputation is still beneficial for small sample sizes. Note that MIA

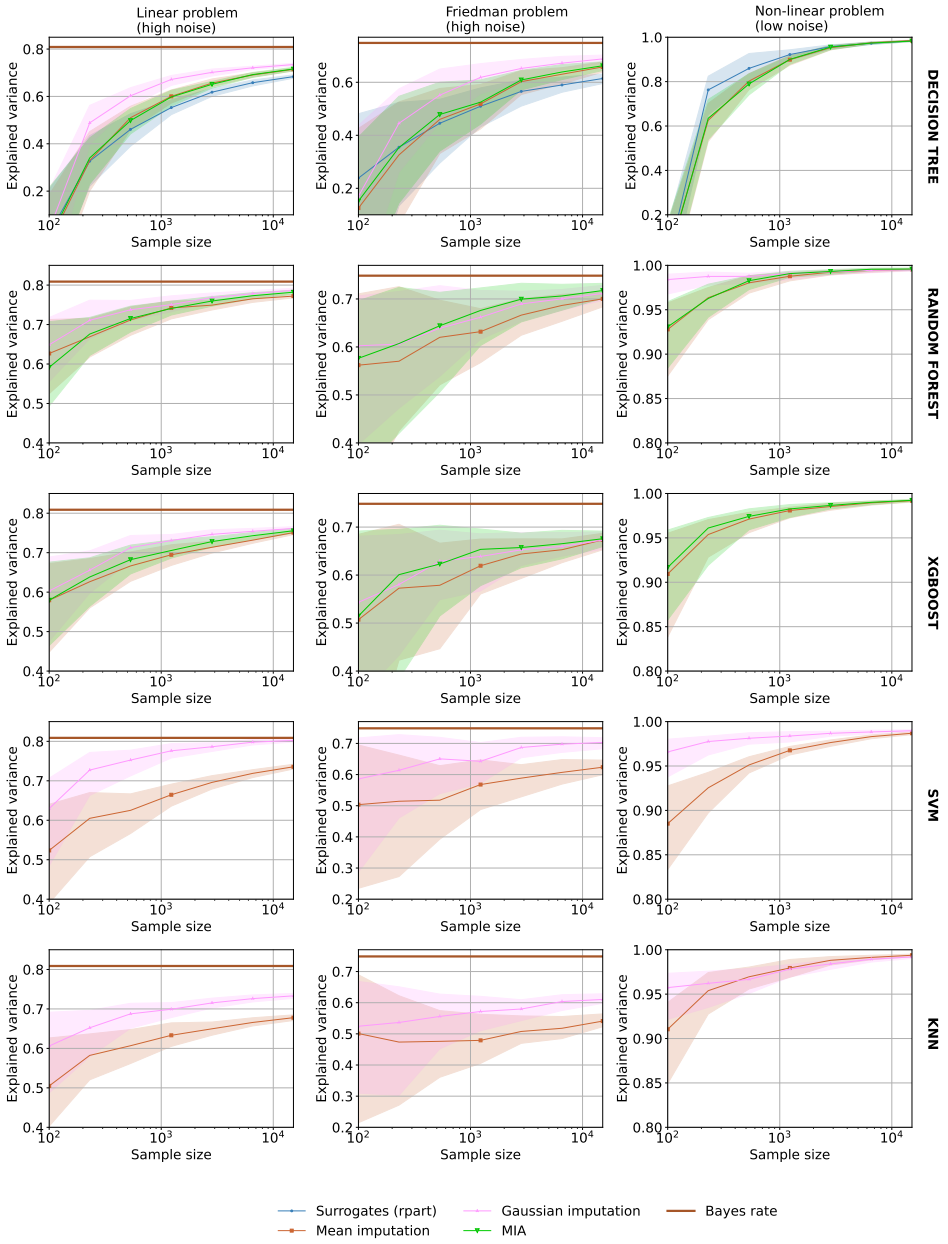


Fig. 4: **Bayes consistency in MCAR** • Consistency with 40% of MCAR values on all variables, on models 2 (linear), 3 (Friedman), 4 (non-linear).

can easily be implemented as a preprocessing step (see Remark 2 in Appendix B.3 for implementation details).

Link to studies on real-life data The experiments above are simple synthetic problems to showcase the principles behind prediction with missing values. A benchmark on real-life data is beyond the scope of this work because it requires studying several datasets for generalizable findings. Perez-Lebel et al. [2022] conducted such study on 13 different prediction tasks on data with missing values. Their findings confirms that our results apply to real-life data. In particular: *1)* given sufficient data, constant imputation (*eg* with the mean) performs as well as conditional imputation; *2)* using MIA in tree-based models gives leading methods for all sample sizes; *3)* often adding the missing indicator $\mathbf{M}$ in the features helps predictions.

Take-home messages First, mean imputation can be appropriate and is consistent in a supervised-learning setting when missing values are MAR and not related to the outcome. Second, tree-based methods are an efficient way to target $\widetilde{f}^{\star}(\widetilde{\mathbf{X}}) = \mathbb{E}[Y|\widetilde{\mathbf{X}}]$ especially when using MIA (Section B.3) and can handle well informative pattern of missing values.

We compare imputation methods, using the "proper way" to impute as described in Section 3.1, *i.e.*, where imputation values from the training set are used to impute the test set.

In addition, we consider imputation with the missing indicator $\mathbf{M}$ in the features. The rationale behind this indicator is that it can be useful to improve the prediction when going beyond the hypothesis of Theorem 3, *i.e.* considering a finite sample, a learning algorithm with a low approximation capacity (as linear regression) and with missing values that can either be MNAR or depend on Y.

6 Discussion and conclusion

We have studied procedures for supervised learning with missing data. Unlike in the classic missing data literature, the goal of the procedures is to yield the best

possible prediction on test data with missing values. Focusing on simple ways of adapting existing procedures, our theoretical and empirical results outline simple practical recommendations:

- In presence of MAR missing values, Bayes-consistent estimate can be built by applying the Bayes predictor (on complete data) to data imputed via conditional multiple imputation, and averaging the resulting prediction (Theorem 2). Thus, Theorem 2 justifies the use of multiple conditional imputation in a MAR setting.
- To train and test on data with missing values, the same imputation model should be used. Constant imputation is consistent, if used in conjunction with a powerful, non-linear learning model (Theorem 3).
- There are a variety of manners to handle missing values inside decision trees. Among them, Missing Incorporated in Attribute (MIA, Twala et al. 2008, see Remark 2 in Appendix B.3), which works by optimizing jointly the split and the handling of the missing values is the most versatile, as it adapts to different missing data scenarios (see Section 5 and theoretical analysis in Appendix B.4).
- Empirically, the choice of imputation methods (applied at train and test time) may lead to a reduction of the number of samples required to reach a given prediction performance (Figure 4).
- When missingness is related to the prediction target, imputation does not suffice and it is useful to add the missingness indicators as features (Example 3 and Figure 1).

These recommendations hold to minimize the prediction error in an asymptotic regime. More work is needed to establish theoretical results in the finite sample regime. In addition, different practices may be needed to control for the uncertainty associated to a prediction.

Declarations

Ethics approval and Consent to participate As the work is purely mathematical and numerical, and all data are synthetic, no ethic approval or consent to participate was necessary.

Competing interests We have no competing interests to disclose.

Funding JJ, NP, and GV acknowledge funding from ANR ("DirtyData" grant ANR-17-CE23-0018) and DataIA ("MissingBigData" grant).

Authors' contributions NP and JJ performed the experiments. NP, JJ, and GV presented their results. GV and JJ oversaw the benchmarking process, provided feedback on the results presentation. ES, JJ, GV, and NP contributed elements of the proofs. JJ, ES, GV, and NP contributed to the literature review and the discussion. JJ, ES, GV, and NP wrote the manuscript, provided feedback, and edited the manuscript.

Acknowledgements We thank Stefan Wager for fruitful discussion and Julie Tibshirani for the suggestion to implement MIA. We also thank Antoine Ogier who initiated our work on imputation in supervised learning during his internship.

Consent for publication Not applicable: no data or individual images were used in this publication

Code availability The code to reproduce the experiments of this work is available on https://github.com/dirty-data/supervised_missing

Availability of data and material - (data transparency) Not applicable as all the experiments in this study use simulated data.

References

Paul D Allison. *Missing data*, volume 136. Sage publications, 2001.

Vincent Arel-Bundock and Krzysztof J Pelc. When can multiple imputation improve regression estimates? *Political Analysis*, 26(2):240–245, 2018.

Gustavo EAPA Batista and Maria Carolina Monard. An analysis of four missing data treatment methods for supervised learning. *Applied artificial intelligence*, 17 (5-6):519–533, 2003.

Dimitris Bertsimas, Colin Pawlowski, and Ying Daisy Zhuo. From predictive methods to missing data imputation: an optimization approach. *Journal of Machine Learning Research*, 18(196):1–39, 2018.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984. ISBN 0-534-98053-8.

S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68, 2010.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *sigkdd international conference on knowledge discovery and data mining*, page 785. ACM, 2016.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27, 1967.

Marie Davidian. Statistical methods for analysis with missing data - course, 2017. URL https://www4.stat.ncsu.edu/~davidian/st790/notes.html.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 2013.

Yufeng Ding and Jeffrey S Simonoff. An investigation of missing data methods for classification trees applied to binary response data. *Journal of Machine Learning Research*, 11:131, 2010.

John Fox. Package 'norm'. 2013.

Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.

Lovedeep Gondara and Ke Wang. Mida: Multiple imputation using denoising autoencoders. In *Advances in Knowledge Discovery and Data Mining: 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part III 22*, pages 260–272. Springer, 2018.

Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16:3367, 2015.

W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

Torsten Hothorn and Achim Zeileis. partykit: A modular toolkit for recursive partytioning in R. *Journal of Machine Learning Research*, 16:3905, 2015.

Torsten Hothorn, Kurt Hornik, and Achim Zeileis. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674, 2006.

Sebastian Jäger, Arndt Allhorn, and Felix Bießmann. A benchmark for data imputation methods. *Frontiers in big Data*, 4:693674, 2021.

Wei Jiang, Julie Josse, and Marc Lavielle. Logistic regression with missing covariates–parameter estimation, model selection and prediction. *Computational and Statistics Analysis*, 2019.

Michael P Jones. Indicator and stratification methods for missing explanatory variables in multiple linear regression. *Journal of the American statistical associ-*

ation, 91(433):222–230, 1996.

Julie Josse and Jerome P. Reiter. Introduction to the special section on missing data. *Statist. Sci.*, 33(2):139–141, 05 2018. doi: 10.1214/18-STS332IN.

Julie Josse, Sylvain Sardy, and Stefan Wager. denoiser: A package for low rank matrix estimation. *Journal of Statistical Software*, 2016.

Adam Kapelner and Justin Bleich. Prediction with missing data via bayesian additive regression trees. *Canadian Journal of Statistics*, 43(2):224–239, 2015.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3146–3154, 2017.

Marine Le Morvan, Nicolas Prost, Julie Josse, Erwan Scornet, and Gaël Varoquaux. Linear predictor on linearly-generated data with missing values: non consistency and solutions. *AISTAT*, 2020.

Steven Cheng-Xian Li, Bo Jiang, and Benjamin Marlin. Misgan: Learning from incomplete data with generative adversarial networks. *arXiv preprint arXiv:1902.09599*, 2019.

Roderick JA Little. Regression with missing x's: a review. *Journal of the American Statistical Association*, 87(420):1227–1237, 1992.

Roderick JA Little and Donald B Rubin. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

Ying Liu, Yuanjia Wang, Yang Feng, and Melanie M. Wall. Variable selection and prediction with incomplete high-dimensional data. *Ann. Appl. Stat.*, 10(1): 418–450, 2016.

Pierre-Alexandre Mattei and Jes Frellsen. MIWAE: Deep generative modelling and imputation of incomplete data sets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4413–4423. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/

mattei19a.html.

Imke Mayer, Sportisse Aude, Nicholas Tierney, Nathalie Vialaneix, and Julie Josse. R-miss-tastic: a unified platform for missing values methods and workflows. *R journal*, 2022.

Xiao-Li Meng. Multiple-imputation inferences with uncongenial sources of input. *Statistical Science*, pages 538–558, 1994.

Karthika Mohan and Judea Pearl. Graphical models for processing missing data. *arXiv preprint arXiv:1801.03583*, 2018.

Jared S. Murray. Multiple imputation: A review of practical and theoretical findings. *Statist. Sci.*, 33(2):142–159, 05 2018.

Jeffrey Näf, Meta-Lina Spohn, Loris Michel, and Nicolai Meinshausen. Imputation scores. *The Annals of Applied Statistics*, 17(3):2452–2472, 2023.

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

Alexandre Perez-Lebel, Gaël Varoquaux, Marine Le Morvan, Julie Josse, and Jean-Baptiste Poline. Benchmarking missing-values approaches for predictive models on health databases. *GigaScience*, 11, 2022.

Jason Poulos and Rafael Valle. Missing data imputation for supervised learning. *Applied Artificial Intelligence*, 32(2):186–196, 2018.

J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. URL https://www.R-project.org/.

Alexis Roche. Em algorithm and variants: An informal tutorial. *arXiv:1105.1476*, 2011.

Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Are loss functions all the same? *Neural Computation*, 16 (5):1063–1076, 2004.

Paul R Rosenbaum and Donald B Rubin. Reducing bias in observational studies using subclassification on the propensity score. *Journal of the American statistical Association*, 79(387):516–524, 1984.

D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. Wiley, 1987.

Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

Maytal Saar-Tsechansky and Foster Provost. Handling missing values when applying classification models. 2007.

Robert E Schapire. The strength of weak learnability. *Machine learning*, 5:197, 1990.

Shaun Seaman, John Galati, Dan Jackson, and John Carlin. What is meant by "missing at random"? *Statistical Science*, 28(2):257–268, 2013.

Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2011.

Jonathan A C Sterne, Ian R White, John B Carlin, Michael Spratt, Patrick Royston, Michael G Kenward, Angela M Wood, and James R Carpenter. Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. *BMJ*, 338, 2009. ISSN 0959-8138.

C. Strobl, A.-L. Boulesteix, and T. Augustin. Unbiased split selection for classification trees based on the gini index. *Computational Statistics and Data Analysis*, 52(1):483–501, 2007.

Masashi Sugiyama, Neil D Lawrence, Anton Schwaighofer, et al. *Dataset shift in machine learning*. The MIT Press, 2017.

Terry Therneau and Beth Atkinson. *rpart: Recursive Partitioning and Regression Trees*, 2018. URL https://CRAN.R-project.org/package=rpart. R package version 4.1-13.

Terry M Therneau, Elizabeth J Atkinson, et al. An introduction to recursive partitioning using the rpart routines, 1997. URL http://www.mayo.edu/hsr/techrpt/61.pdf.

Julie Tibshirani, Susan Athey, and Stefan Wager. *grf: Generalized Random Forests*, 2020. URL https://CRAN.R-project.org/package=grf. R package version 1.1.0.

B. E. T. H. Twala, M. C. Jones, and D. J. Hand. Good methods for coping with missing data in decision trees. *Pattern Recogn. Lett.*, 29(7):950–956, May 2008. ISSN 0167-8655.

S. van Buuren. *Flexible Imputation of Missing Data*. Chapman & Hall/CRC Interdisciplinary Statistics. CRC Press LLC, 2018. ISBN 9781138588318.

Vladimir Naumovich Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

Zhenhua Wang, Olanrewaju Akande, Jason Poulos, and Fan Li. Are deep learning models superior for missing data imputation in surveys? evidence from an empirical comparison. *Survey Methodology*, 48(2):375–399, 2022.

Angela M. Wood, Ian R. White, and Patrick Royston. How should variable selection be performed with multiply imputed data? *Statistics in Medicine*, 27(17): 3227–3246, 2008.

Marvin N Wright and Andreas Ziegler. ranger: A fast implementation of random forests for high dimensional data in c++ and r. *arXiv preprint arXiv:1508.04409*, 2015.

Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. In *International conference on machine learning*, pages 5689–5698. PMLR, 2018.

A Proofs of Section 4

A.1 Proof of Theorem 2

Proof (Proof of Theorem 2: consistency of test-time conditional multiple imputation)

Let $\widetilde{\mathbf{x}} \in (\mathbb{R} \cup \{\texttt{NA}\})^d$. By Assumption 1, $\mathbb{E}[\varepsilon|\mathbf{X}_{obs(\mathbf{M})}] = 0$ a.s., which yields, a.s.,

$$\begin{aligned}\mathbb{E}[Y|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}] &= \mathbb{E}[f^\star(\mathbf{X}) + \varepsilon|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}] \\ &= \mathbb{E}[f^\star(\mathbf{X})|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}] \\ &= \mathbb{E}[f^\star(o(\mathbf{X}_{obs(\mathbf{m})}, \mathbf{X}_{mis(\mathbf{m})}; \mathbf{m}))|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}].\end{aligned}$$

By definition, the multiple imputation procedure described in Theorem 2 is given by

$$\begin{aligned}f^\star_{\mathrm{MI}}(\widetilde{\mathbf{x}}) &= \mathbb{E}_{\mathbf{X}_{mis(\mathbf{m})}|\mathbf{X}_{obs(\mathbf{m})}=\mathbf{x}_{obs(\mathbf{m})}}[f^\star(o(\mathbf{x}_{obs(\mathbf{m})}, \mathbf{X}_{mis(\mathbf{m})}; \mathbf{m}))] \\ &= \mathbb{E}[f^\star(o(\mathbf{X}_{obs(\mathbf{m})}, \mathbf{X}_{mis(\mathbf{m})}; \mathbf{m}))|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}] \\ &= \mathbb{E}[Y|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}]. \qquad (10)\end{aligned}$$

Besides, since the missing pattern is MAR,

$$\begin{aligned}f^\star_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{x}}) &= \mathbb{E}[Y|\widetilde{\mathbf{X}} = \widetilde{\mathbf{x}}] \\ &= \mathbb{E}[Y|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}, \mathbf{M} = \mathbf{m}] \\ &= \mathbb{E}[Y|\mathbf{X}_{obs(\mathbf{m})} = \mathbf{x}_{obs(\mathbf{m})}] \qquad (11)\end{aligned}$$

Combining (10) and (11), we finally obtain

$$f^\star_{\mathrm{MI}}(\widetilde{\mathbf{x}}) = f^\star_{\widetilde{\mathbf{X}}}(\widetilde{\mathbf{x}}).$$

A.2 Proof of Theorem 3

Proof (Proof of Theorem 3: consistency of mean imputation at train and test time)

We distinguish the three following cases in order to make explicit the expression of $\mathbb{E}[Y|\mathbf{X}' = \mathbf{x}]$.

First case : let $\mathbf{x} \in [0,1]^d$ *such that* $x_1 \neq \alpha$.

For $0 < \rho < |x_1 - \alpha|$, letting $B(\mathbf{x}, \rho)$ be the euclidean ball centered at $\mathbf{x}$ of radius ρ,

$$\begin{aligned}\mathbb{E}[Y|\mathbf{X}' \in B(\mathbf{x},\rho)] &= \frac{\mathbb{E}[Y \mathbb{1}_{\mathbf{X}' \in B(\mathbf{x},\rho)}]}{\mathbb{P}[\mathbf{X}' \in B(\mathbf{x},\rho)]} \\ &= \frac{\mathbb{E}[Y \mathbb{1}_{\mathbf{X} \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=0}]}{\mathbb{P}[\mathbf{X} \in B(\mathbf{x},\rho), M_1 = 0]} \\ &= \mathbb{E}[Y|\mathbf{X} \in B(\mathbf{x},\rho), M_1 = 0]\,. \end{aligned} \tag{12}$$

Taking the limit of (12) when ρ tends to zero,

$$\mathbb{E}[Y|\mathbf{X}' = \mathbf{x}] = \lim_{\rho \to 0} \mathbb{E}[Y|\mathbf{X}' \in B(\mathbf{x},\rho)] = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}, M_1 = 0]. \tag{13}$$

Second case : let $\mathbf{x} \in [0,1]^d$ *such that* $x_1 = \alpha$.

First Subcase: assume $\mathbb{P}[M_1 = 1|X_2 = x_2, \ldots, X_d = x_d] = 0$.

We have $\{\mathbf{X}' = \mathbf{x}\} = \{\mathbf{X}' = \mathbf{x}, M_1 = 0\} = \{\mathbf{X} = \mathbf{x}\}$, and consequently,

$$\mathbb{E}[Y|\mathbf{X}' = \mathbf{x}] = \mathbb{E}[Y|\mathbf{X} = \mathbf{x}]. \tag{14}$$

Second Subcase: assume $\mathbb{P}[M_1 = 1|X_2 = x_2, \ldots, X_d = x_d] > 0$.

We have

$$\begin{aligned}\mathbb{P}[\mathbf{X}' \in B(\mathbf{x},\rho)] &= \mathbb{E}[\mathbb{1}_{\mathbf{X}' \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=0}] + \mathbb{E}[\mathbb{1}_{\mathbf{X}' \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=1}] \\ &= \mathbb{E}[\mathbb{1}_{\mathbf{X} \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=0}] + \mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d) \in B((x_2,\ldots,x_d),\rho)} \mathbb{1}_{M_1=1}],\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}[f(\mathbf{X}) \mathbb{1}_{\mathbf{X}' \in B(\mathbf{x},\rho)}] = E[f(\mathbf{X}) \mathbb{1}_{\mathbf{X} \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=0}] \\ + E[f(\mathbf{X}) \mathbb{1}_{(X_2,\ldots,X_d) \in B((x_2,\ldots,x_d),\rho)} \mathbb{1}_{M_1=1}].\end{aligned}$$

Therefore,

$$\begin{aligned}\mathbb{E}[Y|\mathbf{X}' \in B(\mathbf{x},\rho)] =& \frac{\mathbb{E}[f(\mathbf{X}) \mathbb{1}_{\mathbf{X}' \in B(\mathbf{x},\rho)}]}{\mathbb{P}[\mathbf{X}' \in B(\mathbf{x},\rho)]} \\ =& \frac{\mathbb{E}[f(\mathbf{X}) \mathbb{1}_{\mathbf{X} \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=0}] + \mathbb{E}[f(\mathbf{X}) \mathbb{1}_{(X_2,\ldots,X_d) \in B((x_2,\ldots,x_d),\rho)} \mathbb{1}_{M_1=1}]}{\mathbb{E}[\mathbb{1}_{\mathbf{X} \in B(\mathbf{x},\rho)} \mathbb{1}_{M_1=0}] + \mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d) \in B((x_2,\ldots,x_d),\rho)} \mathbb{1}_{M_1=1}]}.\end{aligned} \tag{15}$$

The terms in (15) involving $M_1 = 0$ satisfy

$$\mathbb{E}[\mathbb{1}_{\mathbf{X}\in B(\mathbf{x},\rho)}\mathbb{1}_{M_1=0}] \leq \mu(B(\mathbf{x},\rho)) \leq \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}\|g\|_\infty \rho^d, \tag{16}$$

and

$$\begin{aligned}|\mathbb{E}[f(\mathbf{X})\mathbb{1}_{\mathbf{X}\in B(\mathbf{x},\rho)}\mathbb{1}_{M_1=0}]| &\leq \mathbb{E}[|f(\mathbf{X})|\mathbb{1}_{\mathbf{X}\in B(\mathbf{x},\rho)}]| \\ &\leq \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)}\|g\|_\infty \|f\|_\infty \rho^d. \end{aligned}\tag{17}$$

The second term of the denominator in (15) can be bounded from below,

$$\begin{aligned}&\mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{1}_{M_1=1}] \\ &= \mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{P}[M_1 = 1|X_2,\ldots,X_d]] \\ &\geq \frac{\pi^{(d-1)/2}}{\Gamma(\frac{d-1}{2}+1)}\big(\inf_{[0,1]^d} g\big)\rho^{d-1}\eta. \end{aligned}\tag{18}$$

The second term of the numerator in (15) verifies

$$\begin{aligned}&\mathbb{E}[f(\mathbf{X})\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{1}_{M_1=1}] \\ &= \mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{E}[f(\mathbf{X})\mathbb{1}_{M_1=1}|X_2,\ldots,X_d]] \\ &= \mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{E}[f(\mathbf{X})|X_2,\ldots,X_d]\mathbb{E}[\mathbb{1}_{M_1=1}|X_2,\ldots,X_d]].\end{aligned}$$

If $\mathbb{E}[f(\mathbf{X})|X_2 = x_2,\ldots,X_d = x_d] > 0$, by uniform continuity of f and g,

$$\begin{aligned}&\mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{E}[f(\mathbf{X})|X_2,\ldots,X_d]\mathbb{E}[\mathbb{1}_{M_1=1}|X_2,\ldots,X_d]] \\ &\geq \mathbb{E}[f(\mathbf{X})|X_2 = x_2,\ldots,X_d = x_d]\frac{\pi^{(d-1)/2}}{\Gamma(\frac{d-1}{2}+1)}\big(\inf_{[0,1]^d} g\big)\rho^{d-1}\eta.\end{aligned}$$

Similarly, if $\mathbb{E}[f(\mathbf{X})|X_2 = x_2,\ldots,X_d = x_d] < 0$, we have

$$\begin{aligned}&\mathbb{E}[\mathbb{1}_{(X_2,\ldots,X_d)\in B((x_2,\ldots,x_d),\rho)}\mathbb{E}[f(\mathbf{X})|X_2,\ldots,X_d]\mathbb{E}[\mathbb{1}_{M_1=1}|X_2,\ldots,X_d]] \\ &\leq \mathbb{E}[f(\mathbf{X})|X_2 = x_2,\ldots,X_d = x_d]\frac{\pi^{(d-1)/2}}{\Gamma(\frac{d-1}{2}+1)}\big(\inf_{[0,1]^d} g\big)\rho^{d-1}\eta \\ &\leq 0.\end{aligned}$$

Hence, if $\mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d] \neq 0$

$$\begin{aligned}
&|\mathbb{E}[f(\mathbf{X})\mathbb{1}_{(X_2,\dots,X_d)\in B((x_2,\dots,x_d),\rho)}\mathbb{1}_{M_1=1}]| \\
&\geq |\mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d]| \frac{\pi^{(d-1)/2}}{\Gamma(\frac{d-1}{2}+1)} \big(\inf_{[0,1]^d} g\big)\rho^{d-1}\eta. \qquad (19)
\end{aligned}$$

Gathering inequalities (16)-(19) and using equation (15), we have, if $\mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d] \neq 0$

$$\begin{aligned}
\lim_{\rho\to 0} \mathbb{E}[Y|\mathbf{X}' \in B(\mathbf{x},\rho)] &= \lim_{\rho\to 0} \frac{E[f(\mathbf{X})\mathbb{1}_{(X_2,\dots,X_d)\in B((x_2,\dots,x_d),\rho)}\mathbb{1}_{M_1=1}]}{\mathbb{E}[\mathbb{1}_{(X_2,\dots,X_d)\in B((x_2,\dots,x_d),\rho)}\mathbb{1}_{M_1=1}]} \\
&= \mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d, M_1 = 1].
\end{aligned}$$

Finally, if $\mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d] = 0$ then by uniform continuity of f, there exists ε_ρ such that $\varepsilon_\rho \to 0$ as $\rho \to 0$ satisfying,

$$|\mathbb{E}[f(\mathbf{X})\mathbb{1}_{(X_2,\dots,X_d)\in B((x_2,\dots,x_d),\rho)}\mathbb{1}_{M_1=1}]| \leq \varepsilon_\rho \rho^{d-1} \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} \|g\|_\infty,$$

hence

$$\begin{aligned}
\lim_{\rho\to 0} \mathbb{E}[Y|\mathbf{X}' \in B(\mathbf{x},\rho)] &= 0 \\
&= \mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d] \\
&= \mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d, M_1 = 1],
\end{aligned}$$

since $M_1 \perp\!\!\!\perp X_1 | (X_2, \dots, X_d)$. Consequently, for all $\mathbf{x} \in [0,1]^d$ such that $x_1 = \alpha$,

$$\lim_{\rho\to 0} \mathbb{E}[Y|\mathbf{X}' \in B(\mathbf{x},\rho)] = \mathbb{E}[f(\mathbf{X})|X_2 = x_2, \dots, X_d = x_d, M_1 = 1]. \qquad (20)$$

Combining equations (13), (14) and (20), the prediction given by the mean imputation followed by learning is, for all $\mathbf{x}' \in \mathbb{R}^d$,

$$\begin{aligned}
f^\star_{\widetilde{\mathbf{X}}}(\mathbf{x}') &= \mathbb{E}[Y|X_2 = x_2, \dots, X_d = x_d, M_1 = 1]\mathbb{1}_{x'_1=\alpha}\mathbb{1}_{\mathbb{P}[M_1=1|X_2=x_2,\dots,X_d=x_d]>0} \\
&\quad + \mathbb{E}[Y|\mathbf{X} = \mathbf{x}']\mathbb{1}_{x'_1=\alpha}\mathbb{1}_{\mathbb{P}[M_1=1|X_2=x_2,\dots,X_d=x_d]=0} \\
&\quad + \mathbb{E}[Y|X_2 = x_2, \dots, X_d = x_d, M_1 = 0]\mathbb{1}_{x'_1\neq\alpha},
\end{aligned}$$

which concludes the proof.

B Decision trees: an example of empirical risk minimization with missing data

Aside from using procedures that require imputation prior to learning in order to handle missing values (as theoretically discussed in the previous section), one can use procedures intrinsically able to handle missing values. Among them, decision trees offer a natural way for empirical risk minimization with missing values [Saar-Tsechansky and Provost, 2007, Twala et al., 2008]. This is due to their ability to handle the half-discrete nature of $\tilde{\mathbf{X}}$, as they rely on greedy decisions rather than smooth optimization.

We first present the different approaches available to handle missing values in tree-based methods in Sections B.2 and B.3. We then compare them theoretically in Section B.4, highlighting the interest of using the "missing incorporated in attribute" approach in particular when the missing values are informative. In Section 5, we will compare numerically the two different strategies: imputation prior to learning and learning directly with missing data.

B.1 Tree construction with CART

CART (Classification And Regression Trees, Breiman et al. 1984) is one of the most popular tree algorithm, originally designed for complete data sets. It recursively builds a partition of the input space $\mathcal{X} = \mathbb{R}^d$, and predicts by aggregating the observation labels (average or majority vote) inside each terminal node, also called leaf. For each node $A = \prod_{j=1}^{d} [a_{j,L}, a_{j,R}] \subset \mathbb{R}^d$, CART algorithm finds the best split $(j^\star, z^\star)$ among the set of all eligible splits $\mathcal{S} = \{(j, z), j \in [\![1, d]\!],\ z \in \mathbb{R}, z_j \in [a_{j,L}, a_{j,R}]\}$, where a split is defined by the variable j along which the split is performed and the position z of the split. More precisely, the best split $(j^\star, z^\star)$ in any node A is the solution of the following optimization problem

$$(j^\star, z^\star) \in \underset{(j,z)\in\mathcal{S}}{\operatorname{argmin}} \mathbb{E}\Big[\big(Y - \mathbb{E}[Y|X_j \le z, \mathbf{X} \in A]\big)^2 \cdot \mathbb{1}_{X_j \le z, \mathbf{X}\in A} + \big(Y - \mathbb{E}[Y|X_j > z, \mathbf{X} \in A]\big)^2 \cdot \mathbb{1}_{X_j > z, \mathbf{X}\in A}\Big]. \tag{21}$$

For each cell A, problem (21) can be rewritten as

$$f^\star \in \underset{f\in\mathcal{P}_c}{\operatorname{argmin}} \mathbb{E}\Big[\big(Y - f(\mathbf{X})\big)^2 \mathbb{1}_{\mathbf{X}\in A}\Big], \tag{22}$$

where $\mathcal{P}_c$ is the set of piecewise-constant functions on $A \cap \{x_j \le s\}$ and $A \cap \{x_j > s\}$ for $(j, s) \in \mathcal{S}$. Therefore the optimization problem (22) amounts to solving a least square problem on the subclass of functions $\mathcal{P}_c$. Thus, by minimizing the mean squared error, the CART

procedure targets the quantity $\mathbb{E}[Y|\mathbf{X}]$. In the presence of missing values, this criterion must be adapted and several ways to do so have been proposed. Section B.2 and Section B.3 detail the existing criteria that can be used when dealing with missing values.

B.2 Splitting criterion discarding missing values

A simple option to extend CART methodology in presence of missing values is to select the best split only on the available cases for each variable. More precisely, for any node A, the best split in presence of missing values is a solution of the new optimization problem

$$(j^\star, z^\star) \in \underset{(j,z)\in\mathcal{S}}{\operatorname{argmin}}\, \mathbb{E}\Big[\big(Y - \mathbb{E}[Y|X_j \le z, M_j = 0, \mathbf{X} \in A]\big)^2 \cdot \mathbb{1}_{X_j \le z, M_j = 0, \mathbf{X}\in A} + \big(Y - \mathbb{E}[Y|X_j > z, M_j = 0, \mathbf{X} \in A]\big)^2 \cdot \mathbb{1}_{X_j > z, M_j = 0, \mathbf{X}\in A}\Big], \tag{23}$$

which is nothing but problem (21) computed, for each $j \in \{1, \dots, d\}$, on observed values "$M_j = 0$" only. Note that, by convention, the problem (23) is optimised empirically along the variables j that have at least two observed entries (we do not allow splits to be performed along variables that have less than two observed entries). This splitting strategy is described in Algorithm 1. As the missing values were not used to calculate the criterion, it is still necessary to specify to which cell they are sent. The solution consisting in discarding missing data at each step would lead to a drastic reduction of the data set and is therefore not viable. The different methods to propagate missing data down the tree are explained below.

Algorithm 1 Splitting strategy

1: **Input:** a node A, a sample $\mathcal{D}_n$ of observations falling into A.
2: For each variable $j \in \{1, \dots, d\}$ and position z, compute the CART splitting criterion on observed values "$M_j = 0$" only.
3: Choose a split $(j^\star, z^\star)$ that minimizes the previous criterion (see optimization problem 23).
4: Split the cell A accordingly. Two new cells A_L and A_R are created.
5: **Output:** Split $(j^\star, z^\star)$, A_L, A_R.

*Surrogate splits*Once the best split is chosen via Algorithm 1, surrogate splits search for a split on another variable that induces a data partition close to the original one. More precisely, for a selected split $(j_0^\star, z_0^\star)$, to observations send down the tree with no $j_0^\star$th variable, a new stump, *i.e.*, a tree with one cut, is fitted to the response $\mathbb{1}_{X_{j_0^\star} \le z_0^\star}$, using variables $(X_j)_{j \ne j_0^\star}$. The split $(j_1^\star, z_1^\star)$ which minimizes the misclassification error is selected, and observations are split accordingly. Those that lack both variables $j_0^\star$ and $j_1^\star$ are split with the second best,

$j_2^\star$, and so on until the proposed split has a worse misclassification error than the blind rule of sending all remaining missing values to the same daughter, the most populated one. To predict, the training surrogates are kept. This construction is described in Algorithm 2 and is the default method in `rpart` [Therneau et al., 1997]. Surrogate method is expected to be appropriate when there are relationships between covariates.

*Probabilistic splits*Another option is to propagate missing observations according to a Bernoulli distribution $\mathcal{B}(\frac{\#L}{\#L+\#R})$, where $\#L$ (resp. $\#R$) is the number of points already on the left (resp. right) (see Algorithm 3 in Appendix B.5). This is the default method in C4.5 [Quinlan, 2014].

*Block propagation*A third option is to choose the split on the observed values, and then send all incomplete observations as a block, to a side chosen by minimizing the error (see Algorithm 4 in Appendix B.5). This is the method used in LightGBM [Ke et al., 2017].

Note that Hothorn et al. [2006] proposed conditional trees, a variant of CART which also uses surrogate splits, but adapts the criterion (23) to missing values. Indeed, this criterion implies a selection bias: it leads to underselecting the variables with many missing values due to multiple comparison effects [Strobl et al., 2007]. As a result, it favors variables where many splits are available, and therefore those with fewer missing values. Conditional trees are based on the calculation of a linear statistic of association between Y and each feature X_j, $T = \langle X_j, Y \rangle$. Then, its distribution under the null hypothesis of independence between Y

Algorithm 2 Surrogate strategy

Input: a node A, a sample $\mathcal{D}_n$ of observations falling into A, a split $(j_0^\star, z_0^\star)$ produced by Algorithm 1
Create the variable $\mathbb{1}_{X_{j_0^\star} \leq z_0^\star}$
Let $\mathcal{J} = \{1, \ldots, d\} - \{j_0\}$
while Missing data have not been sent down the tree **do**
  Compute the misclassification error $\varepsilon^\star$ corrresponding to sending all remaining missing observation on the most populated side.
  for all $j \in \mathcal{J}$ **do**
    Fit a tree with one split along j, on data in $\mathcal{D}_n$ with observed values X_j and $X_{j_0^\star}$, in order to predict $\mathbb{1}_{X_{j_0^\star} \leq z_0^\star}$.
    For $j, \in \mathcal{J}$, let ε_j be the misclassification error of the previous trees
    Let $j_{min} \in \text{argmin}_{j \in \mathcal{J}}\ \varepsilon_j$
  end for
  if $\varepsilon_{j_{min}} < \varepsilon^\star$ **then**
    Use the tree built on j_{min} to send data with missing values on j_0 into A_L or A_R, depending on the tree prediction
    $\mathcal{J} \leftarrow \mathcal{J} \cup \{j_{min}\}$
  else
    Send all remaining missing values to the most populated cell (A_L or A_R)
  end if
end while

and X_j is estimated by permutation, and the variable with the smallest p-value is selected. As illustrated in Appendix C.1, conditional trees are meant to improve the selection of the splitting variables but do not ensure an improvement in prediction performance.

B.3 Splitting criterion with missing values: MIA

A second class of methods uses missing values to compute the splitting criterion itself. Consequently, the splitting location depends on the missing values, contrary to all methods presented in Section B.2. Its most common instance is "missing incorporated in attribute" (MIA, Twala et al. 2008,). More specifically, MIA considers the following splits, for all splits (j, z):

- $\{\widetilde{X}_j \leq z \text{ or } \widetilde{X}_j = \texttt{NA}\}$ vs $\{\widetilde{X}_j > z\}$,
- $\{\widetilde{X}_j \leq z\}$ vs $\{\widetilde{X}_j > z \text{ or } \widetilde{X}_j = \texttt{NA}\}$,
- $\{\widetilde{X}_j \neq \texttt{NA}\}$ vs $\{\widetilde{X}_j = \texttt{NA}\}$.

In a nutshell, for each possible split, MIA tries to send all missing values to the left or to the right, and compute for each choice the corresponding error (right-hand side in 21, as well as the error associated to separating the observed values from the missing ones. Finally, it chooses the split among the previous ones with the lowest error (see Algorithm 5 in Appendix B.5). Note that block propagation can be seen as a greedy way of successively optimizing the choices in two first options. However, as we show in Prop. 1, these successive choices are sub-optimal. Missing values are treated as a category by MIA, which is thus nothing but a greedy algorithm minimizing the square loss between Y and a function of $\widetilde{\mathbf{X}}$ and consequently targets the quantity (4) which separate $\mathbb{E}[Y|\widetilde{\mathbf{X}}]$ into 2^d terms. However, it is not exhaustive: at each step, the tree can cut for each variable according to missing or non missing and selects this cut when it is relevant, *i.e.* when it minimizes the prediction error. The final leaves can correspond to a cluster of missing values patterns (observations with missing values on the two first variables for instance and any missing patterns for the other variables).

MIA is thought to be a good method to apply when missing pattern is informative, as this procedure allows to cut with respect to missing/ non missing and uses missing data to compute the best splits. Note this latter property implies that the MIA approach does not require a different method to propagate missing data down the tree. Notably, MIA is implemented in the R packages `partykit` [Hothorn and Zeileis, 2015] and `grf` [Tibshirani et al., 2020], as well as in XGBoost [Chen and Guestrin, 2016] and for the `HistGradientBoosting` models in scikit-learn [Pedregosa et al., 2011].

Remark 2 **Implementation:** A simple way to implement MIA consists in duplicating the incomplete columns, and replacing the missing entries once by $+\infty$ and once by $-\infty$ (or an

extreme out-of-range value). This creates two dummy variables for each original one containing missing values. Splitting along a variable and sending all missing data to the left (for example) is the same as splitting along the corresponding dummy variable where missing entries have been completed by $-\infty$. Alternatively, MIA can be with two scans on a feature's values in ascending and descending orders [Chen and Guestrin, 2016, Alg 3].

Remark 3 **Implicit imputation:** Whether it is in the case where the missing values are propagated in the available case method (Section B.2), or incorporated in the split choice in MIA, missing values are assigned either to the left or the right interval. Consequently, handling missing values in a tree can be seen as implicit imputation by an interval value.

B.4 Theoretical comparison of CART versus MIA

We now compare theoretically the positions of the splitting point at the root and the prediction errors on simple examples with MCAR values. Proposition 1 computes the splitting position of MIA and CART, and highlights that the splitting position of MIA varies even for MCAR missing data. Proposition 2 then compares the risk of the different splitting strategies: probabilistic split, block propagation, surrogate split, and MIA. We prove that MIA and surrogate splits are the two best strategies, one of which may be better than the other depending on the dependence structure of covariates.

Proposition 1 *Let $p \in [0,1]$. Consider the regression model*

$$\begin{cases} Y &= X_1 \\ X_1 &\sim U([0,1]) \end{cases}, \quad \begin{cases} \mathbb{P}[M_1 = 0] = 1-p \\ \mathbb{P}[M_1 = 1] = p \end{cases},$$

where $M_1 \perp\!\!\!\perp (X_1, Y)$ is the missingness pattern on X_1. Let $C_{MIA}(1, s, q, p)$ be the value of the splitting MIA criterion computed on X_1 at threshold s such that $(1, s) \in \mathcal{S}$, and $q \in \{L, R\}$, where q stands for the side where missing values are sent. Therefore,

1. *The best split $s^\star$ given by the CART criterion (23) is $s^\star = 1/2$.*
2. *The best splits $s^\star_{MIA,L}(p)$ and $s^\star_{MIA,R}(p)$ given by the MIA procedure (described in Section B.3), assuming that all missing values are sent to the left node (resp. to the right node), satisfy*

$$s^\star_{MIA,L}(p) = \operatorname*{argmin}_{s \in [0,1]} C_{MIA}(1, s, L, p), \tag{24}$$

where

$$C_{MIA}(1, s, L, p) = \frac{1}{3} - \frac{1}{p + (1-p)s}\Big(\frac{p}{2} + \frac{(1-p)s^2}{2}\Big)^2 - (1-p)(1-s)\Big(\frac{1+s}{2}\Big)^2,$$

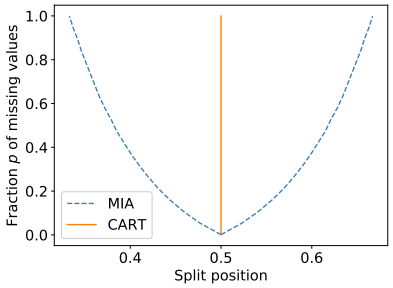


Fig. 5: Split position chosen by MIA and CART criterion, depending on the fraction p of missing values on X_1.

and $s^\star_{MIA,R}(p) = 1 - s^\star_{MIA,L}(p)$.

The proof is given in Appendix A. Proposition 1 shows that the split given by optimizing the CART criterion does not depend on the percentage p of missing values since the pattern is independent of (X, Y). A numerical solution to equation (24) is displayed in Figure 5. When there are no missing values ($p = 0$), the split occur at $s = 1/2$ as expected. When p increases, the threshold does not correspond anymore to the one calculated using observed values only as it is influenced by the missing entries even in the MCAR setting. It may be surprising that the splitting position change in a MCAR setting but the missing values correspond to data from the whole interval $[0, 1]$ and thus introduce noise in the side they are sent to. This implies that the cell that receives missing values must be bigger than usual so that the noise of the missing data is of the same order than the noise of original observations belonging to the cell. Recall that, since the threshold in MIA is chosen by taking into account missing values, it is straightforward to propagate a new element with missing values down the tree.

Recall that the quadratic risk R of a function $f^\star$ is defined as $R(f^\star) = \mathbb{E}[(Y - f^\star(X))^2]$. Proposition 2 enables us to compare the risk of a tree with a single split computed with the different strategies. It highlights that even in the simple case of MCAR, MIA gives more accurate predictions than block propagation or probabilistic split.

Proposition 2 *Consider the regression model*

$$\begin{cases} Y = X_1 \\ X_1 \sim U([0,1]) \\ X_2 = X_1 \mathbb{1}_{W=1} \end{cases}, \quad \begin{cases} \mathbb{P}[W=0] = \eta \\ \mathbb{P}[W=1] = 1-\eta \end{cases}, \quad \begin{cases} \mathbb{P}[M_1=0] = 1-p \\ \mathbb{P}[M_1=1] = p, \end{cases},$$

where $(M_1, W) \perp\!\!\!\perp (X_1, Y)$. *The random variable* M_1 *is the pattern of missingness for* X_1 *and* W *stands for the link between* X_1 *and* X_2. *Let* $f^\star_{MIA}$, $f^\star_{block}$, $f^\star_{prob}$, $f^\star_{surr}$ *be respectively, the theoretical prediction resulting from one split according to MIA, CART with block propagation and CART with probabilistic splitting strategy, and a single split, where missing data are*

handled via surrogate split (in the infinite sample setting). We have

$$
\begin{aligned}
R(f^\star_{MIA}) &= \min_{s\in[0,1]} C_{MIA}(1,s,L,p)\mathbb{1}_{p\leq\eta} + \min_{s\in[0,1]} C_{MIA}(1,s,L,\eta)\mathbb{1}_{p>\eta},\\
R(f^\star_{block}) &= C_{MIA}(1,1/2,L,p) = C_{MIA}(1,1/2,R,p)\\
R(f^\star_{prob}) &= -\frac{p^2}{16} + \frac{p}{8} + \frac{1}{48},\\
R(f^\star_{surr}) &= \frac{1}{48} + \frac{6}{48}\eta p.
\end{aligned}
$$

where $C_{MIA}(1,s,L,p)$ is defined in Proposition 1. In particular,

$$R(f^\star_{MIA}) \leq R(f^\star_{block}) \quad \textit{and} \quad R(f^\star_{MIA}) \leq R(f^\star_{prob}).$$

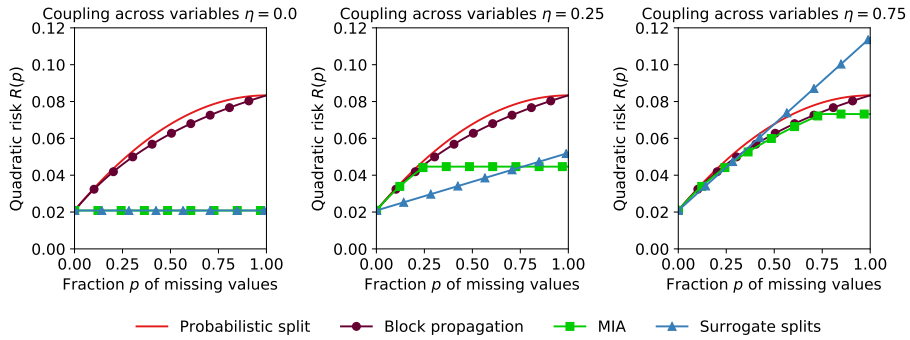


Fig. 6: Theoretical risk of the splitting methods as a function of p, for three values of η parameter that controls the amount of coupling between X_1 and X_2 in the model of Proposition 2.

Proof is given in Appendix A. Figure 6 depicts the risk of each estimate, in the context of proposition 2, resulting from a split computed via one of the four methods described above. Only surrogate and MIA risks depend on the value η which measures the independence between X_1 and X_2. As proved, the risk of probabilistic split and block propagation is larger than that of MIA. Besides, surrogate split is better than MIA if the link between X_1 and X_2 is strong (small values of η) and worse if this link is weak (high values of η).

B.5 Splitting algorithms

Algorithm 3 Probabilistic split

Input: a node A, a sample $\mathcal{D}_n$ of observations falling into A, a split $(j_0^\star, z_0^\star)$ produced by Algorithm 1
Compute the number n_L of points with observed $X_{j_0^\star}$ falling into A_L.
Compute the number n_R of points with observed $X_{j_0^\star}$ falling into A_R.
for all data with missing value along $j_0^\star$ **do**
    Send the data randomly to A_L (resp. A_R) with probability $n_L/(n_L + n_R)$ (resp. $n_R/(n_L + n_R)$)
end for

Algorithm 4 Block propagation

Input: a node A, a sample $\mathcal{D}_n$ of observations falling into A, a split $(j_0^\star, z_0^\star)$ produced by Algorithm 1
Consider sending all observations with missing values on $j_0^\star$ into A_L. Compute the corresponding error (criterion on the right-hand side in 21)
Consider sending all observations with missing values on $j_0^\star$ into A_R. Compute the corresponding error (criterion on the right-hand side in 21).
Choose the alternative with the lowest error and send all missing data on the same side accordingly.

Algorithm 5 Missing Incorporated in Attribute (MIA)

Input: a node A, a sample $\mathcal{D}_n$ of observations falling into A
for each split (j, z) **do**
    Send all observations with missing values on j on the left side. Compute the error $\varepsilon_{j,z,L}$ (right-hand side in 21)
    Send all observations with missing values on j on the right side. Compute the error $\varepsilon_{j,z,R}$ (right-hand side in 21)
end for
for each $j \in \{1, \dots, d\}$ **do**
    Compute the error $\varepsilon_{j,-,-}$ associated to separating observations with missing data on j from the remaining ones.
end for
Choose the split corresponding to the lowest error $\varepsilon_{\cdot,\cdot,\cdot}$. Split the data accordingly.

B.6 Proof of Proposition 1

Cart splitting criterion. Under the model given in Proposition 1, simple calculations show that

$$\mathbb{E}[Y|X \in [0,s]] = \frac{s}{2}, \quad \mathbb{E}[Y^2|X \in [0,s]] = \frac{s^2}{3}$$
$$\mathbb{E}[Y|X \in [s,1]] = \frac{1+s}{2}, \quad \mathbb{E}[Y^2|X \in [s,1]] = \frac{1-s^3}{3(1-s)}$$
$$\mathbb{P}[X \in [0,s]] = s, \quad \mathbb{P}[X \in [s,1]] = 1-s.$$

Thus the CART spltting criterion can be written as

$$\begin{aligned} C(1,s) &= \mathbb{E}[Y^2] - (\mathbb{P}[X \in [0,s]](\mathbb{E}[Y|X \in [0,s]])^2 + \mathbb{P}[X \in [s,1]](\mathbb{E}[Y|X \in [s,1]])^2) \\ &= \frac{1}{3} - \left(s\left(\frac{s}{2}\right)^2 + (1-s)\left(\frac{1+s}{2}\right)^2\right) \\ &= \frac{s(s-1)}{4} + \frac{1}{12}. \end{aligned}$$

By definition,

$$s^\star = \underset{s\in[0,1]}{\operatorname{argmin}} \left(\frac{1}{4}s(s-1) + \frac{1}{12}\right) = 1/2,$$

and the criterion evaluated in $s = 1/2$ is equal to $1/48$. The calculations are exactly the same when a percentage of missing value is added if $M_1 \perp\!\!\!\perp X_1$.

MIA splitting criterion. By symmetry, we can assume than missing values are sent left. It is equivalent to observing

$$X' = 0\mathbb{1}_{M=1} + X\mathbb{1}_{M=0}.$$

The MIA splitting criterion is then defined as

$$\begin{aligned} s^\star_{\text{MIA,L}} &= \underset{s\in[0,1]}{\operatorname{argmin}}\ \mathbb{E}\left[\left(Y - \mathbb{E}[Y|X' \le s]\mathbb{1}_{X'\le s} - \mathbb{E}[Y|X' > s]\mathbb{1}_{X'>s}\right)^2\right] \\ &= \underset{s\in[0,1]}{\operatorname{argmin}}\ \mathbb{P}(X' \le s)\mathbb{E}\left[\left(Y - \mathbb{E}[Y|X' \le s]\right)^2 \middle| X' \le s\right] \\ &\quad + \mathbb{P}(X' > s)\mathbb{E}\left[\left(Y - \mathbb{E}[Y|X' > s]\right)^2 \middle| X' > s\right]. \end{aligned}$$

We have

$$\begin{aligned} \mathbb{E}[Y|X' \in [0,s]] &= \mathbb{E}[X|X' \in [0,s]] \\ &= \mathbb{E}[X\mathbb{1}_{M=1} + X\mathbb{1}_{M=0}|X' \in [0,s]] \\ &= \frac{1}{\mathbb{P}[X' \in [0,s]]}\mathbb{E}[X\mathbb{1}_{M=1,X'\in[0,s]} + X\mathbb{1}_{M=0,X'\in[0,s]}] \\ &= \frac{1}{p + (1-p)s}\Big(\frac{p}{2} + \frac{(1-p)s^2}{2}\Big). \end{aligned}$$

Besides,

$$\begin{aligned}\mathbb{E}[Y^2|X' \in [0,s]] &= \mathbb{E}[X^2|X' \in [0,s]] \\ &= \mathbb{E}[X^2\mathbb{1}_{M=1} + X^2\mathbb{1}_{M=0}|X' \in [0,s]] \\ &= \frac{1}{p+(1-p)s}\mathbb{E}[X^2\mathbb{1}_{M=1,X'\in[0,s]} + X^2\mathbb{1}_{M=0,X'\in[0,s]}] \\ &= \frac{1}{p+(1-p)s}\Big(\frac{p}{3} + \frac{(1-p)s^3}{3}\Big)\end{aligned}$$

Thus the left-part of the criterion is given by

$$\begin{aligned}&\mathbb{P}(X' \in [0,s])\mathbb{E}[(Y - \mathbb{E}[Y|X' \in [0,s]])^2|X' \in [0,s]] \\ &= \Big(p+(1-p)s\Big)\Big(\mathbb{E}[Y^2|X' \in [0,s]] - (\mathbb{E}[Y|X' \in [0,s]])^2\Big) \\ &= \Big(p+(1-p)s\Big)\Big(\frac{1}{p+(1-p)s}\Big(\frac{p}{3} + \frac{(1-p)s^3}{3}\Big) \\ &\quad - \Big(\frac{1}{p+(1-p)s}\Big(\frac{p}{2} + \frac{(1-p)s^2}{2}\Big)\Big)^2\Big) \\ &= \Big(\frac{p}{3} + \frac{(1-p)s^3}{3}\Big) - \frac{1}{p+(1-p)s}\Big(\frac{p}{2} + \frac{(1-p)s^2}{2}\Big)^2\end{aligned}$$

On the other hand, we have

$$\begin{aligned}\mathbb{E}[Y|X' \in [s,1]] &= \mathbb{E}[X|X' \in [s,1]] \\ &= \mathbb{E}[X\mathbb{1}_{M=1} + X\mathbb{1}_{M=0}|X' \in [s,1]] \\ &= \frac{1}{(1-p)(1-s)}\mathbb{E}[X\mathbb{1}_{M=1,X'\in[s,1]} + X\mathbb{1}_{M=0,X'\in[s,1]}] \\ &= \frac{1}{(1-p)(1-s)}\Big((1-p)\frac{1-s^2}{2}\Big) \\ &= \frac{1+s}{2}.\end{aligned}$$

Besides,

$$\begin{aligned}
\mathbb{E}[Y^2|X' \in [s,1]] &= \mathbb{E}[X^2|X' \in [s,1]] \\
&= \mathbb{E}[X^2\mathbb{1}_{M=1} + X^2\mathbb{1}_{M=0}|X' \in [s,1]] \\
&= \frac{1}{(1-p)(1-s)}\mathbb{E}[X^2\mathbb{1}_{M=1,X'\in[s,1]} + X^2\mathbb{1}_{M=0,X'\in[s,1]}] \\
&= \frac{1}{(1-p)(1-s)}\Big((1-p)\frac{1-s^3}{3}\Big) \\
&= \frac{1-s^3}{3(1-s)}.
\end{aligned}$$

Thus the right-part of the criterion is given by

$$\begin{aligned}
&\mathbb{P}(X' \in [s,1])\mathbb{E}[(Y - \mathbb{E}[Y|X' \in [s,1]])^2|X' \in [s,1]] \\
&= \Big((1-p)(1-s)\Big)\Big(\mathbb{E}[Y^2|X' \in [s,1]] - (\mathbb{E}[Y|X' \in [s,1]])^2\Big) \\
&= \Big((1-p)(1-s)\Big)\Big(\frac{1-s^3}{3(1-s)} - (\frac{1+s}{2})^2\Big) \\
&= (1-p)\frac{1-s^3}{3} - (1-p)(1-s)\Big(\frac{1+s}{2}\Big)^2.
\end{aligned}$$

Finally,

$$\begin{aligned}
s^\star_{\mathrm{MIA,L}} = \underset{s\in[0,1]}{\operatorname{argmin}} \Bigg\{ &\Big(\frac{p}{3} + \frac{(1-p)s^3}{3}\Big) - \frac{1}{p+(1-p)s}\Big(\frac{p}{2} + \frac{(1-p)s^2}{2}\Big)^2 \\
&+ (1-p)\frac{1-s^3}{3} - (1-p)(1-s)\Big(\frac{1+s}{2}\Big)^2\Bigg\},
\end{aligned}$$

which concludes the proof.

B.7 Proof of proposition 2

Probabilistic and block propagation. First, note that the variable $X_2 = X_1\mathbb{1}_{W=1}$ is similar to the variable studied for the computation of the MIA criterion in Proposition 1. Therefore, the value of the CART splitting criterion along the first variable is $C_{\mathrm{MIA}}(1, 1/2, \mathrm{L}, 0)$ and its value along the second variable is $C_{\mathrm{MIA}}(2, s^\star_{\mathrm{MIA,L}}, \mathrm{L}, \eta)$. Since the function

$$\alpha \mapsto C_{\mathrm{MIA}}(\cdot, s^\star_{\mathrm{MIA,L}}, \mathrm{L}, \alpha)$$

is increasing, splitting along the first variable leads to the largest variance reduction. Thus, for probabilistic and block propagation, splits occur along the first variable. Let us now compare the value of these criteria. We have

$$\mathbb{P}[X_1 \leq 1/2] = \mathbb{P}[X_1 \geq 1/2] = 1/2.$$

The quantities related to the left cell are given by

$$\mathbb{E}[Y|X_1 \leq 1/2] = \frac{p+1}{4} \quad \text{and} \quad \mathbb{E}[Y^2|X_1 \leq 1/2] = \frac{p}{4} + \frac{1}{12}.$$

The quantities related to the right cell are given by

$$\mathbb{E}[Y|X_1 \geq 1/2] = \frac{3-p}{4} \quad \text{and} \quad \mathbb{E}[Y^2|X_1 \geq 1/2] = \frac{7}{12} - \frac{p}{4}.$$

Thus, the value of the criterion satisfies

$$\begin{aligned} R(f^\star_{\text{prob}}) &= \frac{1}{2}\Big(\frac{p}{4} + \frac{1}{12} - \Big(\frac{p+1}{4}\Big)^2\Big) \\ &\quad + \frac{1}{2}\Big(\frac{7}{12} - \frac{p}{4} - \Big(\frac{3-p}{4}\Big)^2\Big) \\ &= -\frac{p^2}{16} + \frac{p}{8} + \frac{1}{48}. \end{aligned}$$

Let, for all $p \in [0,1]$,

$$\begin{aligned} h(p) &= R(f^\star_{\text{prob}}) - R(f^\star_{\text{block}}) \\ &= -\frac{p^2}{16} + \frac{p}{8} + \frac{1}{48} - \left(-\frac{11}{48} + \frac{1}{8}\frac{3p+2}{2p+1}\right) \\ &= -\frac{p^2}{16} + \frac{p}{8} + \frac{1}{16} - \frac{1}{16}\frac{1}{2p+1}. \end{aligned}$$

We have,

$$h'(p) = -\frac{p}{8} + \frac{1}{8} + \frac{1}{8(2p+1)^2},$$

and consequently,

$$h''(p) = -\frac{1}{8} - \frac{1}{2(2p+1)^3}.$$

An inspection of the variation of h reveals that $h(p) \geq 0$ for all $p \in [0,1]$, which concludes the first part of the proof.

MIA. As noticed above, the criterion computed along the second variable is given by

$$C_{\text{MIA}}(2, s^\star_{\text{MIA,L}}, \text{L}, \eta)$$

Since the function

$$\alpha \mapsto C_{\text{MIA}}(\cdot, s^\star_{\text{MIA,L}}, \text{L}, \alpha)$$

is increasing, MIA split will occur along the first variable if $p \leq \eta$ and along the second variable if $p \geq \eta$. Therefore, the risk of the MIA splitting procedure is given by

$$R(f^\star_{\text{MIA}}) = \min_{s\in[0,1]} C_{\text{MIA}}(1, s, \text{L}, p)\mathbb{1}_{p\leq\eta} + \min_{s\in[0,1]} C_{\text{MIA}}(1, s, \text{L}, \eta)\mathbb{1}_{p>\eta}.$$

Surrogate split. Consider the model $Y = X_1$ and $X_2 = X_1\mathbb{1}_{W=1}$, where $\mathbb{P}[W = 0] = \eta$. Let us determine the best split along X_2 to predict $Z = \mathbb{1}_{X_1<0.5}$. Since $\{X_2 \leq s\} = \{X_1 \leq s, W = 1\} \cup \{W = 0\}$, and $\{X_2 > s\} = \{X_1 > s, W = 1\}$,

$$\mathbb{P}[X_2 \leq s] = s(1-p) + p \quad \text{and} \quad \mathbb{P}[X_2 > s] = (1-s)(1-p).$$

Consequently,

$$\begin{aligned}
\mathbb{E}[Z|X_2 \leq s] &= \frac{\mathbb{E}[\mathbb{1}_{X_1\leq 0.5, X_2\leq s}]}{\mathbb{P}[X_2 \leq s]} \\
&= \frac{1}{s(1-p)+p}\mathbb{E}[\mathbb{1}_{X_1\leq 0.5, X_1\leq s, W=1} + \mathbb{1}_{X_1\leq 0.5, W=0}] \\
&= \frac{1}{s(1-p)+p}\big[(1-p)\min(0.5, s) + \frac{p}{2}\big].
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[Z|X_2 \geq s] &= \frac{\mathbb{E}[\mathbb{1}_{X_1\leq 0.5, X_2> s}]}{\mathbb{P}[X_2 > s]} \\
&= \frac{1}{(1-s)(1-p)}\mathbb{P}[X_1 \leq 0.5, W = 1, X_1 \geq s] \\
&= \frac{(0.5-s)_+}{1-s}.
\end{aligned}$$

Besides, note that $\mathbb{E}[Z^2] = \mathbb{P}[X_1 \leq 0.5] = 0.5$. Therefore, the splitting criterion to predict $\mathbb{1}_{X_1 \leq 0.5}$ with X_2 is given by

$$\begin{aligned} f(s) &= \frac{1}{2} - \mathbb{P}[X_2 \leq s](\mathbb{E}[Z|X_2 \leq s])^2 - \mathbb{P}[X_2 > s](\mathbb{E}[Z|X_2 > s])^2 \\ &= \frac{1}{2} - \frac{1}{s(1-p)+p}\Big((1-p)\min(0.5, s) + \frac{p}{2}\Big)^2 - \frac{1-p}{1-s}((0.5-s)_+)^2. \end{aligned}$$

For $s \geq 1/2$,

$$h(s) = \frac{1}{2} - \frac{1}{4(s(1-p)+p)},$$

which is minimal for $s = 1/2$. For $s \leq 1/2$,

$$h(s) = \frac{1}{2} - \frac{1}{4}\left(\frac{p^2}{p + s(1-p)} + \frac{1-p}{1-s}\right).$$

Hence,

$$h'(s) = -\frac{1-p}{4}\frac{(1-2p)s^2 + 2ps}{(1-s)^2(s(1-p)+p)^2}.$$

Let $g(s) = (1-2p)s^2 + 2ps$. If $p \leq 1/2$, the solutions of $g(s) = 0$ are negative, thus, $g(s) \geq 0$ for all $s \in [0, 1/2]$ and thus the minimum of h is reached at $s = 1/2$. If $p \geq 1/2$, one solution of $g(s) = 0$ is zero and the other is $s = 2p/(2p-1) > 1$. Thus, $g(s) \geq 0$ for all $s \in [0, 1/2]$ and the minimum of h is reached at $s = 1/2$. Finally, the minimum of h is reached at $s = 1/2$. The risk of the surrogate estimate is then given by

$$\begin{aligned} R(f^\star_{\text{surr}}) &= \mathbb{E}[(Y - f^\star_{\text{surr}}(\mathbf{X}))^2] \\ &= \mathbb{E}[(Y - f^\star_{\text{surr}}(\mathbf{X}))^2\mathbb{1}_{M_1=0} + (Y - f^\star_{\text{surr}}(\mathbf{X}))^2\mathbb{1}_{M_1=1}]. \end{aligned}$$

Here,

$$\begin{aligned} &\mathbb{E}[(Y - f^\star_{\text{surr}}(\mathbf{X}))^2|M_1 = 1] \\ &= \mathbb{E}[(X_1 - 0.25)^2\mathbb{1}_{X_2<0.5} + (X_1 - 0.75)^2\mathbb{1}_{X_2 \geq 0.5}] \\ &= \eta\mathbb{E}[(X_1 - 0.25)^2] + (1-\eta)\mathbb{E}[(X_1 - 0.25)^2\mathbb{1}_{X_1 \leq 0.5}] \\ &\quad + (1-\eta)\mathbb{E}[(X_1 - 0.75)^2\mathbb{1}_{X_1 > 0.5}] \\ &= \frac{1}{48} + \frac{6\eta}{48}. \end{aligned}$$

Finally,

$$R(f^{\star}_{\text{surr}}) = \frac{1-p}{48} + p\Big(\frac{1}{48} + \frac{6\eta}{48}\Big) = \frac{1}{48} + \frac{6}{48}\eta p.$$

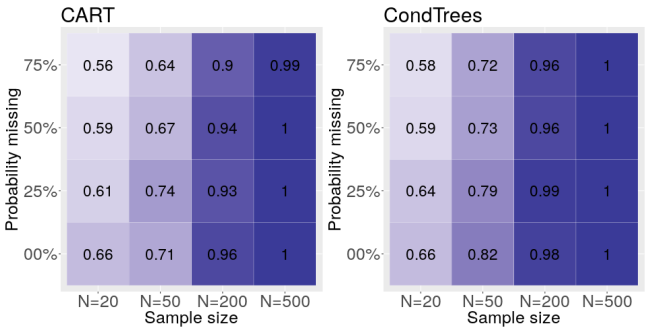
C Miscellaneous

C.1 Variable selection properties of the tree methods with missing values

Decision trees based on the CART criterion (implemented in the R library rpart) and on conditional trees (implemented in the the R library partykit) lead to different ways of selecting splitting variables. We illustrate this behaviour on the simple following model:

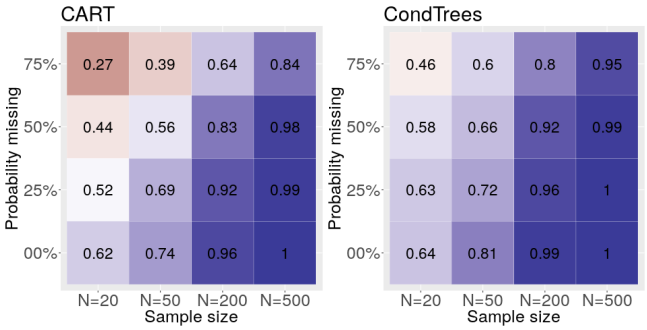
$$\begin{cases} X_1 \perp\!\!\!\perp X_2 \sim \mathcal{N}(0,1) \\ \varepsilon \sim \mathcal{N}(0,1) \\ Y = 0.25X_1 + \varepsilon. \end{cases}$$

We insert MCAR values, either on the first variable or on both variables. Stumps (decision trees of depth one) are fit on 500 Monte-Carlo repetitions. We vary the sample size and the percentage of missing values. Figure 7 show that CART and conditional trees give similar results when there are missing values on both variables. However, Figure 8 shows that CART has a tendency to underselect X_1 when there are missing values only on X_1. For instance, for a sample of size 50 with 75% missing values, CART selects the non-informative variable X_2 more frequently than X_1, while conditional trees keep selecting X_1 more often.



(a) CART (b) Conditional trees

Fig. 7: Frequency of selection of X_1 when there are missing values on X_1 and X_2



(a) CART (b) Conditional trees

Fig. 8: Frequency of selection of X_1 when there are missing values on X_1 only

C.2 Example of EM algorithm

Let us consider a simple case of n observations $(\mathbf{x}_1, \mathbf{x}_2) = (x_{i1}, x_{i2})_{1 \leq i \leq n}$ sampled from the distribution of (X_1, X_2), a bivariate Gaussian distribution with parameters (μ, Σ). We assume that X_2 is subjected to missing values and that only r values are observed. The aim is to get the maximum likelihood estimates of (μ, Σ) from the incomplete data set. The algorithm described below can be straightforwardly extended to the multivariate case. Note that from $(\hat{\mu}, \hat{\Sigma})$, it is then possible to directly estimate the parameters of a linear regression model and thus to perform linear regression with missing values.

We denote by $f_{1,2}(\mathbf{x}_1, \mathbf{x}_2; \mu, \Sigma)$, $f_1(\mathbf{x}_1; \mu_1, \sigma_{11})$ and $f_{2|1}(\mathbf{x}_2|\mathbf{x}_1; \mu, \Sigma)$, respectively, the probability of joint distribution of (X_1, X_2), marginal distribution of X_1 and conditional distribution of $X_2|X_1$. The joint distribution of observed data can be decomposed as:

$$f_{1,2}(\mathbf{x}_1, \mathbf{x}_2; \mu, \Sigma) = \prod_{i=1}^{n} f_1(x_{i1}; \mu_1, \sigma_{11}) \prod_{j=1}^{r} f_{2|1}(x_{j2}|x_{j1}; \mu, \Sigma),$$

and the observed log-likelihood is written (up to an additional constant that does not appear in the maximization and that we therefore drop):

$$\ell(\mu, \Sigma; \mathbf{x}_1, \mathbf{x}_2) = -\frac{n}{2} \log(\sigma_{11}^2) - \frac{1}{2} \sum_{i=1}^{n} \frac{(x_{i1} - \mu_1)^2}{\sigma_{11}^2} - \frac{r}{2} \log\left((\sigma_{22} - \frac{\sigma_{12}^2}{\sigma_{11}})^2 \right)$$
$$- \frac{1}{2} \sum_{i=1}^{r} \frac{\left(x_{i2} - \mu_2 - \frac{\sigma_{12}}{\sigma_{11}}(x_{i1} - \mu_1) \right)^2}{(\sigma_{22} - \frac{\sigma_{12}^2}{\sigma_{11}})^2}$$

We skip the computations and directly give the expression of the closed form maximum likelihood estimates of the mean:

$$\hat{\mu}_1 = n^{-1} \sum_{i=1}^{n} x_{i1}$$

$$\hat{\mu}_2 = \hat{\beta}_{20.1} + \hat{\beta}_{21.1}\hat{\mu}_1,$$

where

$$\hat{\beta}_{21.1} = s_{12}/s_{11}, \quad \hat{\beta}_{20.1} = \bar{x}_2 - \hat{\beta}_{21.1}\bar{x}_1,$$

$$\bar{x}_j = r^{-1} \sum_{i=1}^{r} x_{ij} \quad \text{and} \quad s_{jk} = r^{-1} \sum_{i=1}^{r} (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k), \quad j,k = 1,2.$$

In this simple setting, we have an explicit expression of the maximum likelihood estimator despite missing values. However, this is not always the case but it is possible to use an EM algorithm to get the maximum likelihood estimators in the cases where data are missing.

The EM algorithm consists in maximizing the observed likelihood through successive maximization of the complete likelihood (if we had observed all n realizations of $\mathbf{x}_1$ and $\mathbf{x}_2$). Maximizing the complete likelihood

$$\ell_c(\mu, \Sigma; \mathbf{x}_1, \mathbf{x}_2) = -\frac{n}{2} \log\left(\det(\Sigma)\right) - \frac{1}{2} \sum_{i=1}^{n} (x_{i1} - \mu_1)^T \Sigma^{-1} (x_{i1} - \mu_1)$$

would be straightforward if we had all the observations. However elements of this likelihood are not available. Therefore, we replace them by the conditional expectation given observed data and the parameters of the current iteration. These two steps of computation of the conditional expectation (E-step) and maximization of the completed likelihood (M step) are repeated until convergence. The update formulas for the E and M steps are as follows:

E step: The sufficient statistics of the likelihood are:

$$s_1 = \sum_{i=1}^{n} x_{i1}, \quad s_2 = \sum_{i=1}^{n} x_{i2}, \quad s_{11} = \sum_{i=1}^{n} x_{i1}^2, \quad s_{22} = \sum_{i=1}^{n} x_{i2}^2, \quad s_{12} = \sum_{i=1}^{n} x_{i1}x_{i2}.$$

Since some values of x_2 are not available, we fill in the sufficient statistics with:

$$\mathbb{E}[x_{i2}|x_{i1}; \mu, \Sigma] = \beta_{20.1} + \beta_{21.1}x_{i1}$$

$$\mathbb{E}[x_{i2}^2|x_{i1}; \mu, \Sigma] = (\beta_{20.1} + \beta_{21.1}x_{i1})^2 + \sigma_{22.1}$$

$$\mathbb{E}[x_{i2}x_{i2}|x_{i1}; \mu, \Sigma] = (\beta_{20.1} + \beta_{21.1}x_{i1})x_{i1}.$$

with, $\beta_{21.1} = \sigma_{12}/\sigma_{11}$, $\beta_{20.1} = \mu_2 - \beta_{21.1}\mu_1$, and $\sigma_{22.1} = \sigma_{22} - \sigma_{12}^2/\sigma_{11}$.

M step: The M step consists in computing the maximum likelihood estimates as usual. Given $s_1, s_2, s_{11}, s_{22},$ and s_{12}, update $\hat{\mu}$ and $\hat{\sigma}$ with

$$\hat{\mu}_1 = s_1/n,\ \hat{\mu}_2 = s_2/n,$$

$$\hat{\sigma}_1 = s_{11}/n - \hat{\mu}_1^2,\ \hat{\sigma}_2 = s_{22}/n - \hat{\mu}_2^2,\ \hat{\sigma}_{12} = s_{12}/n - \hat{\mu}_1\hat{\mu}_2$$

Note that s_1, s_{11}, $\hat{\mu}_1$ and $\hat{\sigma}_1$ are constant across iterations since we do not have missing values on $\mathbf{x}_1$.

Remark 4 Note that EM imputes the sufficient statistics and not the data.

D Additional experiments

D.1 Varying the correlation strength and the missing rate

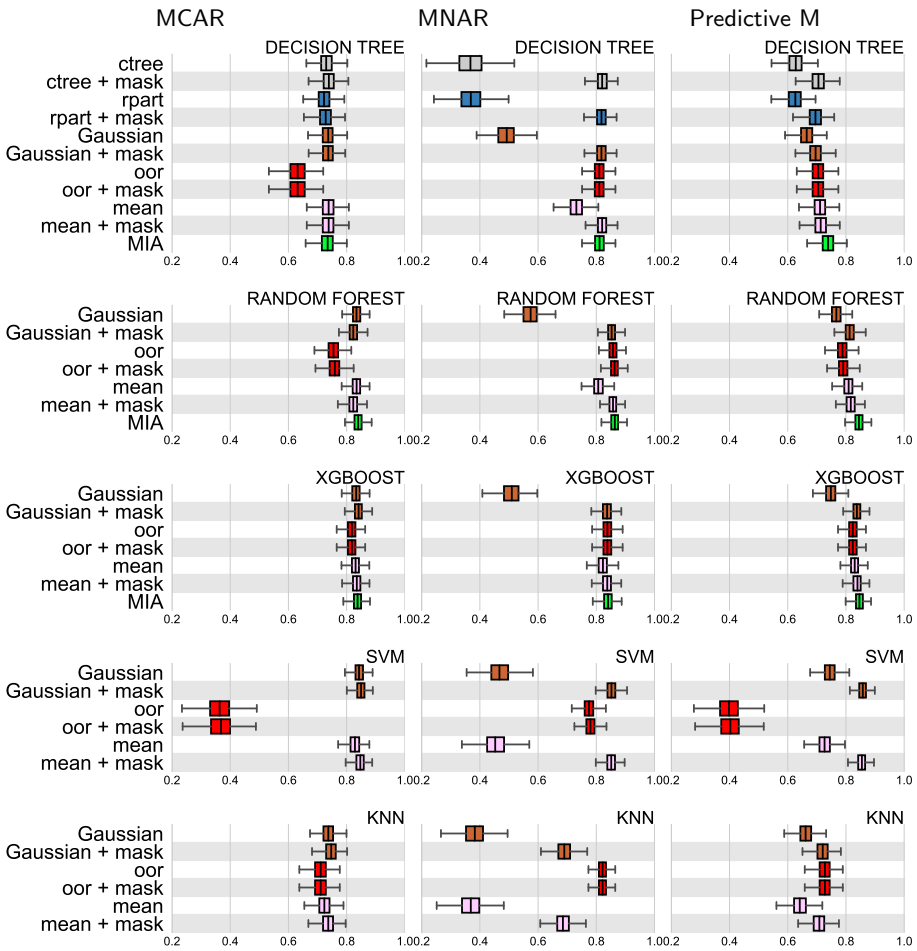


Fig. 9: R^2 **scores on model 1** • Normalized explained variance for different mechanisms with 20% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.2$.

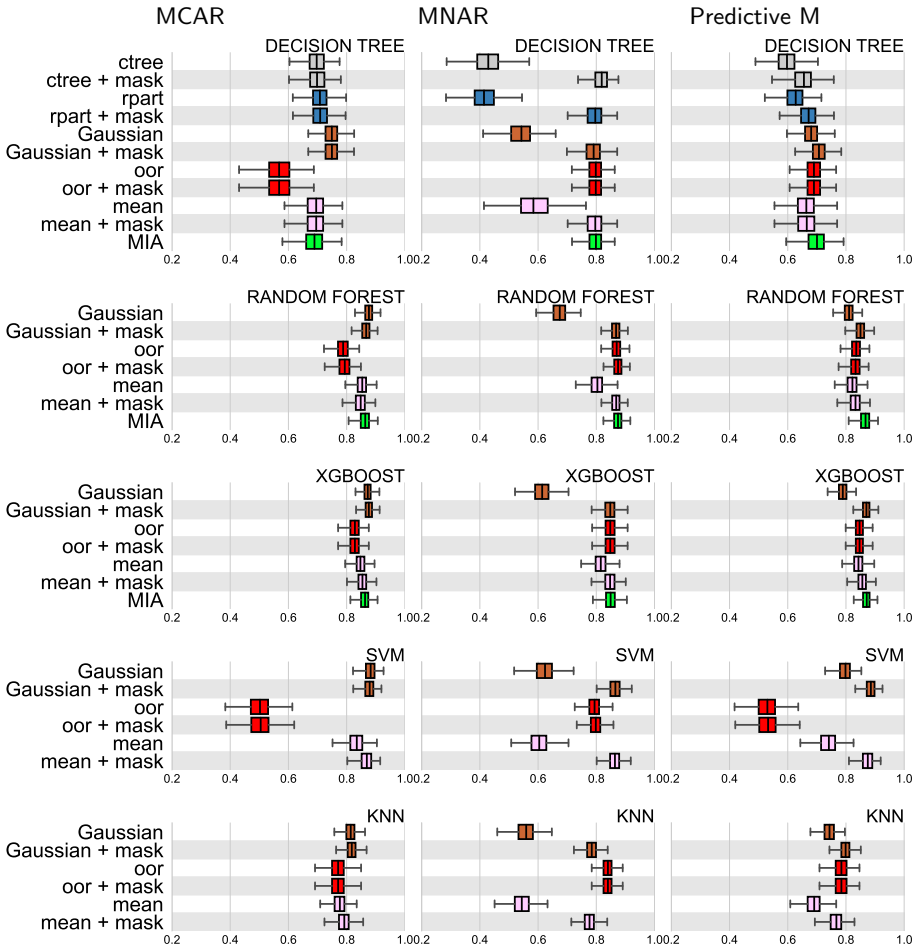


Fig. 10: R^2 **scores on model 1** • Normalized explained variance for different mechanisms with 20% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.5$.

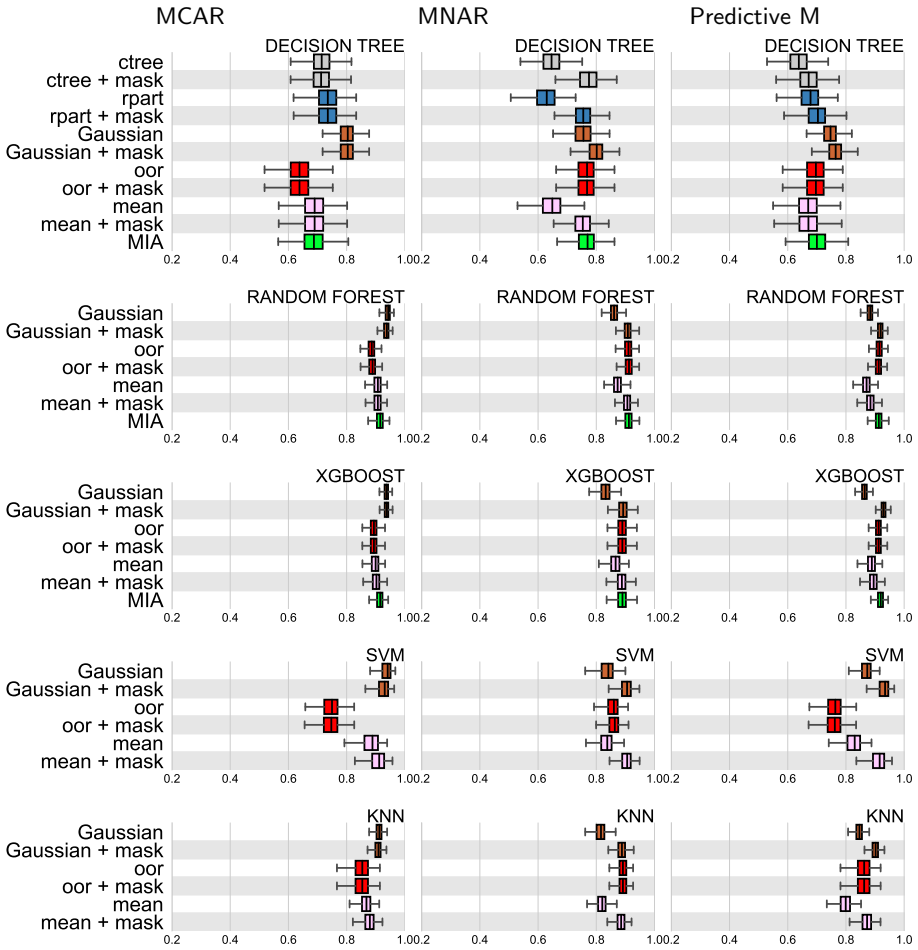


Fig. 11: R^2 **scores on model 1** • Normalized explained variance for different mechanisms with 20% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.8$.

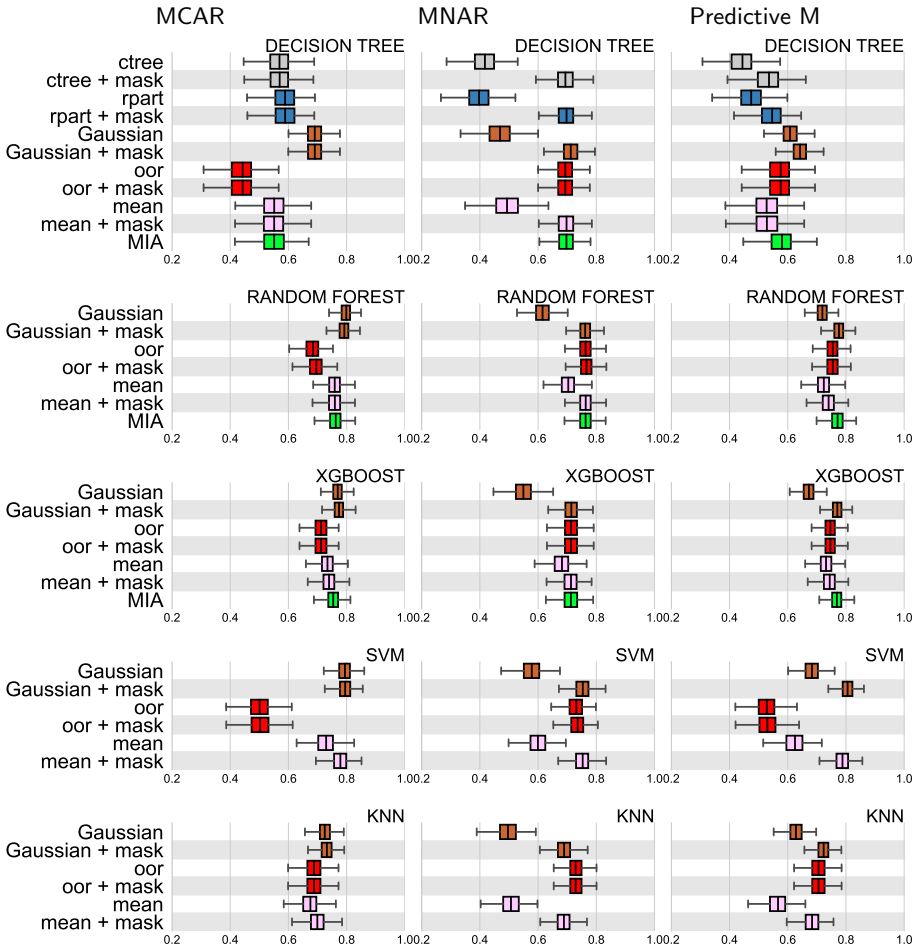


Fig. 12: R^2 **scores on model 1** • Normalized explained variance for different mechanisms with 40% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.5$.

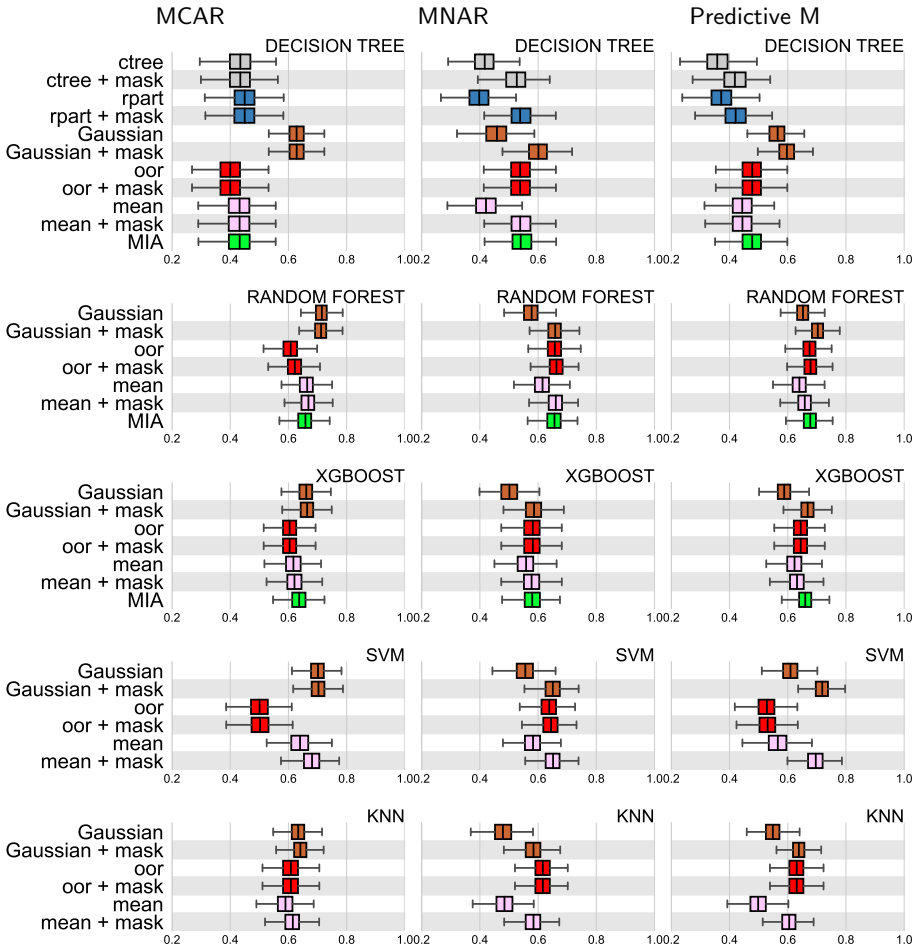


Fig. 13: R^2 **scores on model 1** • Normalized explained variance for different mechanisms with 60% of missing values, $n = 1000$, $d = 9$ and $\rho = 0.5$.

D.2 Statistical tests

Fig. 14: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MCAR model ($\rho = 0.2$, missing rate of 20%)

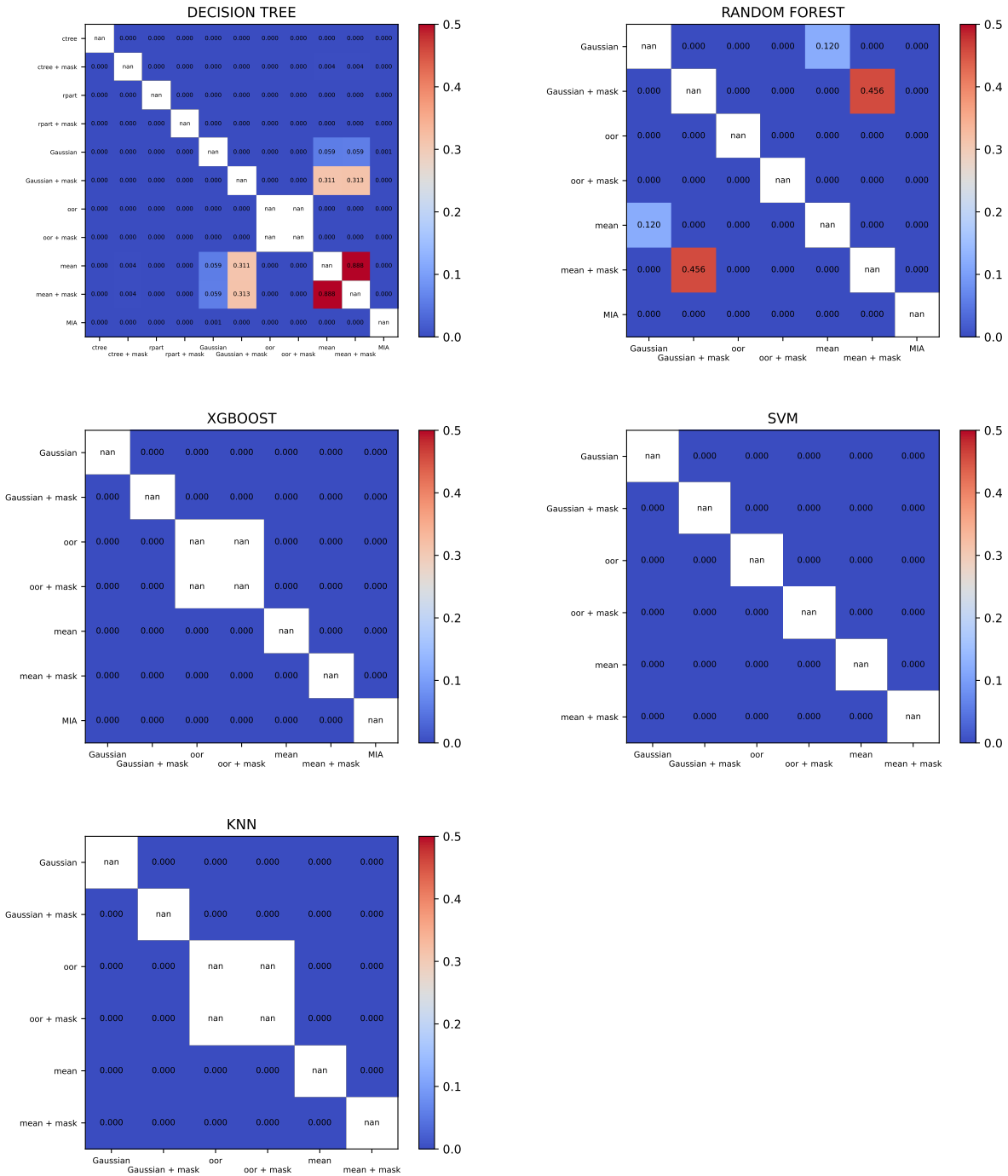


Fig. 15: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MNAR model ($\rho = 0.2$, missing rate of 20%)

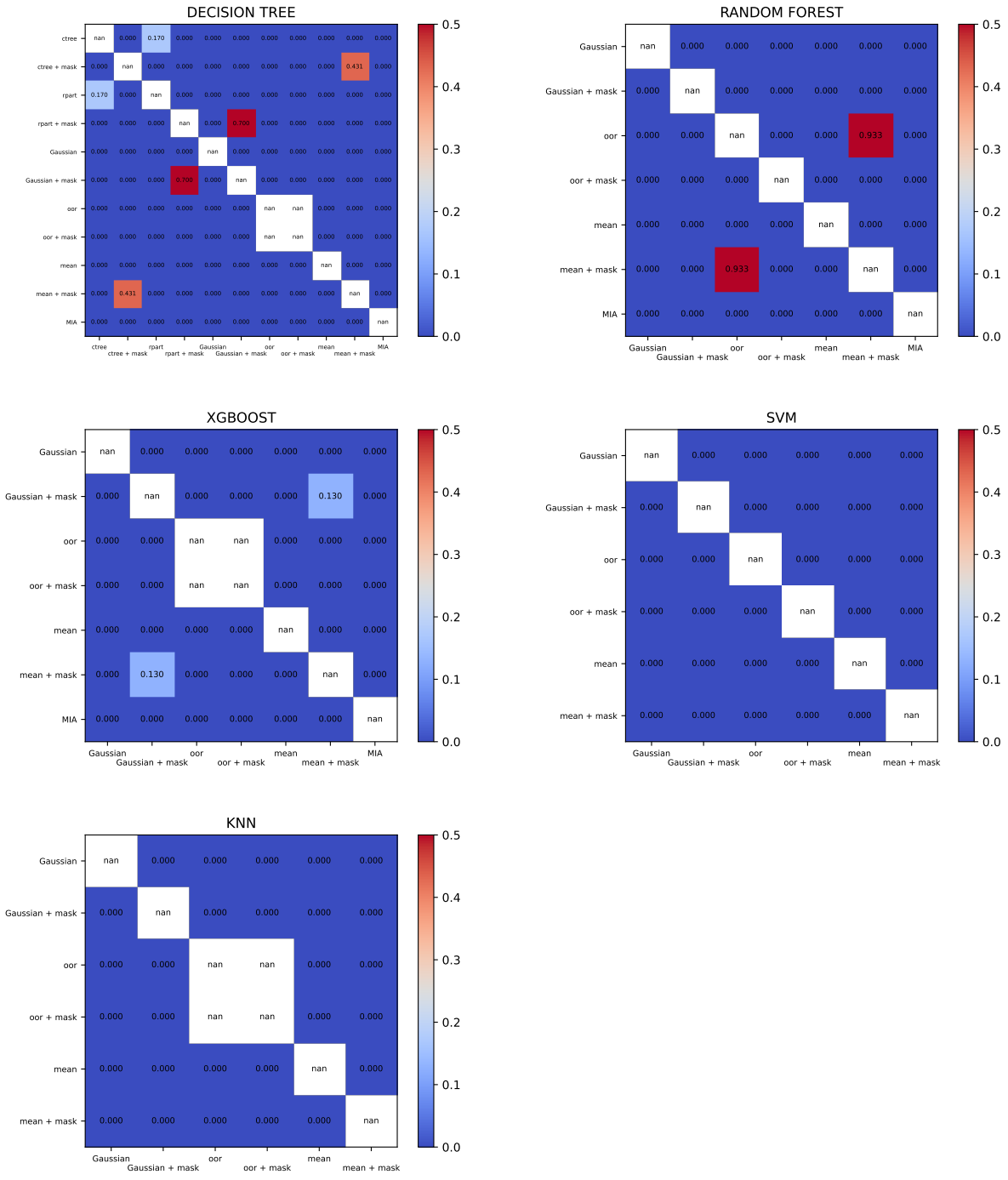


Fig. 16: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a Predictive M model ($\rho = 0.2$, missing rate of 20%)

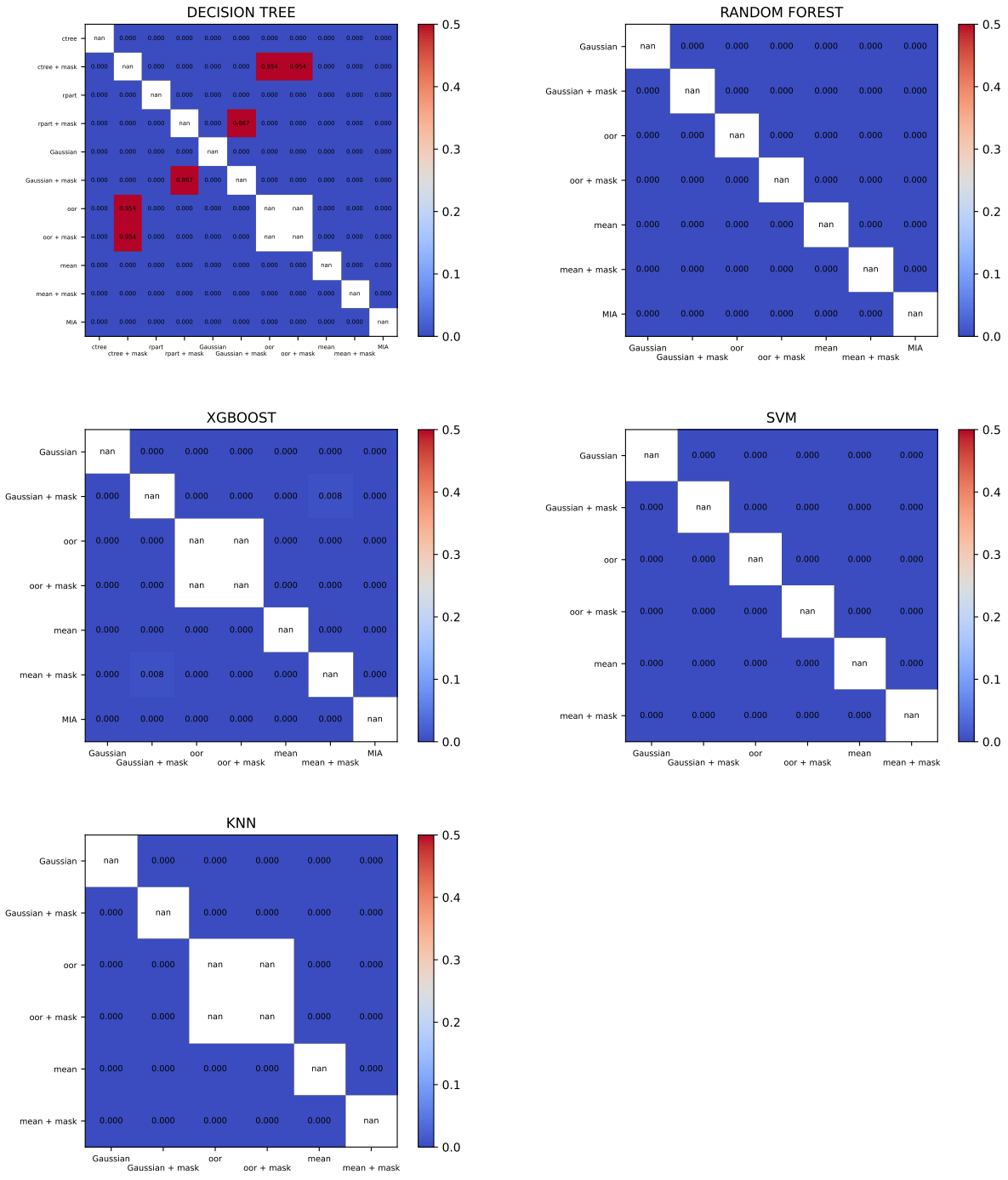


Fig. 17: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MCAR model ($\rho = 0.5$, missing rate of 20%)

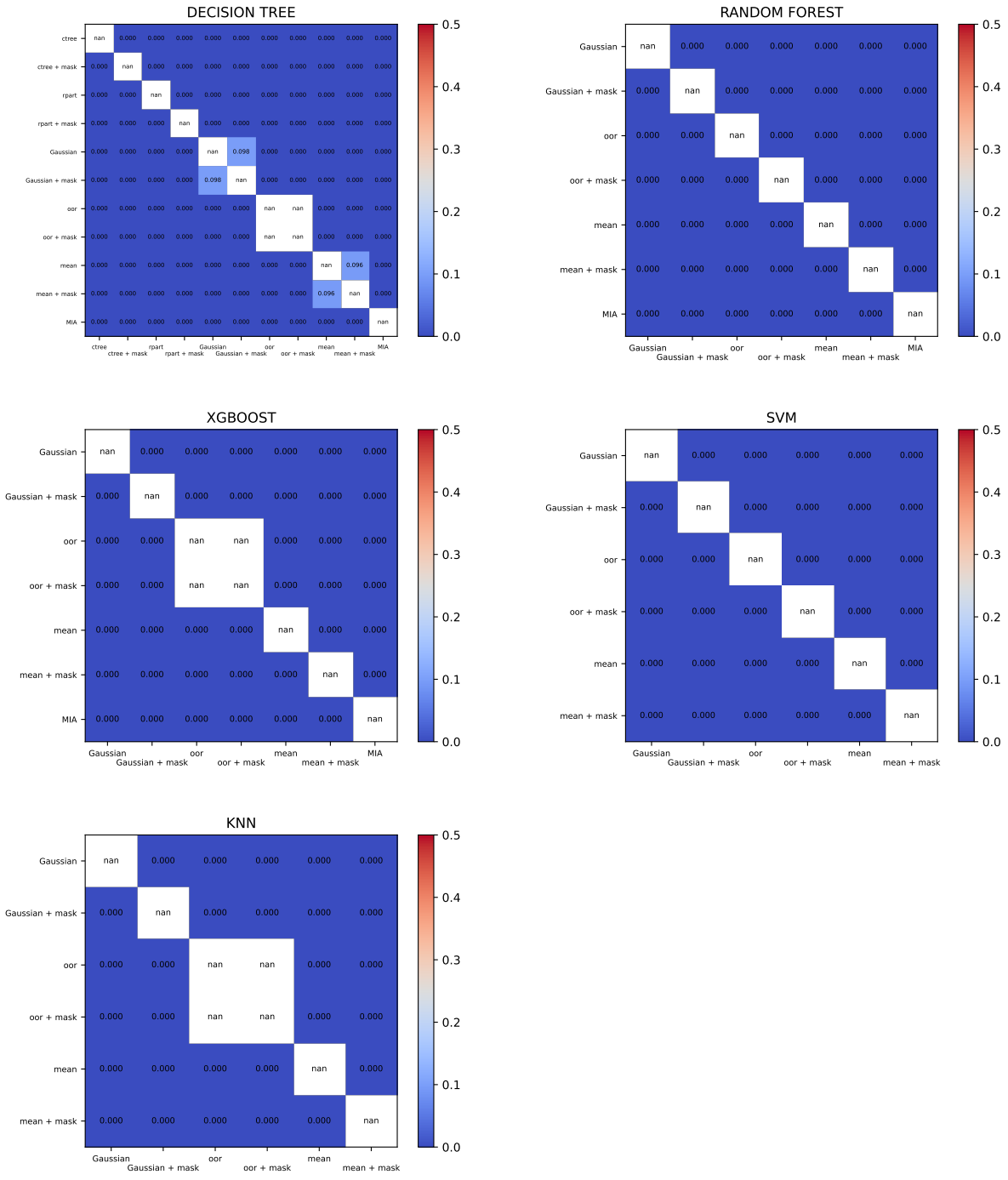


Fig. 18: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MNAR model ($\rho = 0.5$, missing rate of 20%)

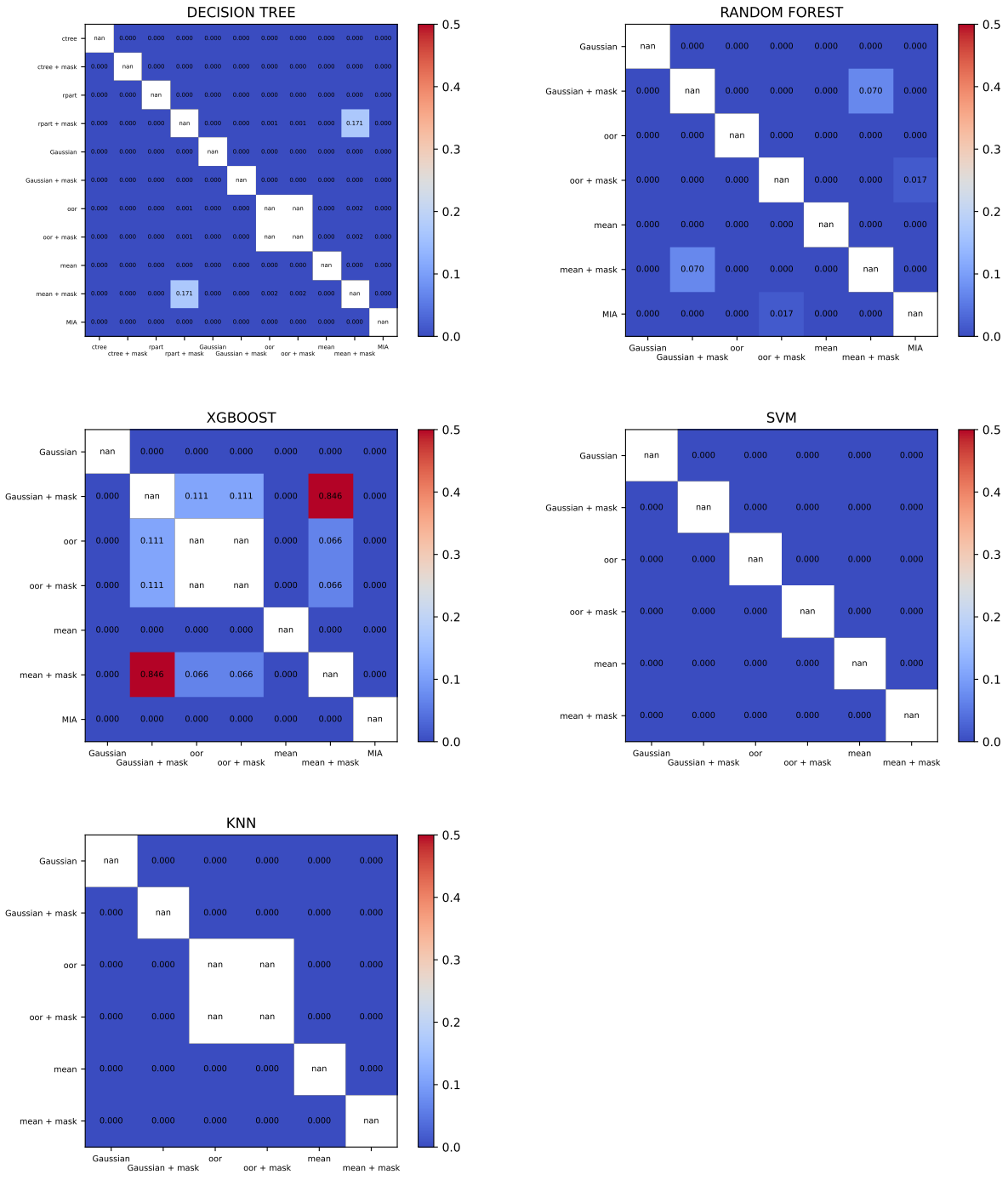


Fig. 19: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a Predictive M model ($\rho = 0.5$, missing rate of 20%)

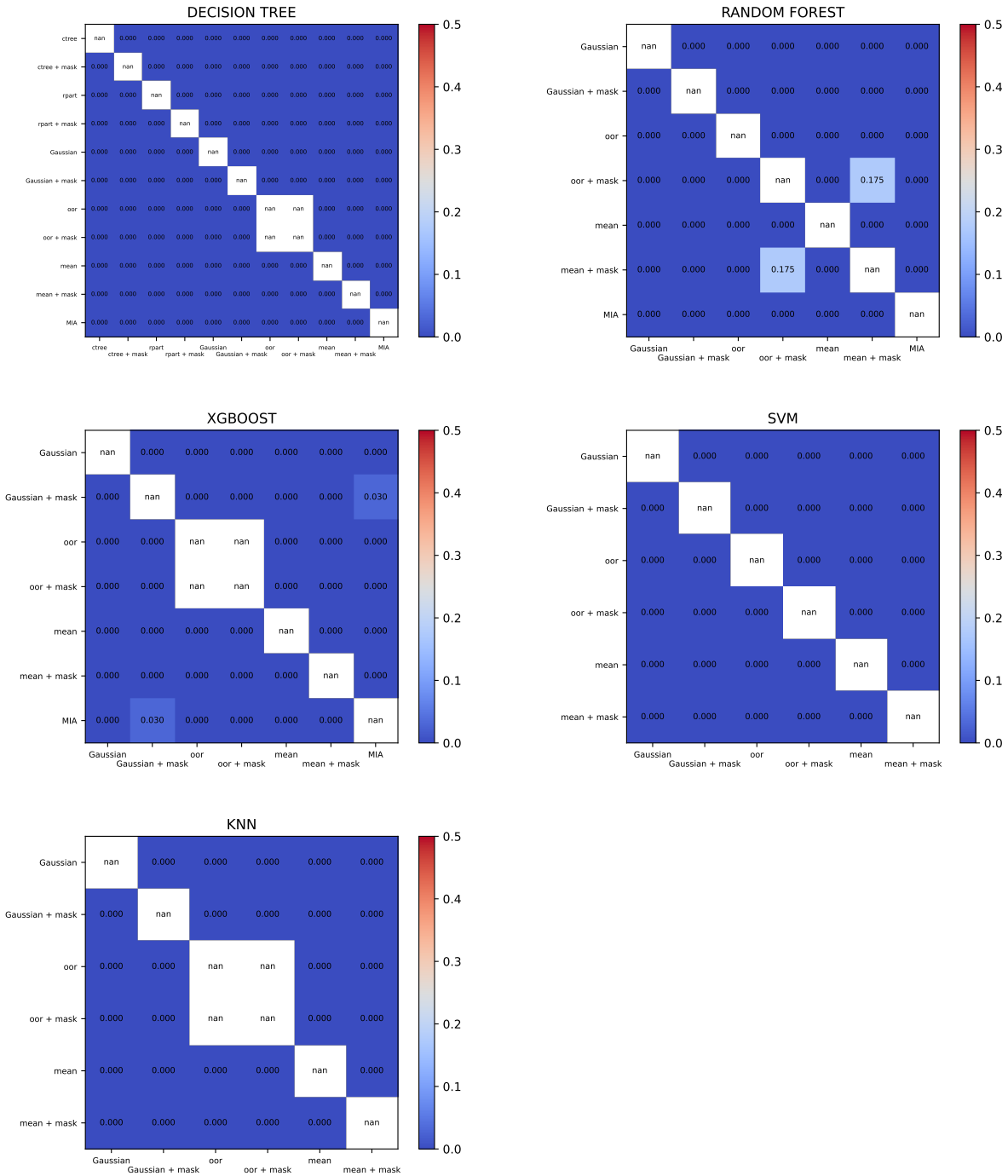


Fig. 20: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MCAR model ($\rho = 0.8$, missing rate of 20%)

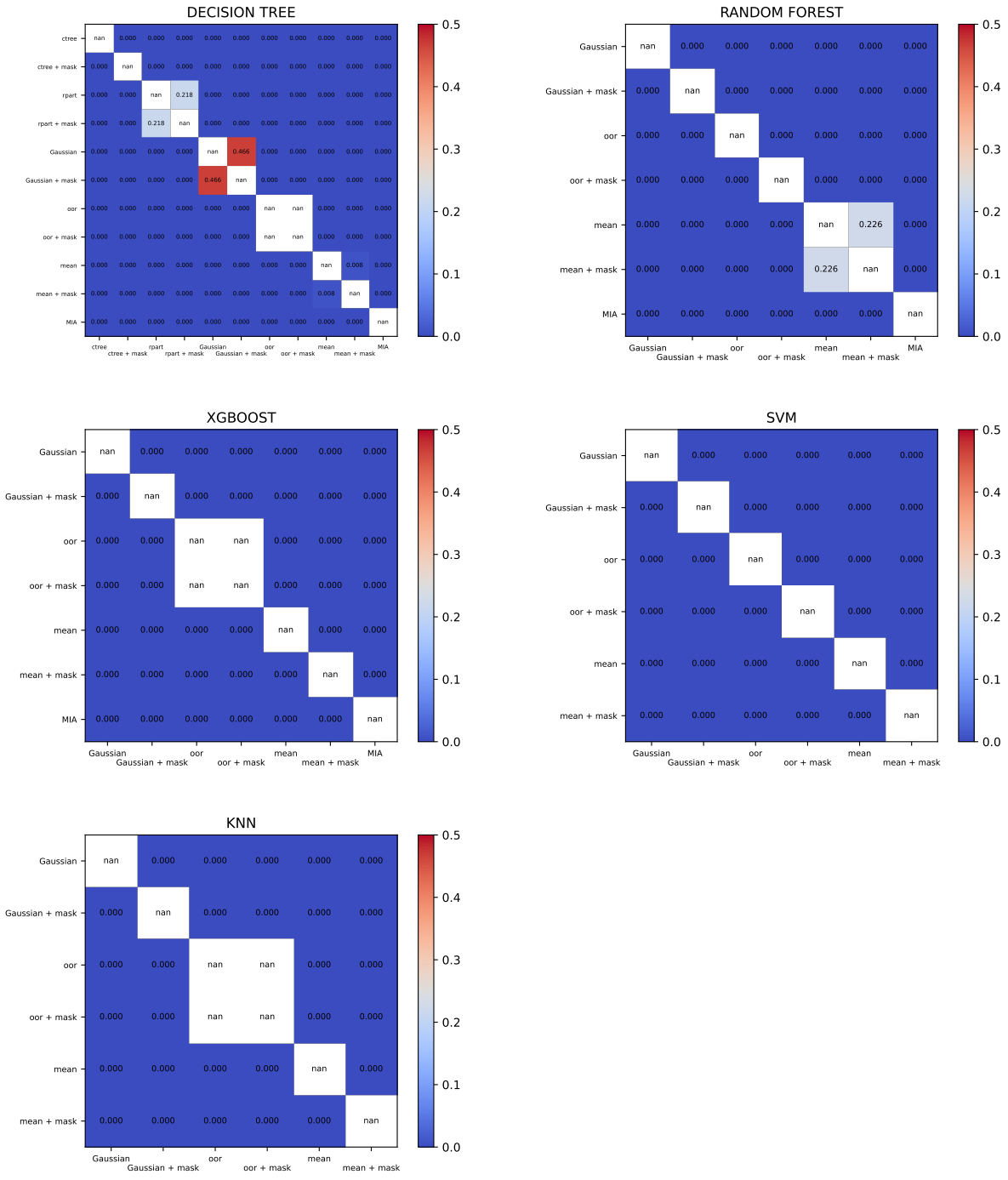


Fig. 21: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MNAR model ($\rho = 0.8$, missing rate of 20%)

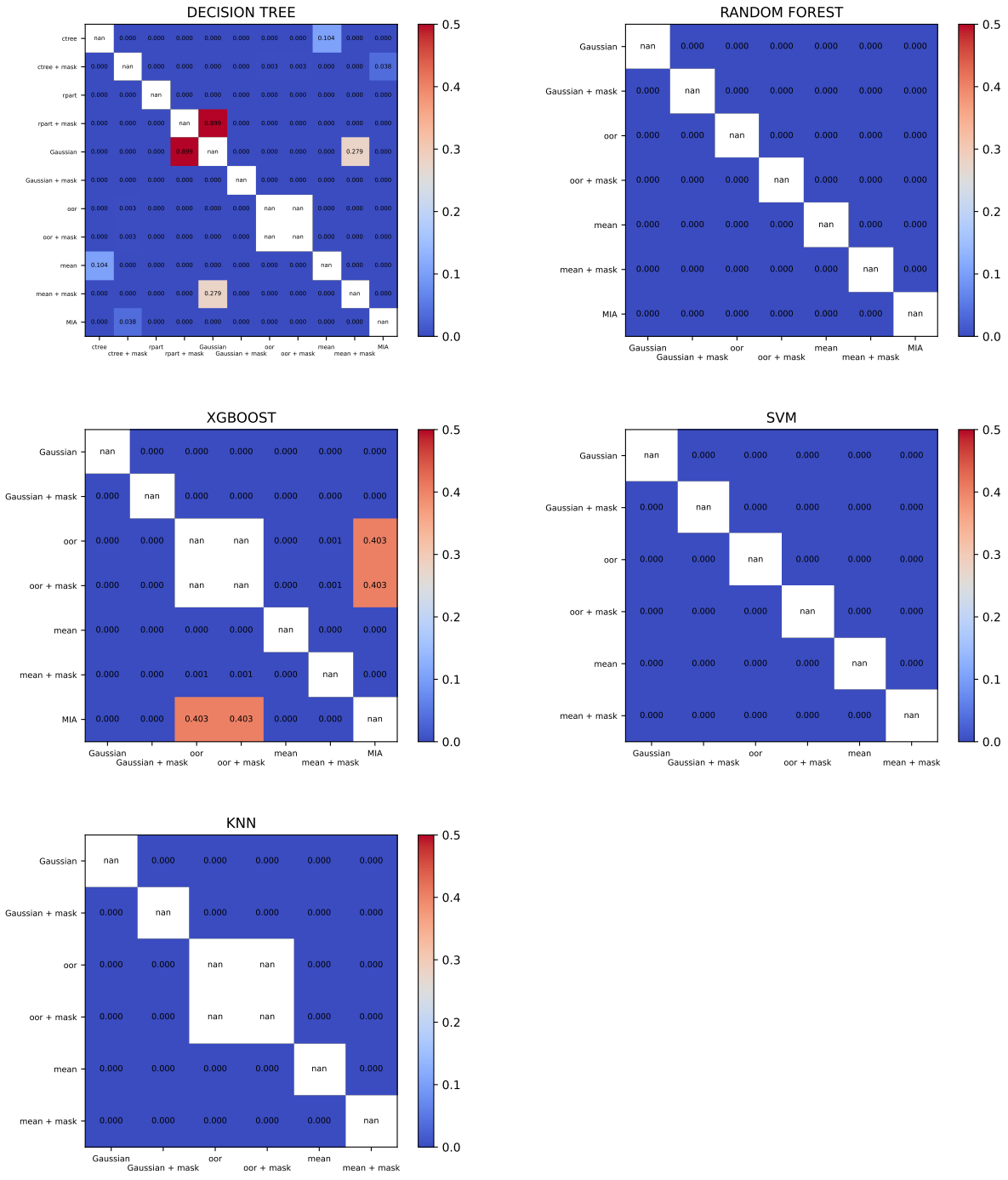


Fig. 22: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a Predictive M model ($\rho = 0.8$, missing rate of 20%)

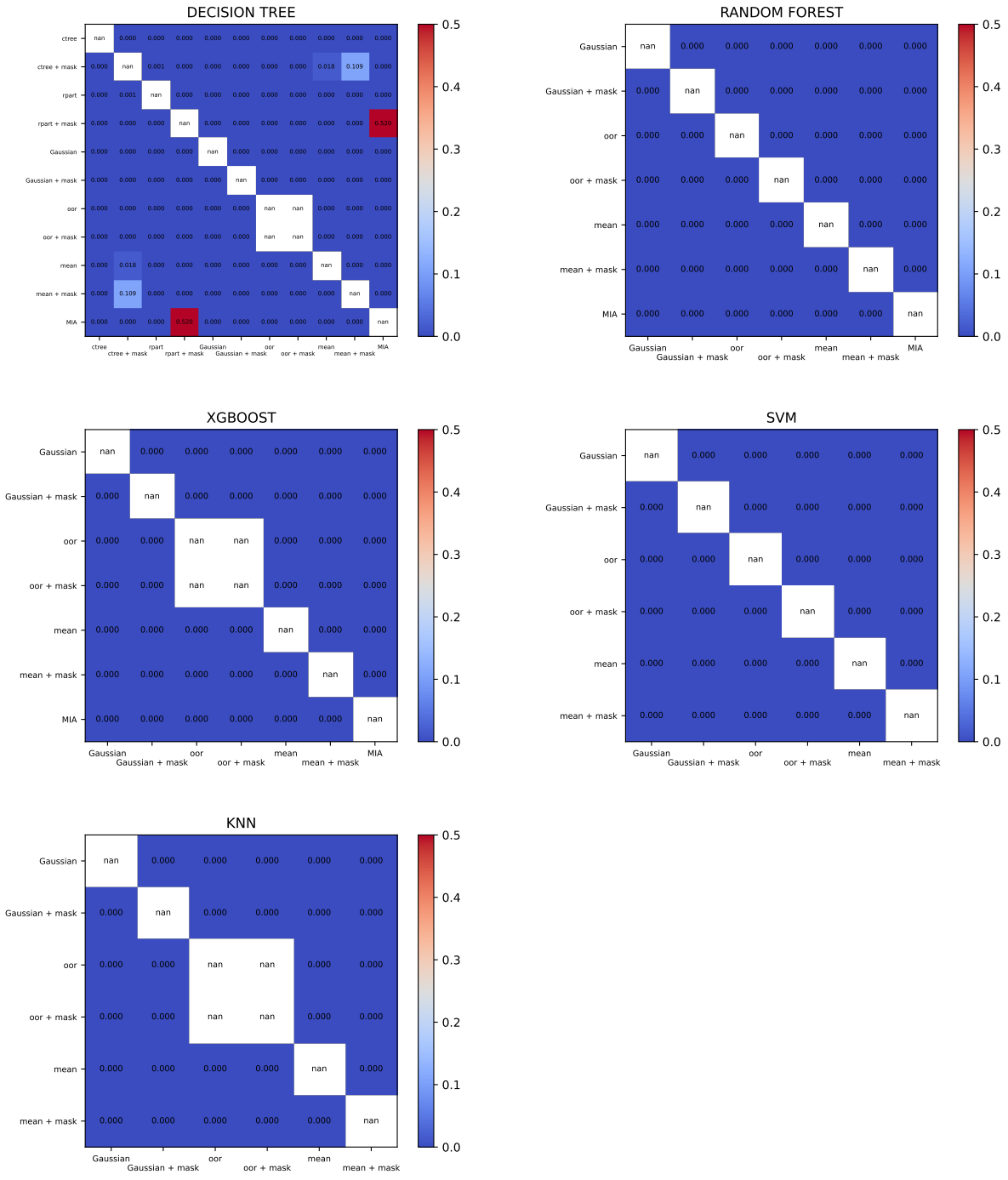


Fig. 23: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MCAR model ($\rho = 0.5$, missing rate of 40%)

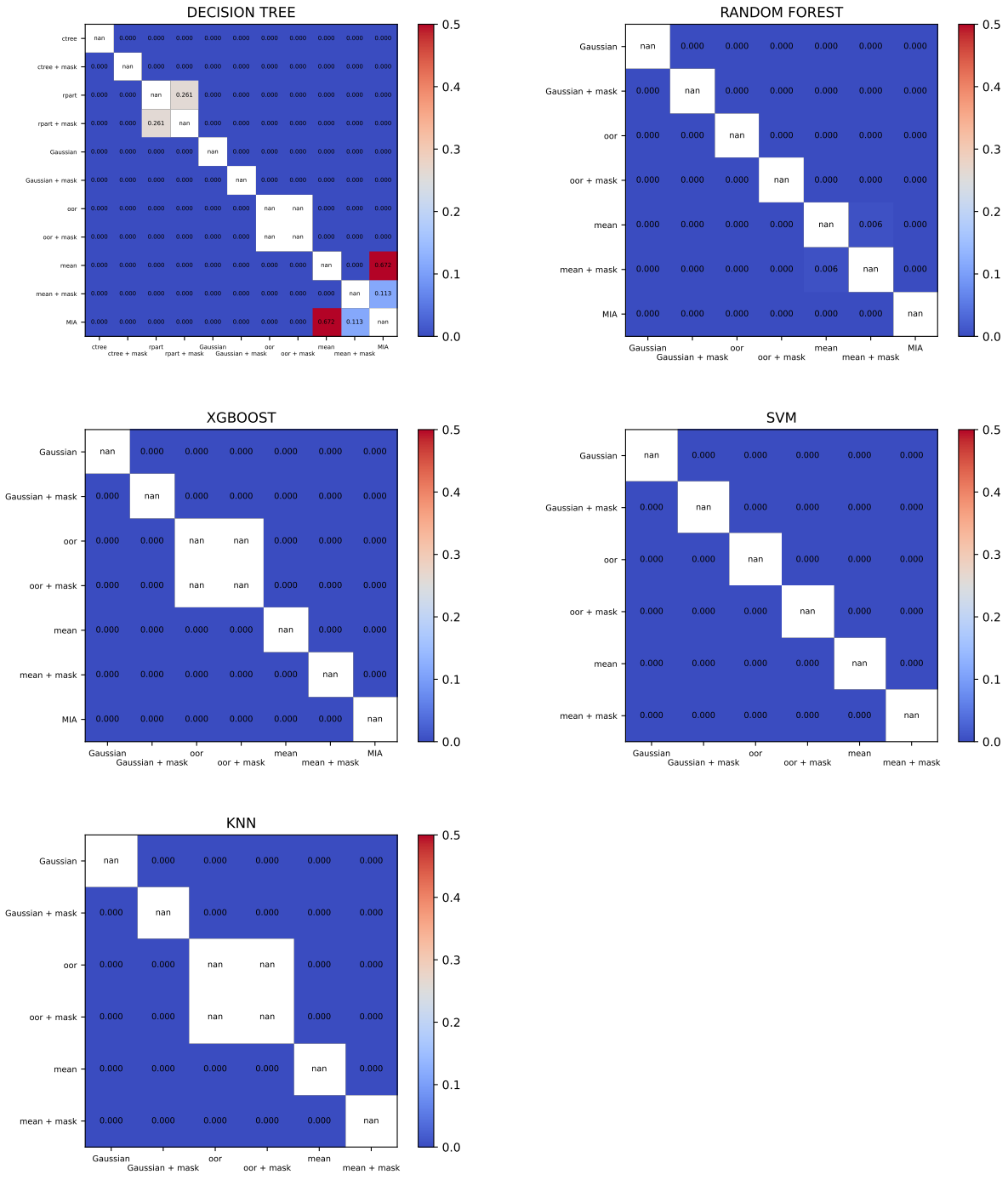


Fig. 24: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MNAR model ($\rho = 0.5$, missing rate of 40%)

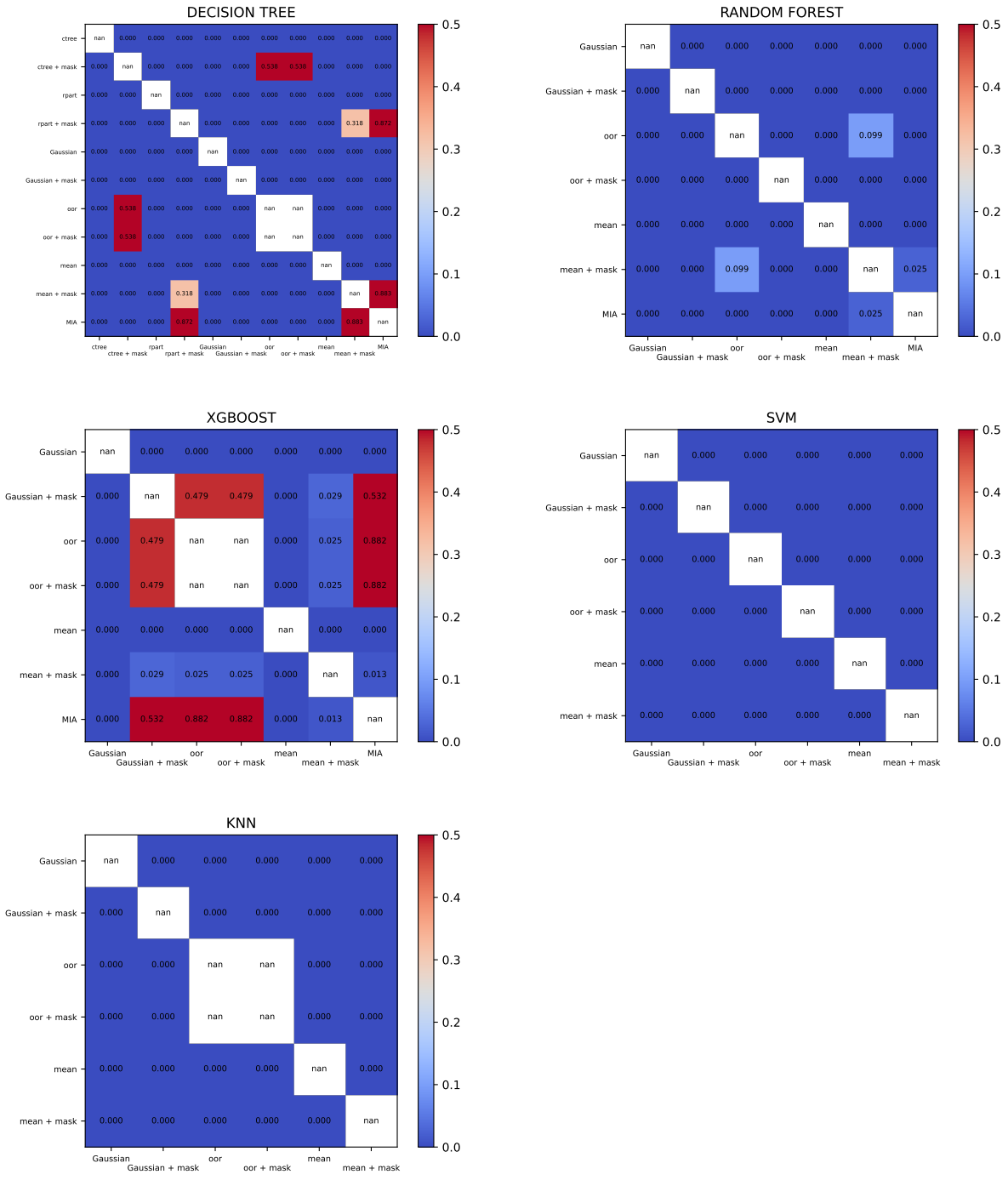


Fig. 25: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a Predictive M model ($\rho = 0.5$, missing rate of 40%)

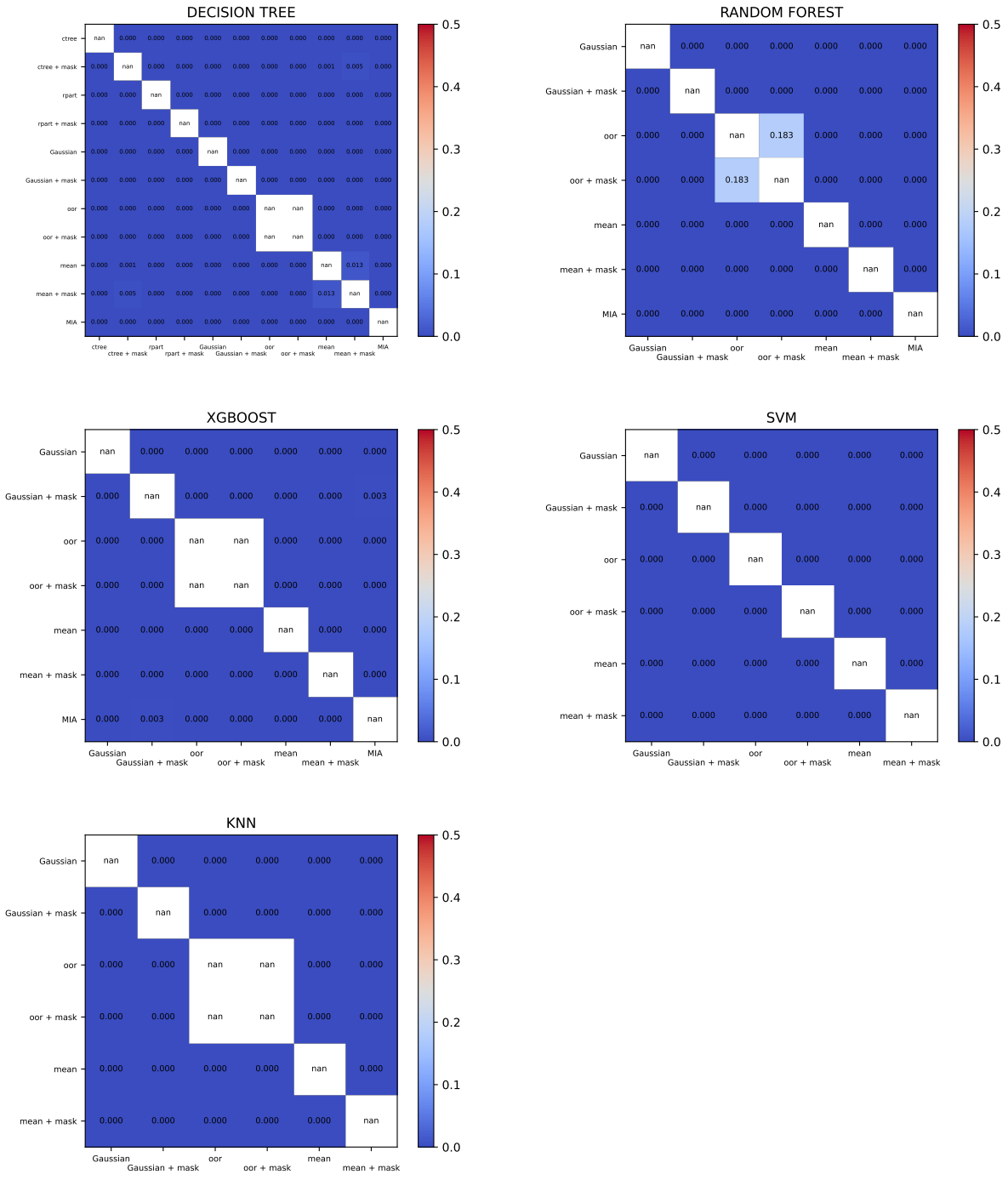


Fig. 26: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MCAR model ($\rho = 0.5$, missing rate of 60%)

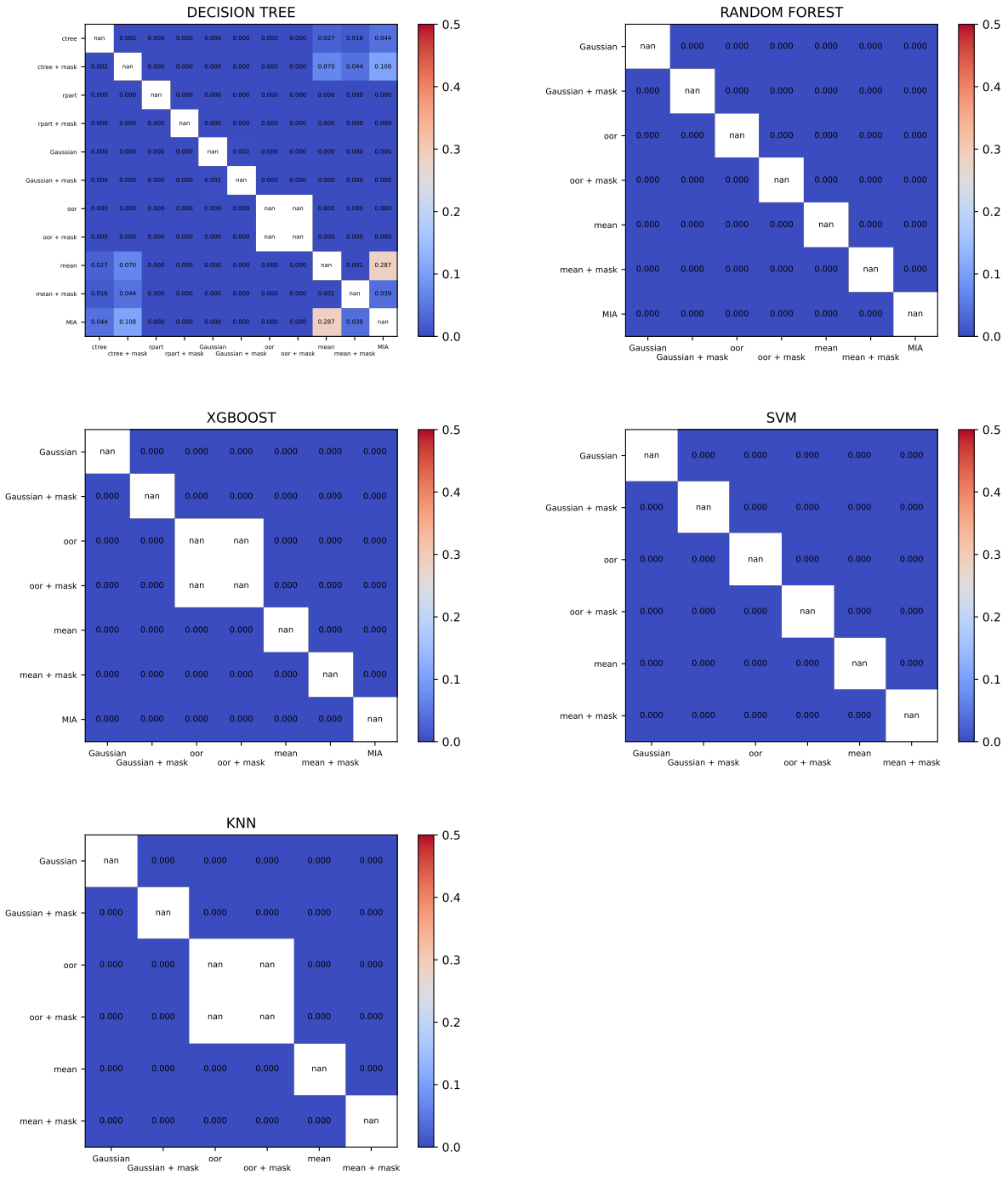


Fig. 27: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a MNAR model ($\rho = 0.5$, missing rate of 60%)

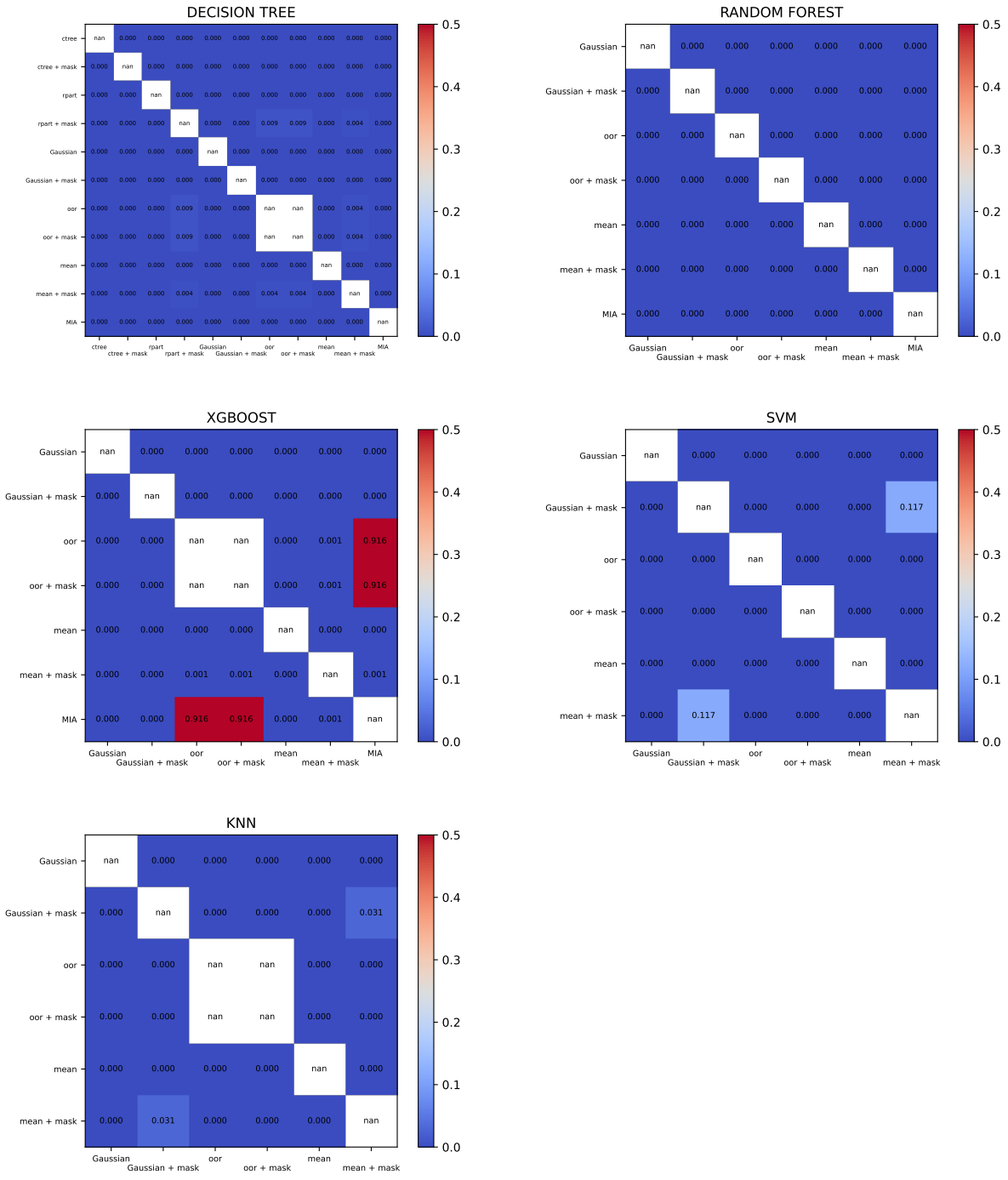


Fig. 28: P-values of paired t-tests for each pair of imputation method and for each learning algorithm in a Predictive M model ($\rho = 0.5$, missing rate of 60%)