UCLouvain

FACULTE DES SCIENCES

Faculté des sciences

# Missing Data

Author: **Aurèle BARTOLOMEO**
Supervisor: **Catherine LEGRAND**
Readers: **Anouar EL GHOUCH, Christian RITTER**
Academic year 2021–2022
Master [120] en science des données, orientation statistique

# Contents

# Introduction

Missing data are present almost everywhere. Handling missing data is one of the first preprocessing steps in real projects although this problem is often set aside in statistical courses and books. The general scope of this work is to give an overview on methods designed to fix this problem, and also to evaluate the impact of missing data on an increasingly popular supervised machine learning classification method : Support Vector Machines. This superposition of statistical and data science methods reflects well the substance of the Master in Data Science and Statistics and it is one of the reasons why this work has been organised in this way. Handling of missing data with classical statistical inference methods has been largely studied. The first and main authors on the subject are Little and Rubin [**5**] and they will often be cited in this work. The subject has also been deeply covered by Molenberghs and Kenward [**1**] in the particular context of clinical studies. On the other hand however, handling missing data in the context of newly emerging machine learning methods is almost not studied in the literature.

We start by an overview of the notations and concepts related to missing data in Chapter 1. The patterns and mechanisms describing and causing missing data are defined formally along with examples to provide a better understanding of these concepts.

Chapters 2 and 3 constitute the statistical part of this work. These chapters provide a deep understanding of several common imputation methods, methods that are designed to replace missing data by an appropriate value. A comprehensive look at Multiple Imputation (a technique that repeatedly replaces missing data by a value and not only one) is given in Chapter 3 while basic methods are described in Chapter 2. In these chapters we cover a broad range of contexts. Indeed, in addition to the analysis of classical data, some examples involve longitudinal data and medical studies.

The second part (Chapter 4) is focused on the impact of missing data on Support Vector Machines (SVMs), and therefore constitutes the data science part of this work. After a recall about SVMs, we present the simulation plan and the obtained results. The link with the first part is that we will use the methods presented in that part to handle missing data. In that way, we will see if some imputation methods are better than others in the context of SVMs. This part is restrained to classical data and does not cover longitudinal data since it is not straightforward to use SVMs with this type of data.

The conclusion gives a recap on the work and discusses improvements and approaches that could be useful to tackle the problem of missing data related to machine learning methods.

**Acknowledgment**

We would like to thank Catherine Legrand, the promoter of this work, for the help and advice given all along the elaboration of this thesis.

CHAPTER 1

# Overview

Before diving into imputations methods, we need to give a formal description of the missing data framework.

## 1. Notations

In the following, we will use a set of notations that are a combination of the notations used by Little and Rubin [**5**] and Molenberghs and Kenward [**1**]. The data is described in terms of units $i = 1,...,N$ and measurements $Y_{ij}$, with $j = 1,...,l_i$ and $l_i$ is the number of measurements made for unit $i$. $Y$ represents all the sample data, and can include also covariate measurements (unlike Little and Rubin [**5**] that work also with a matrix $X$ representing the completely observed covariates). In longitudinal studies without covariates, $i$ will be the subject and $j$ the measurement occasion. In a multivariate case, $j$ will be the index for a particular outcome variable. This work will focus mainly on these two previous settings : either longitudinal data without covariates or multivariate non-longitudinal data. In most of the cases, the collected data $Y = (y_{ij})$ is a rectangular matrix of dimensions $N \times l$ and the values for a given subject $i$ can be wrapped in a vector $y_i = (y_{i1},...,y_{il})$. However, in case of unbalanced design, $Y$ is no longer a rectangular matrix. We now introduce the missing data indicator matrix $M = (m_{ij})$ with the same dimensions as $Y$. Its elements are defined as follows:

$$m_{ij} = \begin{cases} 1 & \text{if } y_{ij} \text{ is observed,} \\ 0 & \text{otherwise.} \end{cases}$$

In the following, we will always suppose that the matrix $M$ is completely known. This is the case most of the time in practice. However, some statistical studies with specific design do not come with a completely known matrix $M$. For example, a common practice in questionnaires is to ask the respondents to tick some propositions in a list (such as a list of medications where the respondents need to tick the ones they are taking). Then, if a proposition is ticked, we know that the data is not missing and we also know its value (interpreted as "yes"). But if a proposition is not ticked, we can not differentiate a "no"

from a missing value. However, this can be easily solved by adapting the design of the questionnaire and asking the respondent to answer "yes" or "no" for each proposition. In this way missing values are clearly identifiable (absence of both "yes" and "no"). Since it is more a design problem, it will not be further covered and $M$ is considered complete.

$M$ defines the pattern of missing data (ex: dropout in longitudinal study). Missing data pattern is not to be confused with missing data mechanism. These concepts are covered in section 2 and 3, respectively. Using these notations, we can divide the $Y_i$ vector into two disjoint subsets:

- The observed data $Y_i^o = \{Y_{ij} | M_{ij} = 1\}$
- The missing data $Y_i^m = \{Y_{ij} | M_{ij} = 0\}$

Full data $(Y, M)$ is the object of interest, with density that can be decomposed as

$$f(Y, M|\theta, \phi) = f(Y|\theta) f(M|Y, \phi)$$

where $f(Y|\theta)$ is the density of $Y$ with respect to parameters $\theta$ and $f(M|Y, \phi)$ is the density of a Bernouilli distribution modeling the missing data process and $\phi$ its parameter.

## 2. Missing Data Patterns

In a nutshell, two main patterns of missing data can be distinguished : *monotone* and *non-monotone*. Little and Rubin [**5**] describe even more patterns than these two (such as *file matching*, for example) but they are not so interesting for statistical purposes. The definition of a monotone pattern is given by Molenberghs and Kenward [**1**] as follows. A pattern is called monotone if there exists a permutation of the measurements occasions such that a measurement earlier in the permuted sequence is observed for at least those subjects who are observed at later moments. The permutation has to be the same for all observations. A dropout satisfies this definition. Indeed, even without permuting the measurements occasions, a value at a given time measurement $j$ has to be observed if values at time measurements greater than $j$ are observed. If it was not observed despite an observation at later time, it would not be a dropout, by definition (but rather an intermittent missingness). Note that this definition only makes sense for balanced design (otherwise the permutations between measurement occasions are not well defined).

In the case of a dropout pattern, an often simpler description of the missing data can be done using a single scalar $D_i$ instead of the vector $M_i$ (the $i^{th}$ line of the matrix $M$). $D_i$

is defined to be the occasion of the first missing measurement

$$D_i = 1 + \sum_{j=1}^{l_i} m_{ij}$$

and covers exactly the same information as $M_i$. The missing data pattern influences the quantity of information that can be transferred between variables (and the precision of imputations).

Another distinction that is often made in practice is between univariate missing data and multivariate missing data. The first term refers to data for which only one variable contains missing data while the second refers to data where at least two variables contains missing data. This distinction is also very important since the treatment of multivariate missing data comes with more problems. For example it is more complex to predict a missing value of a given variable based on other variables if those variables also contain missing values. This problem will be addressed in Chapter 2, Section 4 and 5 (Regression Imputation and Stochastic Regression Imputation) as well as in Chapter 3 (Multiple Imputation).

## 3. Missing Data Mechanisms

The formal definitions of the missing data mechanisms presented in this section were first introduced by Rubin in 1976 [**2**]. The aim is to characterize how the density of the missing data process $f(M|Y,\phi)$ can be expressed (and sometimes simplified). This is a crucial concept since the reliability of imputed values using a given imputation method will depend on the missing data mechanism.

**3.1. MCAR-Missing Completely at Random.** Data is said to be MCAR if missingness is independent of both unobserved and observed data [**1**]. Formally, this means that

$$f(M|Y,\phi) = f(M|\phi)$$

Working with MCAR data offers one big advantage : taking only the observed values $\{Y_i^o, i = 1,...,N\}$ into account does not bias the analysis (this will be developed in Chapter 2, Section 1). However, this is a strong assumption that is often unrealistic. We now give a theoretical example of MCAR data.

EXAMPLE 1.1. If $P(M_{ij} = 1) = 0.2\ \forall i = 1,...,N, j = 1,...,l$, data are MCAR.

The following example highlights the fact that MCAR data can often be attributed to "bad luck" [**6**].

EXAMPLE 1.2. Suppose a longitudinal medical study where the weight of the patients has to be taken every week. Then, several "bad luck" phenomena can be the cause of missing data. For example, the weighing scale could run out of batteries, or the patient could not come one week because he had extra work to do at his job, etc. In this case data are MCAR.

**3.2. MAR-Missing at Random.** Data are said to be MAR if missingness is independent of the unobserved data [**1**]. This means that missingness can depend on observed data. We than have

$$f(M|Y,\phi) = f(M|Y_i^o,\phi)$$

In this setting, the probability of being missing can depend on any observed data, including covariates. Here follow two different examples of MAR data.

EXAMPLE 1.3. If $P(M_{ij}=0) = \frac{exp(\alpha+\beta Y_{i(j-1)})}{1+exp(\alpha+\beta Y_{i(j-1)})}$ with $\alpha \in \mathbb{R}, \beta \in \mathbb{R}\backslash\{0\}$, data are MAR.

EXAMPLE 1.4 (longitudinal multivariate study). One of the most cited examples of MAR data concerns an orthodontical study on teeth growth among children between 8 and 14 years old. It has been analysed in detail by Little and Rubin [**5**], Molenberghs and Lesaffre [**4**] and Van Buuren [**6**]. In this example they found that the probability that a value is missing is different among boys and girls.

MAR hypothesis is often more general and more realistic than supposing data are MCAR. If none of these hypothesis hold, we say that data are MNAR.

**3.3. MNAR-Missing Not at Random.** In this case the probability that a particular value is missing depends also of the value itself. It is the most general case and the expression of the density $f(M|Y,\phi)$ can not be simplified.

EXAMPLE 1.5. In a longitudinal medical study, data are often considered MNAR because a dropout could be caused by a degraded health of the patient.

EXAMPLE 1.6. If $P(M_{ij}=0) = \frac{Y_{ij}}{Y_{ij}+1}$, data is MNAR.

Treatment of MNAR data is the most complex. There are two main strategies to deal with MNAR data. The first one is to make the data "more MAR", using the expression of Van Buuren [**6**]. This strategy consists in identifying additional information that explains

differences in the probability to be missing. For example, by integrating new variables to the study. The second strategy, described in details by Molenberghs and Kenward [**1**], is called sensitivity analysis. The goal of the sensitivity analysis is to explore the result of the analysis under alternative scenarios for the missing data. Sensitivity analysis could itself be the main subject of a thesis and will not be further treated here.

### 3.4. Remarks.

- As explained by Molenberghs and Lesaffre [**4**], the previous definitions are conditional on including the correct set of covariates into the model. By *correct*, we refer to a set of covariates that includes every variable that is related to the missing data mechanism. For example, in a MAR mechanism where the probability of being missing is different with respect to a covariate such as the sex of a person, we expect that this variable is included in the study. Of course, this is impossible to know a priori, but it can be used as a guideline during the conception of a study.
- It is often difficult to justify that data is MAR. In particular, observed data alone can not distinguish between MAR and MNAR [**1**]. In general, MNAR approaches rely on untestable assumptions about the nature of the relation between the missing data value and the missing data mechanism. For example, in a clinical study, one could suppose that a missing data for a patient is due to a bad health condition. This means that it is really important to avoid missing data as much as possible, in order not to have to make such assumptions.

CHAPTER 2

# Basic imputation methods

> "The idea of imputation is both seductive and dangerous. It is seductive because it can lull the user into the pleasurable state of believing that the data are complete after all, and it is dangerous because it lumps together situations where the problem is sufficiently minor that it can be legitimately handled in this way and situations where standard estimators applied to the real and imputed data have substantial biases."
>
> Dempster and Rubin [**15**]

## 1. CCA-Complete Case Analysis

The first idea could be to keep only the units $i$ that do not present missing values (i.e $\{i|M_{ij} = 1 \forall j = 1,...,n_i\}$). It is one of the most common methods, CCA being by the way the default method in several statistical softwares. When data are MCAR, estimators (such as mean, variance, regression coefficients, ...) calculated on complete cases are unbiased. This is actually the only case where CCA has a good property. Indeed, estimators are biased if data are not MCAR. For example, Little and Rubin [**5**] showed that the bias of the mean estimator increases if the difference between the mean of observed data and the mean of missing data increases. Moreover, even when data are MCAR, another problem arises : the loss of a huge amount of data, leading to less precision in inference.

Let's illustrate the loss of information that one could be facing when applying CCA on an example by Molenberghs and Kenward [**1**]. We assume that we have collected data for $N$ individuals and $l$ measurements occasions. If data are MCAR and a value is missing with probability $\pi$, we have for each individual $i$ :

$$P(\text{at least one value is missing}) = 1 - (1-\pi)^l$$

We also obtain a formula for the proportion of values kept in the final analysis:

$$\frac{Nl(1-\pi)^l}{Nl(1-\pi)} = (1-\pi)^{l-1}$$

If for example $N = 100$, $l = 20$ and $\pi = 10\%$, we will only keep 13% of the observed values. Thus, it is extremely inefficient.

EXAMPLE 2.1 (Efficiency of CCA for the mean of normal data). Let's write down explicitly what would be the loss of precision on the estimator of the mean on a normally distributed random variable $Y$. The loss of precision is measured by the relative variance increase of the mean estimator. We have :

$$Var(\bar{Y}_{CC}) = Var(\bar{Y}_{NM})(1+\Delta_{CC})$$

where

- $\bar{Y}_{CC}$ is the mean calculated on the complete cases
- $\bar{Y}_{NM}$ is the mean if there was no missing values
- $\Delta_{CC}$ is the relative increase of variance due to the data diminution

Assume that data are MCAR and that $r$ out of $N$ cases are complete, while $(N-r)$ cases have $Y$ observed but a missing covariate value. Since $Y$ is normally distributed (i.e. $Y \sim N(\mu, \sigma^2)$), $\bar{Y} \sim N\{\mu, \frac{\sigma^2}{N}\}$. We have :

- $Var(\bar{Y}_{CC}) = \frac{\sigma^2}{r}$
- $Var(\bar{Y}_{NM}) = \frac{\sigma^2}{N}$

Thus,

$$\Delta_{CC} = \frac{N-r}{r}$$

and excluding half of the values in the analysis would lead to doubling the variance of the estimator of the mean. Furthermore, it is interesting to observe that only values of a covariate of $Y$ were missing, but all values of $Y$ were actually available, but CCA discards these examples anyway. □

EXAMPLE 2.2 (Bias of CCA inference for means). We start by rewriting the inferred mean $\bar{Y}$ :

$$\bar{Y} = \pi_{CC}\bar{Y}_{CC} + (1-\pi_{CC})\bar{Y}_{IC}$$

where :

- $\pi_{CC}$ is the proportion of observations kept in the CCA,
- $\bar{Y}_{CC}$ is the mean computed on complete cases,

- $\bar{Y}_{IC}$ is the mean computed on incomplete cases.

We rearrange the terms and obtain the following equation :

$$\bar{Y}_{CC} - \bar{Y} = (1 - \pi_{CC})(\bar{Y}_{CC} - \bar{Y}_{IC}).$$

We see that the bias of $\bar{Y}_{CC}$ increases with the proportion of discarded observations and with the difference between the mean over complete cases and over incomplete cases. So unless we are in an MCAR case (because then $\bar{Y}_{CC} = \bar{Y}_{IC}$), we will have a bias. A priori, it is not even possible to fix a bound to this bias since it depends on unobserved values. □

As a remark, it is also interesting to note that Complete Case Analysis violates the ITT (intention to treat) principle, which concerns essentially medical studies. In a clinical trial, the ITT principle states that all patients should be included in the final evaluation of the statistical analysis. It is generally recommended as a good practice and it is clear that CCA violates this principle by discarding patients that present missing values [**1**].

## 2. LOCF-Last Observation Carried Forward

Last Observation Carried Forward (LOCF) is the first "true" imputation method (unlike Complete Case Analysis, that does not impute values) that we will introduce. LOCF is an imputation method that is designed for longitudinal data. The idea is to replace a missing value by the last observed value. Formally, suppose we want to apply LOCF on a given individual $i$ with data vector $y_i = (y_{i1}, ..., y_{il})$ and missing vector indicator $m_i = (m_{i1}, ..., m_{il})$. Then, the LOCF imputation method can be formalized as

$$y_{ij}^{LOCF} = y_{ij'}, j = 2, ..., l$$

where

$$j' = \max_{\substack{j'' \leq j \\ m_{ij''}=1}} j''.$$

It is interesting to note that if data is missing for $j = 1$, this technique is not clearly defined. Sometimes, when this configuration happens, the applied rule is to replace it by the first observed value. LOCF is usually applied in case of dropout but can also be applied in case of general missing data. It relies on strong and unrealistic assumptions (stronger than MCAR assumption). To apply LOCF, one should believe that a subject's measurements do not change from the moment of dropout (or between two observed measurements, in case of intermittent missingness). In practice, this behaviour is unlikely to happen. However, for a long time, LOCF has been viewed as the primary method for

treating missing data in clinical trials by the U.S. Food and Drug Administration. There were two main justifications to use LOCF. The first one is that it resolves the ITT (intention to treat) problem in a very simple way. There is no patient selection like in Complete Case Analysis. The second one is that it was thought to be a conservative method (a method that favors the error of concluding that there is no treatment difference whereas in fact there is a treatment difference). But Molenberghs and Kenward [**1**] showed that the bias on treatment effects can operate in both directions if the LOCF assumption is not satisfied. The bias can vary in sign, but also in magnitude, depending on the values of the missing data. Thus, if LOCF can exaggerate the magnitude of treatments effects, it can inflate Type I Error. Verbeke and Molenberghs [**14**] show that all statistics can be affected by this bias : group difference, evolution over time, variance structure, correlation structure, etc.

Kenward and Molenberghs [**13**] propose the following example to understand why LOCF is not appropriate for clinical trials. They highlight that LOCF is often presented with examples in which the disease shows a progressive improvement over time. But, if the goal of the treatment is maintenance in a progressively worsening disease state, LOCF can exaggerate treatment benefit by assuming that a patient state stabilizes after dropout while it is likely to worsen. LOCF implies no improving state, but the crucial point is that it also implies no worsening state and so it is not conservative. Molenberghs and Kenward [**1**] suggest to replace LOCF by more complex methods such as Multiple Imputation (covered in Chapter 3), since statistical theory has evolved and computer power is much higher than before. Van Buuren [**6**] agrees with this conclusion for outcome variables, but adds that LOCF can still be convenient for administrative variables in longitudinal studies. Indeed, the address, social security number, ID card number, ... is unlikely to change over time. An example of such an acceptable way of using LOCF can be found in Table 1. A simulation study made by Zhu [**3**] that compares LOCF to other simple methods also supports the fact that LOCF should be avoided.

Sometimes in clinical trials another similar method is used : BOCF (Baseline observation carried forward). It consists in replacing each missing value for a given patient by its baseline value. Unsurprisingly, Kenward and Molenberghs show that it suffers from similar drawbacks as LOCF [**13**].

| | Observed Data | | | | LOCF Data | | | |
|---|---|---|---|---|---|---|---|---|
| | Week 1 | | Week 2 | | Week 1 | | Week 2 | |
| Patient | Outcome | Address | Outcome | Address | Outcome | Address | Outcome | Address |
| 1 | 18 | 5th Avenue, LA | 16 | 5th Avenue, LA | 18 | 5th Avenue, LA | 16 | 5th Avenue, LA |
| 2 | 16 | Coleman St., CA | 16 | * | 16 | Coleman St., CA | 16 | Coleman St., CA |
| 3 | 17 | Kinney St., CA | 16 | Kinney St., CA | 17 | Kinney St., CA | 16 | Kinney St., CA |
| 4 | 20 | Granville Lane, NJ | 19 | * | 20 | Granville Lane, NJ | 19 | Granville Lane, NJ |

TABLE 1. Acceptable way of using Last Observation Carried Forward, on administrative data. Symbol '*' represents missing data.

## 3. Mean Imputation

The idea behind mean imputation (also referred to as unconditional mean imputation) is to replace the missing values for a given variable $Y$ by its mean among the observed values. In fact, in a multivariate setting, this means that the imputation will be accomplished on each variable that contains missing data. For longitudinal data, mean imputation is usually performed in another way. This way is to compute the mean for a given patient/observation instead of by time-points. It is called *person mean imputation* [**6**] but will not be further treated here. For a categorical variable, one should replace missing values by the mode.

Van Buuren [**6**] highlights the fact that this method leads to a lot of distortions in statistical analysis. For example, even in a MCAR case, mean imputation biases almost any estimate other than the mean. If not in a MCAR case, it introduces a bias also for the mean. It also leads to underestimate variance and standard deviation. This underestimation can be easily quantified following an argument from Little and Rubin [**5**], as follows. Suppose we look at a variable $Y$ containing missing data, and this data is MCAR. We have $N_{obs}$ observed data and $N - N_{obs}$ missing data. Then, the variance of the observed values $Y^o$ is

$$Var(Y^o) = \frac{\sum_{i=1}^{N_{obs}} (Y_i^o - \bar{Y})^2}{N_{obs} - 1}$$

and this variance is unbiased, since we are in a MCAR case. We can compare it to the variance of the observed and imputed data, that is our completed data after imputation. We have :

$$Var(Y^{\text{Comp}}) = \frac{\sum_{i=1}^{N} (Y_i^{\text{Comp}} - \bar{Y})^2}{N - 1} = \frac{\sum_{i=1}^{N_{obs}} (Y_i^o - \bar{Y})^2}{N - 1}.$$

From this we deduce that mean imputation underestimates the variance by a factor of $(N_{obs}-1)/(N-1)$. Similarly, for two variables $Y_j$ and $Y_k$, mean imputation will underestimate their covariance by a factor $(N_{obs}^{jk}-1)/(N-1)$, where $N_{obs}^{jk}$ is the number of cases where both $Y_j$ and $Y_k$ are observed. This underestimation is not surprising, since we introduce imputed data for a variable, but this imputed data is completely uncorrelated with data from other variables. For these quantities, we can easily find adjustments factors since we know by what factor they are biased.

Another drawback that comes with this method is the distortion of the shape of the distribution of a variable. It often leads to bimodal distributions even though the original variable distribution is unimodal. This phenomenon is illustrated in Figure 1 in the particular case of an original chi-squared distribution.

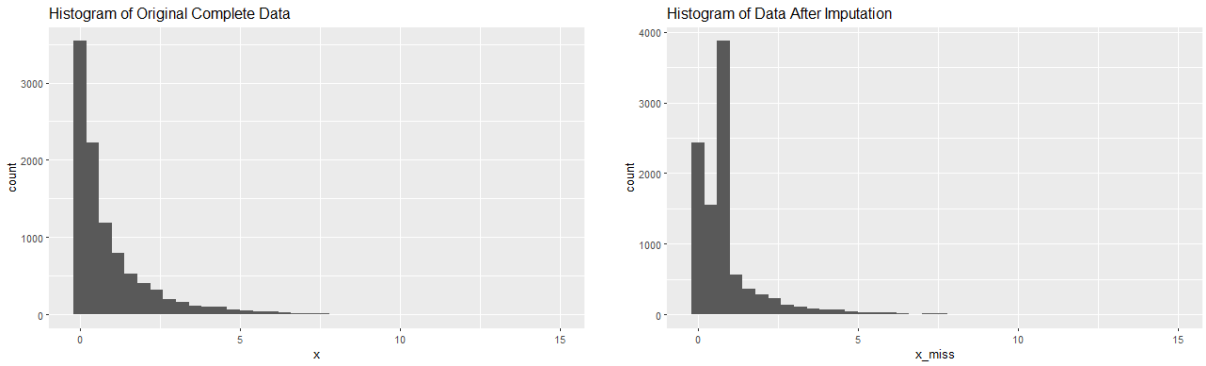


FIGURE 1. Left : Histogram of 10000 values that are randomly drawn from a chi-squared distribution with 1 degree of freedom. Right : From the original data, 30% are randomly set as missing (so it is a MCAR case). The graph shows the histogram of the data after mean imputation. A bimodal shape is appearing.

The general suggestion is to use mean imputation as a rapid fix (for example to have a first look at the data at hand) if only a handful of values are missing. Otherwise, it should be avoided.

## 4. Regression Imputation

Regression imputation (also referred to as conditional mean imputation) has been introduced by Buck [**16**] with the aim of producing smarter imputations than the previous imputations methods. A natural way to achieve this goal is to incorporate the knowledge contained in other variables in the imputation procedure. This can be done by replacing missing values with predicted values from a regression model. So, the first step is to build a regression model from observed data (usually using a complete case analysis). The second step consists in making predictions for the incomplete cases, by picking the most likely value under the fitted model. The intuition behind this method is that it makes sense to try to generate imputed values from observed data, since it is not infrequent to have correlations between variables. At the end, it generates a complete dataset from which one can perform traditional statistical analysis for complete datasets. Van Buuren [**6**] outlines the problems that arise with this method.

First, imputed values are the most likely values under the model, but it is actually unlikely that the real values are exactly these values. As a consequence, regression imputation lowers the variability of the data.

Secondly, the method suffers from the exact opposite problem as mean imputation. Indeed, the correlation coefficients are overestimated since imputed values have a correlation of 1 with the regressors. This makes regression imputation a dangerous method because it artificially strengthens the relations in the data. For example, this behaviour could lead to inflating Type I errors for clinical trials. In fact, if some variable of interest (for example a drug effect) has its correlation with a covariate (for example the dose of the drug) artificially increased, one could conclude that there is an effect of increasing the drug dose while there is not in reality.

As an assessment for this method, we can also say that it gives unbiased estimates of the means under MAR. Another positive result is that regression weights are also unbiased even under MAR [**6**]. In the case where different variables contain missing data, the number of regression equations to solve can become very large. We illustrate this problem in Example 2.3. Formally, the number of equations needed for any combinations of $v$ among $l$ variables is

| Missing variables | Regression equations | |
|---|---|---|
| $Y_1$ | $\hat{y}_1 = \hat{\beta}_0 + \hat{\beta}_1 y_2 + \hat{\beta}_2 y_3$ | |
| $Y_2$ | $\hat{y}_2 = \hat{\beta}_0 + \hat{\beta}_1 y_1 + \hat{\beta}_2 y_3$ | |
| $Y_3$ | $\hat{y}_3 = \hat{\beta}_0 + \hat{\beta}_1 y_1 + \hat{\beta}_2 y_2$ | |
| $Y_1$ and $Y_2$ | $\hat{y}_1 = \hat{\beta}_0 + \hat{\beta}_1 y_3$ | $\hat{y}_2 = \hat{\beta}_0 + \hat{\beta}_1 y_3$ |
| $Y_1$ and $Y_3$ | $\hat{y}_1 = \hat{\beta}_0 + \hat{\beta}_1 y_2$ | $\hat{y}_3 = \hat{\beta}_0 + \hat{\beta}_1 y_2$ |
| $Y_2$ and $Y_3$ | $\hat{y}_2 = \hat{\beta}_0 + \hat{\beta}_1 y_1$ | $\hat{y}_3 = \hat{\beta}_0 + \hat{\beta}_1 y_1$ |

TABLE 2. List of equations used for regression imputation in Example 2.3.

$$v\frac{l!}{v!(l-v)!}.$$

The left multiplicative factor $v$ comes from the fact that each of the $v$ variables needs a specific regression equation. The fraction on the right is the number of possible combinations of $v$ among $l$ variables. Then, a simple expression for the total number of regression equations needed for the dataset (in the worst case, i.e. the case where all the missing combinations are present in the data) is

$$\sum_{v=1}^{l-1} v\frac{l!}{v!(l-v)!} = l\sum_{v=1}^{l-1}\binom{l-1}{v-1}.$$

EXAMPLE 2.3 (Multivariate missing data (Enders [**17**])). Suppose we have a dataset composed of 3 variables, $Y_1$, $Y_2$ and $Y_3$. Each of them contains missing data. Besides the complete cases, 6 configurations of missing data are possible. Either $Y_1$, $Y_2$ or $Y_3$ is missing, if there is only one missing value. If there are two missing values, we have the following missing variable combinations : $Y_1$ and $Y_2$, $Y_1$ and $Y_3$, or $Y_2$ and $Y_3$. A summary of all the equations needed is shown in Table 2. We see that indeed we obtain $3\sum_{v=1}^{3-1}\binom{3-1}{v-1} = 3(1+2) = 9$ equations, which is consistent with the theoretical formula. □

Finally, we mention an iterative method proposed by Van Buuren [**6**]. The algorithm is made of the following steps (for a dataset composed of variables $Y_1$, ..., $Y_l$) :

(1) Fill the dataset using a simple imputation method (Mean imputation, ...).
(2) Repeat T times.
(3) Repeat for j=1,...,l.
(4) Fit the regression weights $\hat{\beta}_{j'}$, $j' = 0,...,l$ and $j' \neq j$ with
    -$Y_j$ as response variable

-$Y_{j'}$, $j' = 1,...,l$ and $j' \neq j$ as explanatory variables.

(5) Replace the originally missing values with the equation $\hat{y}_j = \hat{\beta}_0 + \sum_{\substack{j'=1 \\ j' \neq j}}^{l} \hat{\beta}_{j'} y_{j'}$.

Missing data are imputed variable by variable. It allows to always use the entire (completed) dataset to fit the regression weights and potentially include more information in the analysis than with Buck's method. One iteration through the algorithm is defined as an entire cycle through all $Y_j$, and $T$ is then the number of iterations. Often, in practice, $T$ is not taken very large (in general between 5 and 20). This iterative algorithm is worth mentioning since sometimes this method is used implicitly for regression imputation. It will be further detailed for its use and generalization in Multiple Imputation (see Chapter 3, Section 7). Also, it will be the one used in the simulation part (see Chapter 4). However, even if the method works not so badly in practice, it comes with no clear theoretical performance assessment [**6**].

## 5. Stochastic Regression Imputation

As we have discussed in the previous section, the major drawback of regression imputation is that it overestimates the correlation coefficients between variables. Stochastic regression imputation is designed to tackle this problem. Indeed, it is a simple refinement of standard regression imputation. It also uses regression equations to predict the incomplete variables from the complete variables but adds an extra step at the end that consists in adding a normally distributed residual term to the predicted values. At first sight, it seems counter-intuitive to deviate from the best prediction of the model by adding noise, but this is actually preferable for imputation purposes, since the goal is to yield data that seem realistic (in the sense that imputed data should reflect as much as possible what the data could be if they were not missing).

The method is decomposed by Enders [**17**] into the following steps :

(1) estimate a set of regression equations that predict the incomplete variables from the complete variables (using a CCA analysis).
(2) substitute the observed scores into these equations to obtain the predicted values for missing data.
(3) add to each predicted value a draw from a normal distribution with mean 0 and variance equal to the residual variance from the regression.

As for regression imputation, if we are in presence of multivariate missing data, the first step can be a bit complex. Indeed, it is necessary to model different regression equations for each missing data pattern. The procedure is then similar, with the only difference consisting in estimating also the residual distribution for each equation. Little and Rubin [**5**] show that stochastic regression imputation yields unbiased estimates under MAR, even for correlation. This makes it the only traditional and basic imputation method completely theoretically justified under MAR.

## 6. Summary

To conclude this chapter, we provide in Table 3 a summary of the imputation methods analyzed. First, it is interesting to look at the assumptions that lead to unbiased estimates (that is, all columns except for the last one that is about standard error). We see that LOCF is by far the poorest method since even in the MCAR case, the mean, the regression weights and the correlation are biased. Secondly, mean imputation is only unbiased for mean estimate and only in the MCAR case. CCA is always unbiased under MCAR assumption. Stochastic regression is the very best method here since it leads to unbiased estimates even under MAR assumption (standard regression is similar, except for an introduced bias in correlation coefficient).

| | Mean | Reg. Weights | Correlation | Standard Error |
|---|---|---|---|---|
| CCA | MCAR | MCAR | MCAR | Too large |
| LOCF | - | - | - | Too small |
| Mean Imputation | MCAR | - | - | Too small |
| Regression Imputation | MAR | MAR | - | Too small |
| Stochastic Reg. Imputation | MAR | MAR | MAR | Too small |

TABLE 3. For each method encountered in this chapter, the table shows the assumptions needed to lead to unbiased estimates (for mean, regression weights and correlation). The symbol '-' indicates that there is no assumption (between MCAR and MAR) that leads to unbiased estimates. In addition, the last column indicates whether the estimated standard error of these estimates is too small or too large.

Secondly, it is also interesting to look at the estimated standard errors of these estimates. Complete Case Analysis yields too large standard errors. This come from the fact that it

excludes a part of the observations from the analysis, leading to more uncertainty over the value of the estimate. Such high standard errors are the direct cause of a diminution of power of the statistical tests computed after CCA. Formally, it is easy to check that the number of items $N$ appears at the denominator of every standard error formula, and so a diminution of $N$ implies a higher standard error. Other methods than CCA have too small standard errors. This is actually the case for every single-imputation method. As outlined by Little and Rubin [**5**], the key problem of single imputation techniques is that inferences about parameters based on the completed data do not account for imputation uncertainty. Indeed, they actually treat imputed data and observed data on the same level.

In Chapter 3, we will see that Multiple Imputation corrects this standard error problem. Another idea could be to search for corrections factors (Schafer and Schenker proposed such factors [**22**]), but Multiple Imputation seems to remain the most convenient imputation method.

CHAPTER 3

# Multiple Imputation

The main principle of multiple imputation is to replace each missing value with a set of $K$ plausible values instead of just one as for single imputation techniques described in Chapter 2. In this way, we create $K$ complete datasets and each one can then be analyzed using classical statistical tools, as we would have done if the initial dataset was complete. Then, the $K$ obtained results are pulled together in order to produce the final analysis. There are several methods to impute data for Multiple Imputation and we will detail two of them : Joint Modeling (Section 6) and Fully Conditional Specification (Section 7).

## 1. Procedure

As described by Molenberghs and Kenward [**1**], the whole procedure of multiple imputation can be divided into 3 main tasks. First, the imputation model. It will be used as a first step in the procedure, in order to obtain $K$ completed datasets out of the initial one containing missing data. Secondly, the substantive model, which is the model used to analyze each completed dataset. Finally, the last task is to combine the results obtained for each of the $K$ datasets into a single inference.

Let us detail more formally how these tasks are performed. Let's introduce a parameter vector $\beta$ of size $p \times 1$. It represents the parameters of the substantive model. Suppose that we can make draws from the imputation model (we will detail how to do it later, whether by Joint Modeling or by Fullt Conditional Specification). Then, for each of the $K's$ completed datasets, we have $\hat{\beta}^k$, $V^k$, an estimate of $\beta$ and an estimate of its variance-covariance matrix ($k = 1,...,K$). So far we have completed the first task (imputation) and the second task (estimating parameters of the substantive model for each completed dataset). For the third and final task, we take as final estimator for $\beta$:

$$\hat{\beta}^* = \frac{1}{K} \sum_{k=1}^{K} \hat{\beta}^k$$

which is the average of all estimators calculated in the second task. The formula for the variance-covariance matrix of $\hat{\beta}^*$ is less intuitive and we need to introduce some notions first. Rubin [**9**] details the covariance matrix decomposition as follows. The first element of the final formula is $W$, the so-called *within covariance matrix*

$$W = \frac{1}{K}\sum_{k=1}^{K} V^k$$

It is the average of the covariance matrices associated to each $\hat{\beta}^k$. This part of the variability comes from the fact that we are looking at a sample of the population rather than looking at the entire population. The second component that will appear in the final formula for the covariance matrix of $\hat{\beta}^*$ is the variance between the $\hat{\beta}^k$. It is called the *between covariance matrix*

$$B = \frac{1}{K-1}\sum_{k=1}^{K} (\hat{\beta}^k - \hat{\beta}^*)(\hat{\beta}^k - \hat{\beta}^*)'$$

Now that if we have these two components, we could think of an estimator of the total variance $V$ of $\hat{\beta}^*$ that would be calculated as the sum of the within covariance matrix and the between covariance matrix

$$V_{\text{naive}} = W + B$$

Actually, this formula is unbiased if $K \to \infty$. But for finite $K$, the formula does not take into account that $\hat{\beta}^*$ is approximated. Rubin [**9**] shows that there is then another contribution to the total variance $V$, and that this contribution is equal to $\frac{1}{K}B$. If we put everything together, we get

$$V = W + B + \frac{1}{K}B = W + \frac{K+1}{K}B$$

The term $\frac{1}{K}B$ can be quite high if K is small (and in practice there are a lot of examples where multiple imputation is computed with K=3 or K=5). So it is crucial to take it into account in order not to shorten artificially confidence intervals, for example.

## 2. Theoretical Justification

Molenberghs and Kenward [**1**] give a Bayesian argument that gives the intuition on why to use multiple imputation. Some equations are not developed entirely, so we reproduce the argument here with all the details. Suppose that we look at a problem with two

parameters $\gamma_1$, $\gamma_2$ and data $Y$. We consider $\gamma_1$, $\gamma_2 \in \mathbb{R}$, even though the argument can be generalized to $\gamma_1$, $\gamma_2$ being vectors. The joint distribution of $\gamma_1$, $\gamma_2$ given the data can be written as

$$f(\gamma_1, \gamma_2|Y) = f(\gamma_1|Y)f(\gamma_2|\gamma_1, Y)$$

Now suppose that we consider $\gamma_1$ as a nuisance parameter that is not of particular interest. Using the law of total probability we obtain a marginal posterior for $\gamma_2$

$$f(\gamma_2|Y) = \int_{-\infty}^{\infty} f(\gamma_2|\gamma_1, Y)f(\gamma_1|Y)d\gamma_1 = \mathbb{E}_{\gamma_1}[f(\gamma_2|\gamma_1, Y)]$$

Similarly, with the law of total expectation and the law of total variance we obtain

- $\mathbb{E}[\gamma_2|Y] = \mathbb{E}_{\gamma_1}[\mathbb{E}_{\gamma_2}(\gamma_2|\gamma_1, Y)]$
- $\text{var}[\gamma_2|Y] = \mathbb{E}_{\gamma_1}[\text{var}_{\gamma_2}(\gamma_2|\gamma_1, Y)] + \text{var}_{\gamma_1}[\mathbb{E}_{\gamma_2}(\gamma_2|\gamma_1, Y)]$

Suppose that we can make K random draws from the marginal distribution of $\gamma_1 : \{\gamma_1^1, ..., \gamma_1^K\}$. Using these values, we can now give an approximation of the expectation and variance of $\gamma_2$:

- $\mathbb{E}[\gamma_2|Y] \approx \frac{1}{K}\sum_{k=1}^{K} \mathbb{E}_{\gamma_2}(\gamma_2|\gamma_1^k, Y) = \tilde{\gamma}_2$
- $\text{var}[\gamma_2|Y] \approx \frac{1}{K}\sum_{k=1}^{K}[\text{var}_{\gamma_2}(\gamma_2|\gamma_1^k, Y)] + \frac{1}{K-1}\sum_{k=1}^{K}[\mathbb{E}_{\gamma_2}(\gamma_2|\gamma_1^k, Y) - \tilde{\gamma}_2]^2$

Now that we have these results, we can see the link with the multiple imputation procedure (in Section 1). We just think of $\gamma_2$ as being the parameters of the substantive model and $\gamma_1$ as the missing data, viewed as noise for the estimation of the distribution of $\gamma_2$. The formula for $\mathbb{E}[\gamma_2|Y]$ can then be linked to the formula for $\hat{\beta}^*$. Similarly, for $\text{var}[\gamma_2|Y]$, the left side of the addition corresponds to the formula of $W$, while the right side corresponds to the formula for $B$. These results highlights the fact that the formulas stated in Section 1 can be obtained by the previous Bayesian argument.

## 3. Efficiency

Rubin [**9**] calculated the relative efficiency of an estimator using multiple imputation (with finite K) compared to the theoretical efficiency of this same estimator with $K \to \infty$. He showed that the relative efficiency is equal to

$$\frac{\text{efficiency}_{\text{finite}K}}{\text{efficiency}_{K\to\infty}} = \left(1 + \frac{\pi}{K}\right)^{-1/2}$$

where $\pi$ is the fraction of missing data among all data. Observe that this quantity is always smaller than 1, which is normal since the efficiency of the estimator is expected

| K\$\pi$ | 0.1 | 0.3 | 0.5 | 0.7 |
|---|---|---|---|---|
| 2 | 98 | 93 | 89 | 86 |
| 3 | 98 | 95 | 93 | 90 |
| 5 | 99 | 97 | 95 | 94 |
| 10 | 100 | 99 | 98 | 97 |
| $\infty$ | 100 | 100 | 100 | 100 |

TABLE 1. Relative efficiency (in percentage, rounded to unity) of multiple imputation estimator with respect to K and $\pi$

to grow with $K$.

In Table 1, we show different efficiency values of the multiple imputation estimator. The benefit of making the number of imputations $K$ grow is slowing down rapidly. Even with a 70 % of missing data, taking $K = 10$ imply already a relative efficiency of 97%. It seems then that there is no need to take $K = 50, 100, ...$ as it is sometimes done, especially if one has limited computational power.

## 4. Making proper imputations

The validity of the final statistical analysis will of course rely heavily on the imputation method. Rubin [**9**] first introduced the term of *proper imputations*, which would be defined precisely as imputation methods leading to valid estimates.
About the formalization of the definition of *proper imputations* methods, Rubin adds the following disclaimer:

*"If imputations are drawn to approximate repetitions from a Bayesian posterior distribution of $Y_{mis}$ under the posited response mechanism and an appropriate model for the data, then in large samples the imputation method is proper ... There is little doubt that if this conclusion were formalized in a particular way, exceptions to it could be found. Its usefulness is not as a general mathematical result, but rather as a guide to practice. Nevertheless, in order to understand why it may be expected to hold relatively generally, it is important to provide a general heuristic argument for it."*

This citation is extremely important, since it defines the goal of this section. This goal is not to prove (in a mathematical sense) that some methods are proper, but rather to give

| Incomplete Sample | | Complete Sample | | Population |
|---|---|---|---|---|
| $Y_{obs}$ | | $Y = (Y_{obs}, Y_{mis})$ | | |
| $\hat{\beta}^*$ | $\Rightarrow$ | $\hat{\beta}$ | $\Rightarrow$ | $\beta$ |
| $W$ | $\Rightarrow$ | $\mathbb{W} \doteq V(\hat{\beta})$ | | |
| $B \doteq V(\hat{\beta}^*)$ | | | | |

TABLE 2. Role of symbols (defined in Section 1) at three analytic levels and the relations between them. The relation $\Rightarrow$ means "is an estimate of". The relation $\doteq$ means "is asymptotically equal to".

heuristical arguments that lead to a set of broad guidelines.

We will now explicitly describe these guidelines. First, we start by a quick recall about the different analytic levels on which we are working. The three levels are described in detail by Van Buuren [**6**]. We report all the symbols and concepts on Table 2. Our first and main goal is to find an estimate $\hat{\beta}$ of $\beta$ with good statistical properties for inferences purpose on the population. If data are complete, we have only two analytical levels : the (complete) sample and the population. Then, our knowledge about $\beta$ can be characterized by its estimator $\hat{\beta}$ and $\mathbb{W}$. $\mathbb{W}$ is the variance-covariance matrix of $\hat{\beta}$ and captures its uncertainty under repeated sampling. This step enables us to go from the Complete Sample analytic level to the population level.

In case we are in presence of missing data, a third analytical level appears : the incomplete sample level. We want to go from the incomplete sample level to the complete sample level. Now, instead of having one estimand $\beta$ at the population level, we have two estimands $\hat{\beta}$ and $\mathbb{W}$ at the complete sample level. What we expect from a correct multiple imputation procedure is to lead to estimates of $\hat{\beta}$ and $\mathbb{W}$ with good statistical properties.

Van Buuren [**6**] rewrites the guidelines of Rubin as a set of 3 conditions, as follows. An imputation procedure is said to be proper with respect to a pair of complete sample statistics $(\hat{\beta}, \mathbb{W})$ if for large $K$:

- $\mathbb{E}[\hat{\beta}^* | Y] = \hat{\beta}$
- $\mathbb{E}[W | Y] = \mathbb{W}$
- $\left(1 + \frac{1}{K}\right)\mathbb{E}[B | Y] \geq V[\hat{\beta}^*]$

The first condition states that $\hat{\beta}^*$ should be an unbiased estimator of $\hat{\beta}$. It means that if we start with complete data $Y$ and recreate a large number of times an incomplete dataset (using always the same missing data mechanism), the average of the $\hat{\beta}^*$ obtained will be $\hat{\beta}$.

The second condition states that $W$ should be an unbiased estimator of $\mathbb{W}$. It means that if we start with complete data $Y$ and recreate a large number of times an incomplete dataset (using always the same missing data mechanism), the average of the $W$ obtained will be $\mathbb{W}$.

The third condition is a bit different, since $B$ is not a direct estimator of a quantity on the complete sample level, but rather a quantity characterizing the extra uncertainty on $\hat{\beta}$ due to missing data. The condition implies that the extra inferential uncertainty is correctly reflected.

Note that the definition of proper imputation procedure is conditional on the pair of statistics $(\hat{\beta}, \mathbb{W})$. This implies that a procedure could be proper for the pair $(\hat{\beta}, \mathbb{W})$ but not for another pair of statistics $(\hat{\beta}', \mathbb{W}')$.

## 5. Inference under multiple imputation

The procedure of multiple imputation is based on a Bayesian argument. However, in practice, the inference step in multiple imputation is frequentist. Let's first look at the case where $\hat{\beta}^*$ is asymptotically normally distributed, which is the case for means, standard deviations, regression coefficients, ... In this case tests and confidence intervals are based on the classical Wald statistic

$$P = (\hat{\beta}^* - \beta)' V^{-1} (\hat{\beta}^* - \beta)$$

Note that $P$ depends on the sample size $N$ through the matrix $V$ ($V$ is the total variance matrix, as defined in Section 1). In a conventional setting, the statistic $P$ would follow a chi-squared distribution

$$P \sim \chi^2_p$$

with $p$ being the size of the vector $\hat{\beta}^*$. However, in a multiple imputation context, the distribution of $P$ does not solely depend on the sample size $N$ but also on the number of

imputations $K$ and therefore its distribution is more complex. That is why Li, Raghunathan and Rubin [**8**] introduce another pivotal quantity based on $P$. They propose the use of a $F_{p,w}$ distribution for the pivot

$$F = \frac{P}{p(1+r)}$$

with

$$w = 4 + (\tau - 4)\left[1 + \frac{1 - 2\tau^{-1}}{r}\right]^2,$$

$$r = \frac{1}{p}\left(1 + \frac{1}{K}\right) tr(BW^{-1}),$$

$$\tau = p(K-1).$$

Note that the real distribution of the statistic $F$ is more complex, but the authors show with simulations that the $F_{p,w}$ distribution is a good approximation (and better than previous approximations found in the literature). The $r$ factor can be seen as the relative increase of variance due to missingness across the components of $\beta$ (and supposes this increase is the same for all components). When $K \to \infty$, the F statistic tends to the classical $\chi^2_p$ distribution.

Recall that the previous procedure can be applied when $\hat{\beta}^*$ is asymptotically normally distributed. Suppose now that $\hat{\beta}^*$ is a non-normal quantity. This happens for example when looking at correlation coefficients, odds ratios, explained variance, ... In this case, the only procedure at hand is transformation towards normality, apply Li's rule [**8**] and back-transformation into the original scale. Let's make an example for the correlation coefficient. That is, $\hat{\beta}^k$ is a correlation coefficient, for $k = 1, ..., K$. Schafer [**10**] suggests to use the Fischer z-transformation

$$\hat{\beta}^k_z = \frac{1}{2} ln \frac{1 + \hat{\beta}^k}{1 - \hat{\beta}^k}.$$

For large samples, the distribution of $\hat{\beta}^k_z$ is normal with variance $\sigma^2 = 1/(N-3)$. From these we can pool $\hat{\beta}^*_z$ and infer confidence intervals applying Li's rules. Then the result can be back-transformed using the following formula

$$\hat{\beta}^* = \frac{e^{2\hat{\beta}_z^*} - 1}{e^{2\hat{\beta}_z^*} + 1}.$$

For odds ratio, Agresti [**11**] suggests to use a $log$ transformation while for explained variance, Harel [**12**] suggests to use also a Fisher z-transformation. For non-normal quantities that are not cited here, one should find an appropriate procedure of transformation towards normality.

## 6. Joint Modeling

As explained in Section 1, Multiple Imputation is performed following three distinct phases. Sections 1 to 5 of this chapter were mainly concentrated on the second and third phases. That is, the description of the $K$ substantive model parameter vectors, $\hat{\beta}^k (k = 1, ..., K)$, and their combination into one single inference. However, the first phase, that consists in building an imputation model has not yet been covered. This crucial task that enables to build the $K$ completed datasets will be detailed in this section (for the Joint Modeling approach) as well as in the next one (for the Fully Conditional Specification approach). Both are solutions that work with multivariate missing data. Solutions for univariate missing data also exist, but it is not common in practice to observe such data. In addition, solutions that work for multivariate data can be applied also to multivariate data, so in the following we will focus on that case only.

The Joint Modeling (JM) approach relies on one big assumption : data can be described by a multivariate distribution [**6**]. The idea is then to try to find this multivariate distribution, so that draws from this distribution will constitute the imputed values. The multivariate distribution is theoretically of any form, but in practice the main choice is to try to fit a multivariate normal distribution to the data. It is important to understand that this fitted model will not be unique. Indeed, the distribution from which imputations need to be drawn varies from row to row. It depends on the missing data pattern, that is, for row $i$, the corresponding missing data vector $m_i = (m_{i1}, ..., m_{il})$, with the notations introduced in Chapter 1, Section 1. For rows with the same missing data pattern, imputations are drawn from the same distribution. Below is an example of how the multivariate distribution should be fitted, with respect to $m_i$.

EXAMPLE 3.1 (Imputation model distribution). Suppose data is composed by 4 variables $Y_1$, $Y_2$, $Y_3$, $Y_4$. The $i^{\text{th}}$ row has a corresponding missing data vector $m_i = (0, 0, 1, 1)$. For

this row, the imputation model distribution will take the form of $f_i(Y_1, Y_2|Y_3, Y_4, \phi_{1,2})$ where $\phi_{1,2}$ is the parameter of the imputation model (as explained at the beginning of the section, it has not to be confused with the parameters of the substantive model). In this case, the imputation model is bivariate. Even in a JM approach, some of the imputation models can be univariate. For example, if $m_{i'} = (0,1,1,1)$, the imputation model distribution will take the form of $f_{i'}(Y_1|Y_2, Y_3, Y_4, \phi_1)$.

Schafer [**10**] proposes an algorithm to handle multivariate data by joint modeling. The algorithm exploits the fact that if data is a draw of a multivariate normal (which is the common assumption in this context), the different parameters of the imputation model can be easily computed from the parameters of that multivariate normal. Indeed, as he explains, if data are such that

$$Y \sim N(\tilde{\phi})$$

(with $\tilde{\phi}$ containing all parameters of the normal), then the different imputation models parameters $\phi$ can be described as functions of $\tilde{\phi}$ [**10**]. These transformations from $\tilde{\phi}$ to $\phi$ are performed using the *sweep operator*. The sweep operator enables to build regression models by sweeping in or sweeping out particular variables of the model. An accessible reference for details about this operator is written by Goodnight [**23**]. The definition below explains in a nutshell what are the operations behind the sweep operator.

DEFINITION 3.2 (Sweep Operator [**23**]). *Let $A \in \mathbb{R}^{l\times l}$ be a symmetric positive definite matrix. Let SWEEP : $(\mathbb{R}^{l\times l}, \mathbb{R}) \to \mathbb{R}^{l\times l}$. SWEEP(A, i) modifies the matrix A by using elements of the $i^{th}$ row as follows :*

(1) *Divide the $i^{th}$ row by $A_{ii}$.*
(2) *For every row $i' \neq i$ :*
   (a) *Subtract $A_{i'i} \times$ (row $i$) from row $i'$.*
   (b) *Set $A_{i'i} = -A_{i'i}/A_{ii}$.*
(3) *Set $A_{ii} = 1/A_{ii}$*

This sweep operator can be used in a wide variety of applications. The one who interests us in the context of this section is its application to the problem of all-subsets linear regression (that is, finding all possible regression equations given a set of variables). Indeed, it is exactly what has to be done to go from $\tilde{\phi}$ to the different $\phi$ of the imputation models. The sweep operator, in this context, is applied to the matrix $Y^TY$. By sweeping in a row of this matrix, the corresponding variable is included as explanatory variable in

the model. By sweeping the same row a second time, the corresponding variable will be put back as dependent variable. This application of the sweep operator and the algebra behind it is explained in detail by Little and Rubin [**5**].

Now, we can go back to the algorithm proposed by Schafer [**10**]. The principle problem in the above explanation on how to go from $\tilde{\phi}$ to $\phi$ is that none of them are known. Without going too deep into details, the idea of Schafer is to compute these quantities iteratively, as follows :

(1) Initialize $\tilde{\phi}^0$ by a reasonable starting value.
(2) Repeat for t=1,...,T (T is the total number of iterations).
(3) Using the sweep operator, obtain the different $\phi$'s parameters (one for each missing pattern in the data).
(4) Draw imputation values from the imputation models derived at the previous step.
(5) $\tilde{\phi}^t$ is computed as a draw from the normal-inverse-Wishart distribution (with non-intuitive parameters defined by Schafer [**10**]).

At the end of the procedure, we obtain an imputation model for each missingness pattern. We can create $K$ complete datasets by computing $K$ draws from these distributions.

Observe that the latter formulation of the Joint Modeling approach is well defined for continuous data, but not for categorical data. Often in practice, the method is adapted to categorical data by rounding off continuous imputed values in categorical data to the nearest category. Of course, this works only for categories that can be put in a way that makes sense into numeric categories. Horton et al. [**24**] show that rounding in this way could introduce a bias in some situations. However, rounding is still the most wide-spread method to impute categorical data.

## 7. Fully Conditional Specification (and MICE algorithm)

The method that will be detailed in this section is the one that will be used during the simulation part (see Chapter 4) for Multiple Imputation and is called Fully Conditional Specification (FCS). The main idea is to impute multivariate missing data on a variable-by-variable basis. For each incomplete variable, a specific imputation model is built. Contrary to Joint Modeling, this approach does not fit a multivariate distribution to the data but a set of conditional densities

$$f(Y_j|Y_{-j},\phi_j), j = 1,...,l$$

where $Y_{-j}$ is defined as $Y \setminus Y_j$ [**6**]. Fully Conditional Specification is a more natural generalization from univariate to multivariate missing data than Joint Modeling. FCS is sometimes referred to as chained equations [**25**]. The latter has given its name to the MICE (Multiple Imputation by Chained Equations) algorithm.

Now that the general framework has been given, let us describe in details how the MICE algorithm works. Note that several instances of this algorithm exist, and the version that is proposed here is the one developed by Van Buuren and Groothuis-Oudshoorn [**25**]. Below are the different steps of the algorithm :

(1) Specify an imputation model $f(Y_j|Y_{-j},\phi_j)$ for each variable $Y_j$ with $j = 1,...,l$.
(2) For each $j$, fill in starting imputations $\hat{Y}_j^0$ by random draws from the observed values of $Y_j$.
(3) For t=1,...,T (T is the number of iterations):
(4) For j=1,...l:
(5) Draw $\dot{\phi}_j^t$ from the distribution of $\hat{\phi}_j$, the parameters that characterize the fitted conditional density $f(Y_j|Y_{-j},\hat{\phi}_j)$.
(6) Draw imputations for the initially missing values of $Y_j$ from $f(Y_j|Y_{-j},\dot{\phi}_j)$.

The algorithm starts with a random draw from the observed data in order to obtain a completed dataset. This is important because when the algorithm will try to fit a model to obtain imputed values for $Y_j$, the other variables $Y_{-j}$ need to be completed. Of course, when the focus is on one particular variable $Y_j$, the values imputed at the beginning are again set to missing for this variable. That is why the algorithm keeps track of where the missing data was located in the first place. To obtain $K$ completed datasets (which is the goal of multiple imputation) these steps are computed $K$ times in parallel.

The conditional densities appearing in the MICE algorithm are usually obtained from linear regressions. The imputations drawn randomly from these densities can then be seen as a generalized case of stochastic regression imputation (see Chapter 2, Section 5). The difference is that here, the parameter $\dot{\phi}_j$ is also a random draw from the distribution of $\hat{\phi}_j$, and not $\hat{\phi}_j$ itself. To sum up, stochastic regression is a method that takes into account the noise that has to be added to the optimal prediction. The method used here takes the

parameter uncertainty into account, in addition to the noise [**6**]. Note also that FCS is more flexible than Joint Modeling, since one can choose the type of conditional density to be used for each variable. For example, it is possible to use logistic regression for a binary variable. An analogous way of dealing with continuous variables in the Joint Modeling approach was complicated since it requires the use of a general multivariate distribution.

In the following, two issues coming with this algorithm will be covered. The first one is to determine what are the conditions needed for the algorithm to converge, and see if the underlying assumptions are reasonable. The second technical issue with this algorithm is to establish the number of iterations $T$ that should be computed in order to converge and obtain good results.

Let's start by the analysis of the convergence of the algorithm. Van Buuren highlights the fact that the chains of parameters created during the succession of iterations can be seen as a Markov chain [**6**]. Hence, in order to converge, the algorithm should satisfy the same conditions as for a Markov chain. These conditions are listed by Roberts [**26**] and presented below. In order to converge, the chain needs to be :

- Irreducible : the chain must be able to reach all interesting parts of the state space.
- Aperiodic : the chain should not oscillate between different states.
- Recurrent : all interesting parts can be reached infinitely often, at least from almost all starting points.

Now let's see if the MICE algorithm meets these conditions in practice. Van Buuren [**6**] makes the following comments on these three conditions. First, irreducibility. He states that this is not really a problem since the person using the MICE algorithm has control on the conditional densities and can adapt them in the purpose of reaching interesting parts of the space. For example, by choosing an appropriate model that suits to each variable (logistic regression for binary variables, etc). Secondly, periodicity can be a problem. However, the addition of noise to the imputed values greatly lowers the risk of being stuck into an oscillatory behavior. Finally, non-recurrence could also be a problem. But, as he states, the addition of noise is again the key to tackle this issue. Indeed, imputed values will be different at each iteration because of it, and prevent from being stuck in one particular state.

After having analyzed the convergence conditions, finding a good value for the number of iterations $T$ is a relevant question. Monitoring the convergence in the context of Multiple Imputation is fairly simple. It is done by plotting different statistics (usually mean and variance) for each of the $K$ completed datasets against the iteration number, $t$. The observed plots should be free of trend, and the variance within a chain should not be too far from the variance between chains [**6**]. In practice, the number of iterations chosen is often low (as low as 5 iterations, and not more than 20). This could seem strange since a lot of algorithms implying Markov chains require as much as thousands of iterations to converge. A potential reason why a small amount of iterations is already enough is the following. The injection of noise during the application of MICE to the data is substantial. Indeed, if some variables are not highly mutually correlated, the modeling of one of them with respect to the others will introduce some noise in the imputed values. However, this phenomenon is at the root of the convergence speed of the algorithm to a stationary point [**6**].

As mentioned before, the MICE algorithm presented above will be used to compute imputations in Chapter 4 during the simulations. This final remark concludes the chapter on Multiple Imputation.

CHAPTER 4

# Simulations : Impact of missing values on Support Vector Machines

## 1. Introduction

The goal of this Chapter is to provide a better understanding of the impact of missing values on the supervised classification method called Support Vector Machines. More precisely, the goal is to evaluate the impact of imputation techniques presented in the previous chapters on the accuracy of the support vector machine. In particular, we compare the following methods : mean imputation, regression imputation, stochastic regression imputation and multiple imputation (with two different numbers of imputations).

CCA and LOCF are discarded for this Chapter. The problem with LOCF is that it can be applied only on longitudinal data, and SVM is not known to work well for longitudinal data. There is also a problem with CCA. Indeed, it is not clearly defined what to do with this method if it has to be applied on the test set (data put apart when the support vector machine is fitted and used to determine the accuracy of the SVM afterwards).

Note, indeed, that our choice for these simulations is to consider that the test set also contains missing data. This is because we make the hypothesis that if data related to a specific context is missing with a specific mechanism during the elaboration of a model, it is likely that data in the future will also come with missing data from this particular mechanism. This is of course not the most general assumption but that seems realistic in a lot of contexts. Also, we assume that missing data occur only in the predictors of the SVM and not in the labels.

These choices make the following simulations in some way similar to the one of Stewart et al. [**29**]. However, the imputation methods and the missing data mechanisms used are not the same.

Section 2 gives a recall on SVM and the way they are constructed. In Section 3, we give the details of the way simulations are performed (in terms of data generation and missing data generation). Finally, results are given and discussed in Section 4.

All code and results for this chapter can be found on the following github link : `https://github.com/aurelebartolomeo/MasterThesis`. The code is presented on a RMarkdown document and includes both Python and R code.

Finally, we should highlight that the notations will differ a bit in this chapter from the previous ones. Indeed, since support vector machines are a supervised learning method, we need two different objects : the first one is a matrix that contains the data and the second one is a vector that contains the labels for this data. We choose $X$ for the data matrix and $Y$ for the labels. So in this chapter $X$ plays the role of $Y$ in the previous chapters.

## 2. Support Vector Machines

**2.1. Separating Hyperplane.** Hastie et al. [**18**] explain the concept of separating hyperplane in the following way. Let $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, $N$ pairs with $x_i \in \mathbb{R}^l$ $(i = 1, ..., N)$, $y_i \in \{-1, 1\}$ $(i = 1, ..., N)$. A separating hyperplane for this data is a hyperplane which separates $\mathbb{R}^l$ into two parts such that each vector $x_i$ associated to $y_i = 1$ is on the same side of the hyperplane, while each $x_i$ with $y_i = -1$ is on the other side. Logistic regression with perfect accuracy is an example of such linear decision boundary. Here, the focus will be put on separating hyperplane classifier (also called *linear SVM*). It refers to a procedure that constructs a linear decision boundary (the separating hyperplane) and that explicitly tries to separate the data into different classes. Data can be of two types, as represented in Figure 1. In the first case, data are not linearly separable. It will be treated in sections 2.3 and 2.4. In the second case, data are linearly separable, and there exists an infinite number of separating hyperplanes. In order to obtain a unique solution to the separating hyperplane problem, we introduce the notion of *optimal separating hyperplane*.

**2.2. Optimal Separating Hyperplane.** As we have introduced in the previous section, the problem of finding separating hyperplanes does not give a unique solution. This problem is tackled by Vapnik [**19**] that defines the optimal separating hyperplane as follows.

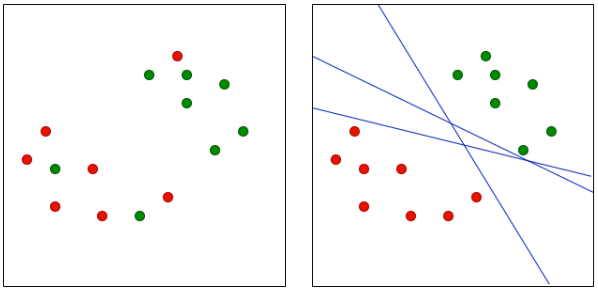

FIGURE 1. Left : Example of 2-dimensional data that are not linearly separable. Right : Example of 2-dimensional data that are separable. Three among the infinite number of separating hyperplanes are shown in blue

DEFINITION 4.1 (Optimal separating hyperplane). *The optimal separating hyperplane (or maximal margin hyperplane) separates the data without error and maximizes the distance to the closest point from either class. This distance is called the margin. In addition, the data points which are the closest to the optimal hyperplane are called support vectors.*

In order to find this optimal hyperplane, it is mandatory to be able to calculate the distance from each data point to the plane. The following proposition gives a convenient formula for this distance.

PROPOSITION 4.2 (Distance between a point and a hyperplane). *Let* $w, x \in \mathbb{R}^n$, $w_0 \in \mathbb{R}$ *and* $G : \mathbb{R}^n \to \mathbb{R}$, $x \mapsto w \cdot x + w_0$. *The signed distance from* $x$ *to the hyperplane* $L = \{x \in \mathbb{R}^n : G(x) = 0\}$ *is equal to* $d = \frac{G(x)}{||w||}$ *(where* $||w||$ *refers to the Euclidean norm of* $w$*).*

PROOF. The proof can be done in two ways. Either using a geometrical argument or using Lagrange multipliers. We will prove the proposition with the first option. Let $x_0 \in \mathbb{R}^n$, $x_0 \in L$. The distance between $x$ and $L$ is equal to the norm of the projection of the vector $x - x_0$ in the direction of $w$ (that is, a normal direction to the hyperplane $L$). A better visualization of the problem is sketched in Figure 2. We have, using natural projection formulas (see Figure 2) :

$$d = (x - x_0)\frac{w}{||w||} = \frac{w \cdot x - w \cdot x_0}{||w||}.$$

Since $x_0$ is chosen arbitrarily in $L$, we can take it such that $w \cdot x_0 = -w_0$. Then,

$$d = \frac{w \cdot x + w_0}{||w||} = \frac{G(x)}{||w||}.$$

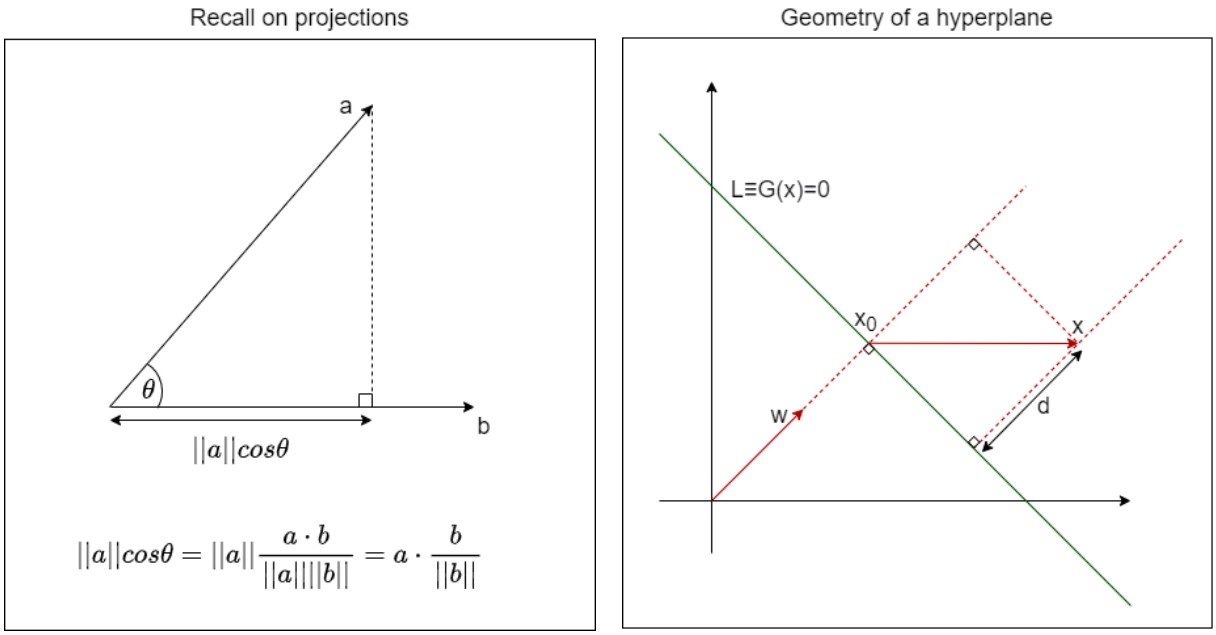


FIGURE 2. Left : Geometrical representation of the projection of a vector $a \in \mathbb{R}^2$ in the direction of a vector $b \in \mathbb{R}^2$. The formula below the graph is obtained by using the scalar product formula for the cosine. Right : 2-dimensional representation of a hyperplane $L$ (in green). The distance $d$ is the length of the projection of the vector $x$ in the direction of the vector $w$, which is a vector normal to the plane $L$. The representation is adapted from Hastie et al. [**18**]

□

Now, keeping the same notations as in the proposition, we can give a first natural formulation for the problem of finding the optimal separating hyperplane :

$$\max_{w,w_0} M$$

$$\text{subject to } y_i \frac{(x_i \cdot w + w_0)}{||w||} \geq M, i = 1, ...N.$$

The number $M \in \mathbb{R}$ to be maximized can be interpreted as the margin, in the sense of definition 4.1. The constraints ensure that the distance between each point and the separating hyperplane is greater than $M$. Indeed, the left sign of the inequality constraint is equal to the distance from a point to the hyperplane (it is the signed distance formula

of Proposition 4.2 multiplied by $y_i$, which gives the absolute distance). Now, recall that the hyperplane $L$ is defined with the help of the function $G : \mathbb{R}^n \to \mathbb{R}$, $x \mapsto w \cdot x + w_0$. The hyperplane remains the same if $G$ is multiplied by any positive constant. In addition, the quantity to be maximized $M$ is equal to $\frac{G(x_i)}{||w||}$ if $x_i$ is a support vector. This means that margin maximization is a well-posed problem only if we fix the scale, and for simplicity the scale is generally fixed to $G(x) = 1$ (as in the UCLouvain machine learning course LINFO2262 from Professor Dupont). At the end, we obtain the relation $M = \frac{1}{||w||}$. Hence, maximizing $M$ is equivalent to minimizing $||w||$, or equivalently, minimizing $\frac{1}{2}||w||^2$. The formulation for the problem of finding the optimal separating hyperplane becomes :

$$\min_{w,w_0} \frac{1}{2}||w||^2$$
$$\text{subject to } y_i(x_i \cdot w + w_0) \geq 1, i = 1, ...N.$$

Now, the problem consists in finding $w, w_0$ that maximizes the margin, which with our manipulations has a thickness of length $\frac{1}{||w||}$. This latter formulation is pleasant since it is a convex optimization problem (a quadratic function with linear inequality constraints) and plenty of softwares are able to solve it. We will not give the details of the resolution of this problem since it is not the one that will be used for the simulations. However, a solution is presented by Hastie et al. [**18**] and uses a Lagrangian combined with the Karush-Kuhn-Tucker optimality conditions, usual methods to solve optimization problems.

**2.3. Support Vector Classifier.** The previous section was a presentation of the separating hyperplane problem with the assumption that data can be perfectly separated by a plane. That is, it is possible to find a linear boundary that does not make any error at classifying the data. Of course, this assumption is rather unrealistic. In the following, we present a way to remove this assumption by relaxing the constraints of the problem. In comparison to the previous method which is called the *hard margin* optimal plane, the following is called the *soft margin* optimal plane. It is a linear version of the non-linear Support Vector Machine presented by Cortes and Vapnik [**20**] (see Section 2.4).

Suppose then that the two classes that compose the data overlap in $\mathbb{R}^l$. This means that the hard margin optimization problem does not have any solution anymore (because it is not possible to satisfy all the inequality constraints at the same time). A way to relax these conditions is still to maximize the margin but at the same time to allow some points

to be on the wrong side of the margin. Note that being "on the wrong side" of the margin does not necessarily mean to be misclassified (see Figure 3). Indeed, a point could be on the side of the hyperplane that corresponds to its class and still be on the wrong side of the margin, because the distance between the plane and the point is smaller than $M = \frac{1}{||w||}$. In practice this relaxation leads to the introduction of the so-called *slack variables* [**18**] $\{\xi_i\}_{i=1,...N}$.

DEFINITION 4.3 (Slack variables). *Slack variables* $\{\xi_i\}_{i=1,...N}$ *are useful in the context of finding optimal separating hyperplane. They are defined as the relative distance (with respect to the size of the margin) between a point and the hyperplane margin if that point is on the wrong side of the margin. Otherwise, slack variables are set to 0. Figure 3 gives the geometrical interpretation of these variables.*

Introducing slack variables into the optimal separating hyperplane problem results in the following optimization problem :

$$\min_{w,w_0} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i$$
$$\text{subject to } y_i(x_i \cdot w + w_0) \geq 1 - \xi_i, i = 1,...N.$$

where $C \in \mathbb{R}, C > 0$ can be interpreted as a regularization parameter that penalizes points that are not inside the margin [**18**]. It is chosen as a hyperparameter for the problem. If $C$ is small, it will allow a lot of points to be on the wrong side of the margin. Conversely, if $C \to \infty$, the problem tends to be the same as for the separable case and focus more attention on correctly classified points. This formulation is still a convex optimization problem, and can be solved numerically using standard optimization techniques [**18**]. The next section presents the generalization of the problem to non-linear boundaries, with the help of a mathematical reformulation known as the *kernel trick*.

**2.4. Support Vector Machines and Kernels.** The support vector classifier that is described in the previous section is designed to find a linear decision boundary in the input feature space $\mathbb{R}^l$. A way of allowing the boundary to be non-linear is presented by Cortes and Vapnik [**20**] and is known as *Support Vector Machine* classifier. The previous linear procedure is made more flexible by enlarging the feature space. This is done with the help of a mapping $\phi : \mathbb{R}^l \to \mathbb{R}^{l^*}$, with $l^* > l$. The search for a separating hyperplane is done in the enlarged space and transposes to a non-linear boundary in the original feature space. The optimization problem becomes :

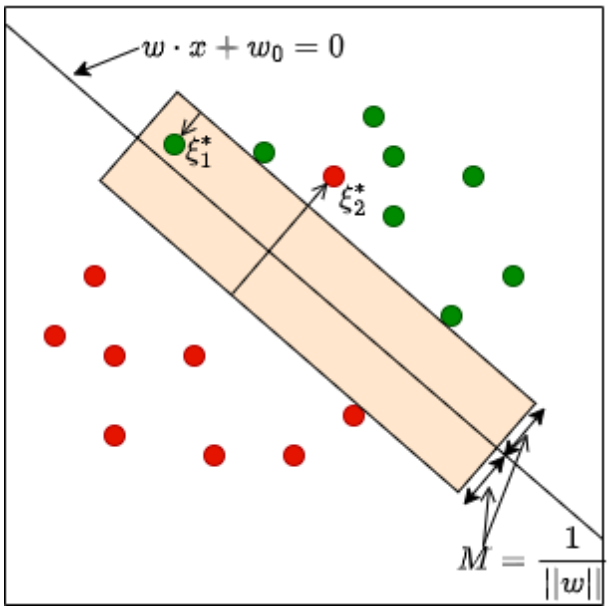


FIGURE 3. The figure depicts a 2-dimensional situation where the data is not linearly separable. The hyperplane of width $2M$ is in orange, traversed in the middle by the separating hyperplane. The points labeled $\xi_1^*$ and $\xi_2^*$ are on the wrong side of their margin by an amount $\xi_i^* = M\xi_i, i = 1,2$ and $\xi_i, i = 1,2$ are the slack variables. Points on the correct side of their margin have their corresponding slack variable equal to 0. The point labeled $\xi_1^*$ is an example of point which is on the wrong side of the margin but yet correctly classified by the hyperplane. The representation is adapted from Hastie et al [**18**].

$$\min_{w,w_0} \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} \xi_i$$

$$\text{subject to } y_i(\phi(x_i)\cdot w + w_0) \geq 1 - \xi_i, i = 1,...N.$$

Now the problem could seem to be computationally hard to solve since the dimensionality of the vector variable $w$ could be very high. To avoid this, the problem is solved in its dual form [**18**]

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \phi(x_i)\cdot\phi(x_{i'}) - \sum_{i=1}^{N} \alpha_i$$

$$\text{subject to } \sum_{i=1}^{N} \alpha_i y_i = 0,$$
$$0 \leq \alpha_i \leq C, i = 1, ..., N.$$

where $\alpha = (\alpha_1, ..., \alpha_N)$. Since the function $\phi$ still appears explicitly, the problem of having high dimensionality vectors in the equation is not solved. However, it is interesting to note that these problematic quantities only appear through scalar products. This observation is at the root of what is called the *kernel trick* [**20**]. The high dimensional function $\phi$ is replaced by a so-called kernel function $K : \mathbb{R}^l \times \mathbb{R}^l \to \mathbb{R}$, $(x_i, x_{i'}) \mapsto K(x_i, x_{i'}) = \phi(x_i) \cdot \phi(x_{i'})$. After this manipulation, the enlarged space is no more exploited and all operations can be performed in the smaller space. A simple example of kernel function is proposed in the following example.

EXAMPLE 4.4. Let $\phi : \mathbb{R}^2 \to \mathbb{R}^3$, $x = (x_1, x_2) \mapsto (x_1^2, x_2^2, \sqrt{2}x_1x_2)$. The corresponding kernel function $K$ is defined as $K : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$, $(x_i, x_{i'}) \mapsto x_{i_1}^2 x_{i'_1}^2$. Indeed, it is easy to verify that for $x_i \in \mathbb{R}^2$, $x_{i'} \in \mathbb{R}^2$, we have $K(x_i, x_{i'}) = \phi(x_i) \cdot \phi(x_{i'})$.

The kernel that we will use during the simulations is called the radial basis function (rbf) kernel and is defined as

$$K(x_i, x_{i'}) = exp(-\gamma ||x_i - x_{i'}||^2)$$

with $\gamma \in \mathbb{R}$, $\gamma > 0$ to be specified.
Finally, we give the formulation of the problem exactly as it will be expressed during the simulations, by giving the details of the software LIBSVM [**21**] that we will use.

**2.5. LIBSVM: A Library for SVM.** LIBSVM is a C++ library developed by Chang and Lin [**21**] which is used by the extremely popular python package *sklearn* [**27**] to build Support Vector Machines. The (dual) optimization problem solved by the software is of the form (in matrix notation)

$$\min_{\alpha} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha$$

$$\text{subject to } y^T \alpha = 0,$$
$$0 \leq \alpha_i \leq C, i = 1, ..., N$$

where $e = (1, ..., 1)^T$ is the vector of all ones, $Q$ is an $N$ by $N$ matrix, $Q_{ij} = y_i y_j K(x_i, x_j)$, and $K$ is the kernel function associated to some mapping $\phi$. The software solves this convex

optimization problem and uses the obtained vector $\alpha$ to reconstruct $w$, the solution of the primal problem. Using the primal-dual relationship, the optimal $w$ is

$$w = \sum_{i=1}^{N} y_i \alpha_i \phi(x_i)$$

.

The decision function can then be expressed alternatively with primal or dual variables [**18**]. This decision function (noted $f_{\text{dec}}$) can be expressed as

$$f_{\text{dec}}(x) = \text{sign}(w \cdot \phi(x) + w_0) = \text{sign}\Big( \sum_{i=1}^{N} y_i \alpha_i K(x_i, x) + w_0 \Big).$$

The parameter $w_0$ is computed using the equality $y_i(w \cdot \phi(x_i) + w_0) = 1$, with $x_i$ a support vector (support vectors satisfy the previous equation since they are situated exactly on the decision boundary). LIBSVM will be the software used for the simulation part.

This ends the recall on Support Vector Machines.

## 3. Simulations

**3.1. Objectives.** The objective of the following simulations is to measure the impact of imputation methods on SVMs. For this we will compare different missing data proportions and different missing data mechanisms. Each time we will compare the accuracy obtained with the following imputation methods : mean imputation, regression imputation, stochastic regression imputation and multiple imputation. The SVM built on the complete datasets will be used as a benchmark during the analysis.

**3.2. Data Generation.** The data generation procedure is done with the help of the python function *make_classification()* from the package *sklearn* [**27**]. This function is commonly employed to generate a random classification problem. In a nutshell, the method creates clusters of points normally distributed, clusters being centered on the vertices of a hypercube. For each cluster, informative features are drawn independently from $N(0,1)$ and then randomly linearly combined within each cluster in order to add covariance. An example of such created classification problem is shown in Figure 4.

Now, we explain in details how the generation is done and the particular parameters set for the simulations. Our data matrix $X$ will be composed of $N = 1000$ samples of 10 variables. To each sample is associated the number of the class, that is $-1$ or 1 since

the simulations will work with 2 distinct classes. Class labels are brought together in a vector $Y$ of size 1000. The 10 variables will not all be of the same form. The first 8 are the "informative variables" and the last 2 are "redundant variables". It will become more clear with the following additional explanations.

Initially, the 8 informative features are drawn from a standard normal. The first 500 sample are assigned to the first class and the last 500 to the second, in a balanced design. At this point of the algorithm, the matrix $X$ is of size $1000 \times 8$ and the label vector $Y$ is of size 1000. For each class, a squared matrix $A$ of size $8 \times 8$ is drawn with elements coming from a uniform distribution between $-1$ and 1. The matrix $X$ is divided in two parts with respect to the sample classes. Each part (of size $500 \times 8$) is multiplied by a different matrix, constructed as $A$. This introduces random variance-covariance to the data, and is different for each class. The last two variables are obtained as a random linear combination of the first 8.

Finally, the two clusters are translated in order for their respective centers to be at a distance of 2. The very last step consists in flipping the class of 1% of the samples, in order to make the classification task more complicated. The latter operation is operated on the vector $Y$. Data are shuffled at the end, in order not to have first all the samples of the first class followed by all the samples of the second class. The same shuffling is applied to $X$ and $Y$.

As it will be explained in Section 3.4, a dataset will be built with the above procedure at each iteration of the simulations. However, these datasets are still complete at the moment. Next section will detail how the missing data is generated to mask parts of this data.

**3.3. Data Masking.** The previous section was about the generation of a complete dataset. This section is about the "Data Masking" part. That is, obtain datasets with missing data from the complete data. The missing data is generated following the three classical mechanisms that produce missing data : MCAR, MAR and MNAR (see Chapter 1, Section 3).

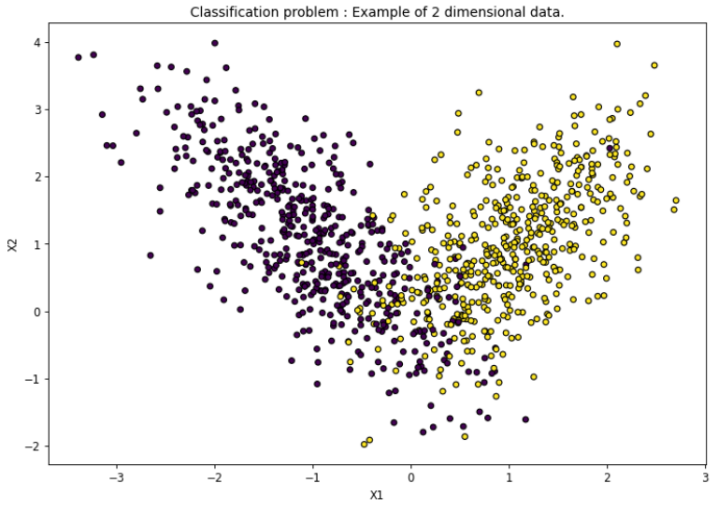


FIGURE 4. Figure shows 2 classes of 500 points each in a two dimensional space. The first one (in purple) is centered at the point (-1,1). The second (in yellow) is centered at the point (1,1). Each cluster follows a Gaussian distribution around their centers and variance-covariance in each cluster is randomly obtained. To add noise, one percent of the points has its class flipped.

The first mechanism, MCAR, is really simple to implement. If the goal is to obtain MCAR data with missing probability $\pi$, every observed value is set to missing with probability $\pi$. This is done concretely in python by generating a matrix with the same dimensions of the initial dataset (in our case $1000 \times 10$) and elements randomly drawn from a Bernoulli distribution with parameter $\pi$. If an element of this matrix is equal to 1, than the corresponding element of the complete dataset is set to missing.

The second mechanism, MAR, is less straightforward to implement. The probability for a value of a given variable to be missing needs to depend on the value of another variable. The implementation is done in the following way based on two probabilities $\pi_1$ and $\pi_2$. For a specific row, if the value of the first variable is less than the median for this first variable, the probability to be missing for another variable of the same row is equal to $\pi_1$. On the contrary, if the value of the first variable is superior to the median, the probability for another variable to be missing is set to $\pi_2$. The first variable can also contain missing data, with probability $\pi_3 = (\pi_1 + \pi_2)/2$. Formally, with "med" being the median of the first variable :

$$P(X_{ij} \text{ is missing}) = \begin{cases} \pi_1 & \text{if } X_{i1} \leq \text{med}, \\ \pi_2 & \text{if } X_{i1} > \text{med} \end{cases}$$

for $i = 1, ..., 1000$ and $j = 2, ..., 10$. In addition,

$$P(X_{i1} \text{ is missing}) = \pi_3$$

for $i = 1, ..., 1000$. As it will be explained is Section 3.4, it is simple to obtain an overall probability of being missing $\pi$. The first probability is simply set to $\pi_1 = \pi - \pi/2$ and the second to $\pi_2 = \pi + \pi/2$. In this way and taking into account how the MAR dataset is constructed, the overall missing probability is $\pi$, as expected. This is important since it will enable us to compare easily data with the same amount of missing data, even if the missing data mechanisms are different.

The third and last missing data mechanism is MNAR. To generate this mechanism it is needed that the probability to be missing depends on the true value of the data. For this, we compare each value of the originally generated dataset to the median of the data. If a value is inferior to the median, it is set to missing with probability $\pi_1$. Otherwise, it is set to missing with probability $\pi_2$. As for the MAR mechanism, it is easy to obtain a specific overall proportion $\pi$ of missing data. The first probability is taken as $\pi_1 = \pi - \pi/2$ and the second as $\pi_2 = \pi + \pi/2$. All missing data mechanisms can then be compared with the same total proportion of missing data.

**3.4. Simulation Plan.** Below are explained the details of the simulations. We start by describing each step of one iteration of the main loop.

The first step is to generate the data. As explained is Section 3.2, a generated dataset $X$ is composed of 1000 samples of 10 variables (8 informative variables and 2 redundant variables) and is associated with a label vector $Y$ of length 1000. Three datasets are created from the complete one : MCAR, MAR and MNAR datasets (see Section 3.3 for details). The choice of the proportion of missing data in each dataset will be detailed later.

At this point we have 4 different datasets : the complete one and the three containing missing data. Each of them is split randomly in the same way into a train and a test

set, the classical decomposition used in the purpose of evaluating machine learning algorithms. The test set is composed of 200 samples and their corresponding labels.

Data imputation comes right after the train/test split. It is important that these steps are performed in that order. Indeed, if imputation is done before the split, information coming from the train set will be transferred to the test set because the imputation models will use all available information. However, the test set absolutely needs to be completely independent of the train set. Of course, imputation does not concern the complete dataset. Imputation is done using four different methods : mean imputation, regression imputation, stochastic regression imputation and multiple imputation (detailed respectively in Chapter 2, Section 3, 4, 5 and Chapter 3). The latter is performed using a number of imputations $K = 5$ and $K = 15$. Imputed values are obtained with the help of the R package *mice* [**28**]. For multiple imputation, it applies the MICE algorithm presented in Chapter 3, Section 7.

All datasets (both train and test sets) are now completed, and in different ways depending on the imputation method. Now the task is to build a Support Vector Machine on each train set, and assess the performance on the corresponding test sets. The construction of the Support Vector Machine is done using the package *sklearn* [**27**] that relies on the LIBSVM algorithm (see Section 2.5 for the details). When the SVM is built, it predicts a class for each test sample. The performance of the model is measured with the mean accuracy, that is the proportion of correctly classified samples

$$\text{Acc} = \frac{1}{200}\sum_{i=1}^{200}\frac{|Y_i + \hat{Y}_i|}{2}$$

where $Y_i$ is the true label of the $i^{th}$ sample of the test set and $\hat{Y}_i$ is the predicted label. Since labels are set to $-1$ or 1, the term inside the sum is equal to 1 if the sample is correctly classified and 0 otherwise. The accuracy for each dataset is then saved, and this concludes the steps for one iteration of the simulation.

The simulations are repeated for an overall missing probability $\pi$ equal to 0.1, 0.2, 0.3 and 0.4. For each of them, the performance of the SVM for the mechanisms and the methods cited earlier is measured on 1000 different randomly generated datasets. Finally, we

should mention that simulations are made perfectly reproducible by the use of a random seed.

## 4. Results

This section presents an analysis of the results obtained through the simulations. We start with an informal look at the boxplots displayed in Figures 5, 6 and 7 (in the Appendix). At first sight, is seems that all methods leads to very good results when the overall missing probability is small, that is equal to 0.1. This seems to be true for all missing data mechanisms. Note that, in this section, a "good" result for a method means that the SVM built on a dataset filled with this specific method has an accuracy that is similar to the accuracy of the SVM built on the complete dataset. It seems clear that accuracy decreases while the proportion of missing data increases. It is not so clear however if the performance is better or worse for some missing mechanism in comparison to another.

Below is given a deeper statistical analysis of the performances. We start by performing F-tests in R to compare the performances of the different methods, within the same couple of missing mechanism and missing proportion. We perform this for every couple of missing mechanism and missing proportion. At a confidence level of 5% (which will be the confidence level for all following statistical tests), we reject the hypothesis that all imputation methods lead to the same mean accuracy (the test gives the same conclusion for all mechanism/proportion pairs).

Now that we have rejected the general hypothesis that all methods have the same accuracy, we perform several Tukey tests to compare methods 2 by 2. The results are presented from Table 1 to 12 in the Appendix. We start with the MCAR case with a missing proportion of 0.1. For all imputation methods, we can reject the hypothesis that a specific imputation method gives an accuracy equal to the one of the complete dataset. That is, none of the imputation methods can compensate the loss of information due to missing data, even when they are present in small proportions. This fact is true for each mechanism and each of the four missing proportions and so it will not be repeated in the following. However, we see that this accuracy difference is not huge at all and varies between 0.004 and 0.027 (by looking at the 95% confidence intervals for the accuracy difference). We proceed and conclude that mean imputation gives also a result that is different from every other method (always by looking at the Tukey test), but this time it

is a lower result than the others.

We continue with MCAR data. For missing proportions 0.2 and 0.4, the accuracy obtained with regression imputation is better than with every other method and the one obtained with mean imputation is lower than any other. We cannot reject the equality of accuracy for any 2 by 2 combination of the three following methods : MI (with $K = 5$), MI (with $K = 15$) and stochastic regression imputation. The same conclusion can be drawn for a missing proportion of 0.3 except that mean imputation is not statistically different from these latter three methods anymore. Unsurprisingly, the accuracy difference with respect to the complete dataset increases with the proportion of missing data. For example, for regression imputation, this difference grows from 0.007 to 0.101. However, we note that the latter is still a very good result considering that 40% of the data is missing.

For MAR and MNAR, the analysis is essentially the same as for MCAR. The only difference is that the accuracy difference with the complete data is slightly bigger. Surprisingly however, MNAR gives slightly better results than MAR.

For all settings then, regression imputation seems to be the best imputation method while mean imputation seems to be the worst. Stochastic regression and multiple imputation (with a different number of imputed values) yield the same results, and this is what expected since they are extremely similar. We should note that these relative performances of the different methods is not what was expected from the statistical part of this work (with Multiple Imputation that should have been the better method). This can be explained simply by the fact that the theoretical results obtained previously are what to expect for direct inference from the imputed dataset. However, what is done in these simulations is not to produce inferences directly on the imputed dataset but to build a SVM on top of it and so the theoretical analysis may not hold.

Overall, support vector machines seem robust to the presence of missing values in the predictors, at least for the type of data and missing mechanisms implemented in these simulations. This robustness result was also previously obtained by Stewart et al. [**29**] although using different imputation methods and different missing mechanisms as those treated in this work. Thus, this reinforces the fact that treating missing data with imputation methods in the purpose of building SVM on it is a very good method. The reason

for this seems to be that SVM essentially depends on support vectors. Indeed, only missingness among the support vectors and observations that would be miss-classified have the potential to impact the SVM solution [**29**].

# Conclusion

In conclusion to this work about missing data, we can say that it is a very broad topic. We have presented a bunch of imputation methods in Chapters 2 and 3 but there exists plenty of them. For example, Molenberghs and Kenward [**1**] introduce direct likelihood method and weighted estimating equations. Another imputation method that is gaining in popularity is based on nearest neighbours algorithm [**30**].

Since all these methods exist and are available in more and more softwares, we suggest to completely avoid the use of Complete Case Analysis and Last Observation Carried Forward, for the reasons stated in Chapter 2. Other basic imputation methods should be used with caution.

A very important result obtained in Chapter 4 is that SVMs seem really robust to the presence of missing data in the predictors, at least with the simulations settings of this work. This result could be an argument in favour of SVMs for classification purposes. However, a good extension to this work could be to reproduce these experiments with other supervised machine learning algorithms for classification. For example, decision trees, nearest neighbours, etc. In this way we could compare formally these methods and see if some are more robust than others.

Another important result is the fact that the performance of an imputation method depends on the purpose of the work that has to be done after imputation. For example, the first part (the statistical part) shows clearly that multiple imputation is the best method between those presented for inference purposes. However, with our simulations settings in the simulations part (the data science part), we show that regression imputation performs better. Therefore, this work could be extended by a more exhaustive search about the best methods for each popular machine learning classification algorithm.

Another way of linking missing data with machine learning methods is to impute data directly using machine learning methods. This is done for example by Gupta and Lam [**7**]. They try to do the imputation step using neural networks, but this technique has not shown better results than classical imputation methods so far. This is also still an open field in the study of missing data.

# Bibliography


[1] G. Molenberghs and M. G. Kenward, *Missing Data in Clinical Studies*, Statistics in Practice, Wiley, Chichester, 2007. ↑1, 3, 4, 5, 6, 7, 9, 11, 12, 21, 22, 51

[2] D. B. Rubin, *Inference and missing data*, Biometrika **63** (1976), 581–592. ↑5

[3] X. Zhu, *Comparison of Four Methods for Handing Missing Data in Longitudinal Data Analysis through a Simulation Study*, Open Journal of Statistics **4** (2014), 933–944. ↑12

[4] G. Molenberghs and E. Lesaffre, *Missing Datas*, Wiley (2007), 1–15. ↑6, 7

[5] R. J. A. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, Journal of Educational Statistics **16** (2002), 150–155. ↑1, 3, 4, 6, 9, 13, 18, 19, 30

[6] S. van Buuren, *Flexible Imputation of Missing Data*, Interdisciplinary Statistics, Chapman&Hall/CRC, Boca Raton, 2018. ↑6, 12, 13, 15, 16, 17, 25, 28, 31, 32, 33

[7] A. Gupta and M. S. Lam, *Estimating Missing Values Using Neural Networks*, The Journal of the Operational Research Society **47** (1996), 229–238. ↑52

[8] K. H. Li, T. E. Raghunathan, and D. B. Rubin, *Large-Sample Significance Levels from Multiply Imputed Data Using Moment-Based Statistics and an F Reference Distribution*, Journal of the American Statistical Association **86** (1991), 1065–1073. ↑27

[9] D. B. Rubin, *Multiple Imputation for Nonresponse in Surveys*, JOHN WILEY & SONS, New York, 1987. ↑22, 23, 24

[10] J. L. Schafer, *Analysis of Incomplete Multivariate Data.*, Chapman & Hall, London, 1997. ↑27, 29, 30

[11] A. Agresti, *Categorical Data Analysis.*, John Wiley & Sons, New York, 1990. ↑28

[12] O. Harel, *The Estimation of $R^2$ and Adjusted $R^2$ in Incomplete Data Sets Using Multiple Imputation.*, Journal of Applied Statistics **36** (2009), 1109–1118. ↑28

[13] M. G. Kenward and G. Molenberghs, *Last Observation Carried Forward: a crystal ball?*, Journal of Biopharmaceutical Statistics **19** (2009), 872–888. ↑12

[14] G. Verbeke and G. Molenberghs, *Linear Mixed Models in Practice: A SAS-Oriented Approach*, Lecture Notes in Statistics, Springer, New York, 1997. ↑12

[15] A. P. Dempster and D. B. Rubin, *"Introduction." In Incomplete Data in Sample Surveys*, Academic Press, New York, 1983. ↑9

[16] S. F. Buck, *A method of estimation of missing values in multivariate data suitable for use with an electronic computer*, Journal of the Royal Statistical Society **22** (1960), 302–306. ↑15

[17] C. K. Enders, *Applied Missing Data Analysis*, Lecture Notes in Statistics, Guilford Press, New York, 2010. ↑16, 17

[18] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer Series in Statistics, Springer, New York, 2009. ↑36, 38, 39, 40, 41, 43

[19] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Information Science and Statistics, Springer, New York, 2000. ↑36

[20] C. Cortes and V. Vapnik, *Support-vector networks*, Machine Learning **20** (1995), 273-–297. ↑39, 40, 42

[21] C.-C. Chang and C.-J. Lin, *LIBSVM: A Library for Support Vector Machines*, ACM Transactions on Intelligent Systems and Technology **2** (2021), 1–27. ↑42

[22] J. L. Schafer and N. Schenker, *Inference with Imputed Conditional Means.*, Journal of the American Statistical Association **95** (2000), 144–54. ↑19

[23] J. H. Goodnight, *A Tutorial on the SWEEP Operator.*, The American Statistician **33(3)** (1979), 149–158. ↑29

[24] N. J. Horton, S. R. Lipsitz, and M. Parzen, *A Potential for Bias When Rounding in Multiple Imputation.*, The American Statistician **57(4)** (2003), 229–232. ↑30

[25] S. Van Buuren and C. G. M. Groothuis-Oudshoorn, *Flexible Multivariate Imputation by MICE.*, Vol. PG/VGZ/99.054., TNO Prevention and Health, Leiden, 1999. ↑31

[26] G. O. Roberts, *Markov chain concepts related to sampling algorithms.*, Markov chain Monte Carlo in practice **57** (1996), 45–58. ↑32

[27] 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** (2011), 2825–2830. ↑42, 43, 47

[28] S. Van Buuren and K. Groothuis-Oudshoorn, *mice: Multivariate Imputation by Chained Equations in R.*, Journal of Statistical Software **45** (2011), 1–67. ↑47

[29] T. G. Stewart, D. Zeng, and M. C. Wu, *Constructing support vector machines with missing data.*, WIREs Computational Statistics. **10(4)** (2018). ↑35, 49, 50

[30] L. Beretta and A. Santaniello, *Nearest neighbor imputation algorithms: a critical evaluation.*, BMC Medical Informatics and Decision Making **16** (2016), 74. ↑51

# Appendix

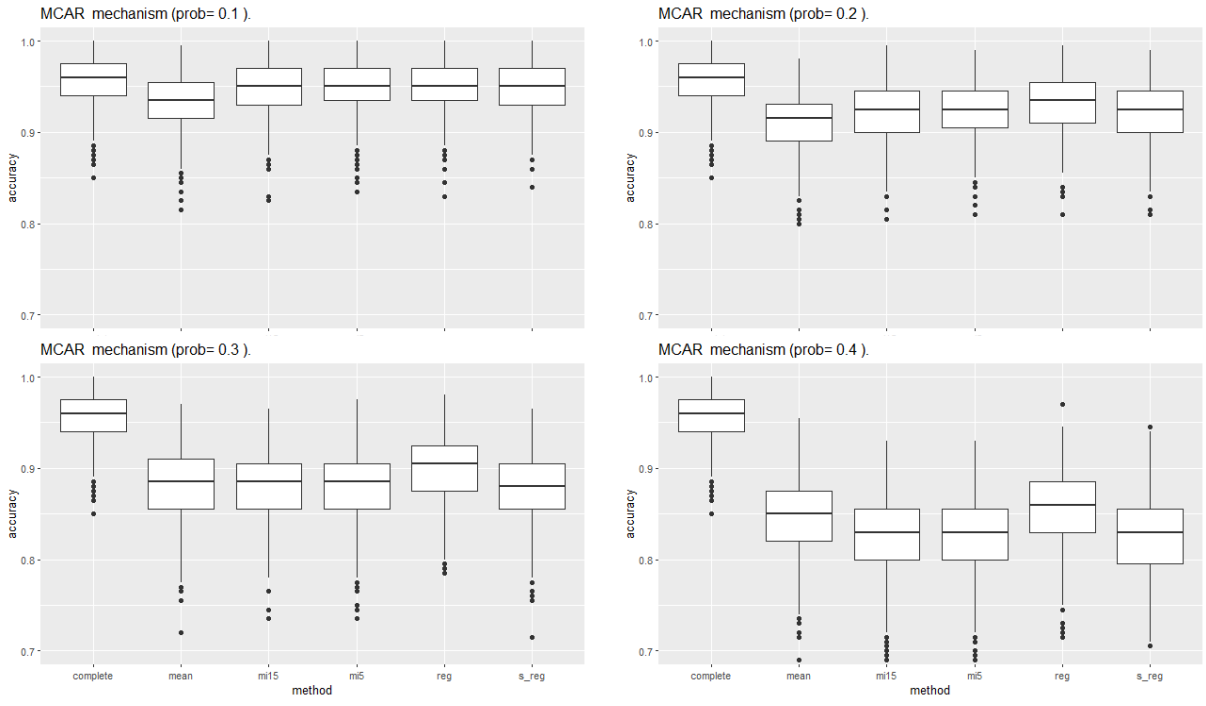


FIGURE 5. Boxplots comparison for MCAR data and missing probability equal to 0.1, 0.2, 0.3 and 0.4 (from left to right and top to bottom). "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. The $Y$ axis goes from 0.7 to 1.

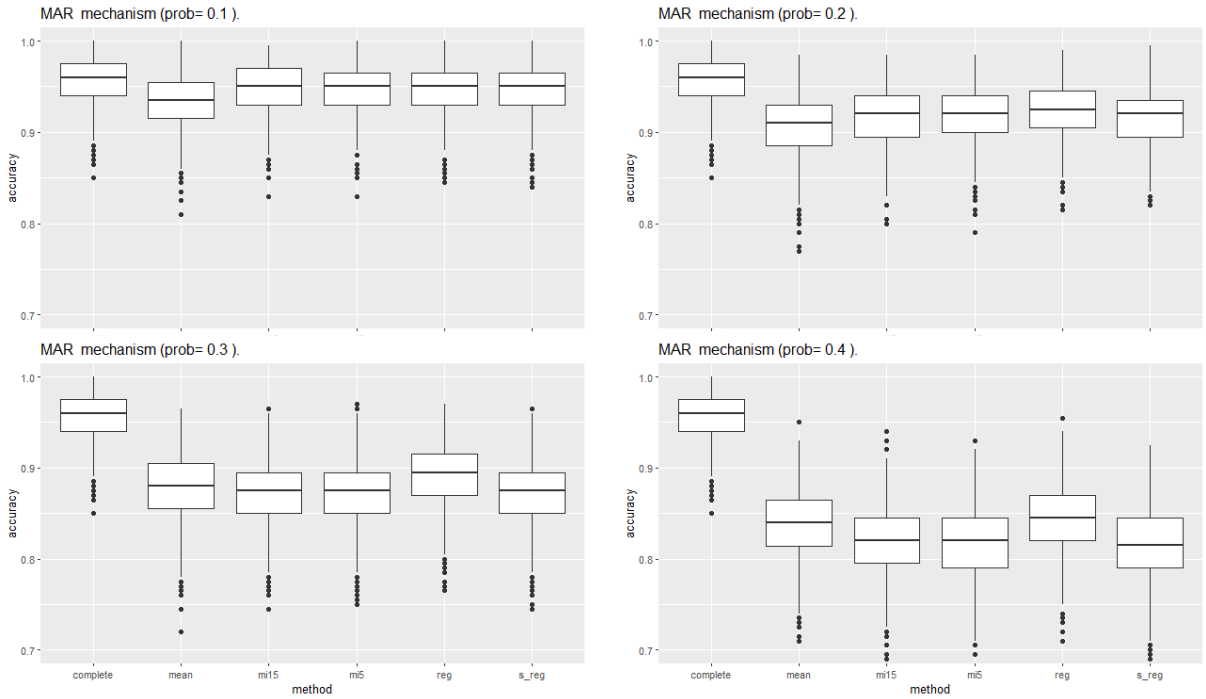


FIGURE 6. Boxplots comparison for MAR data and missing probability equal to 0.1, 0.2, 0.3 and 0.4 (from left to right and top to bottom). "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. The $Y$ axis goes from 0.7 to 1.

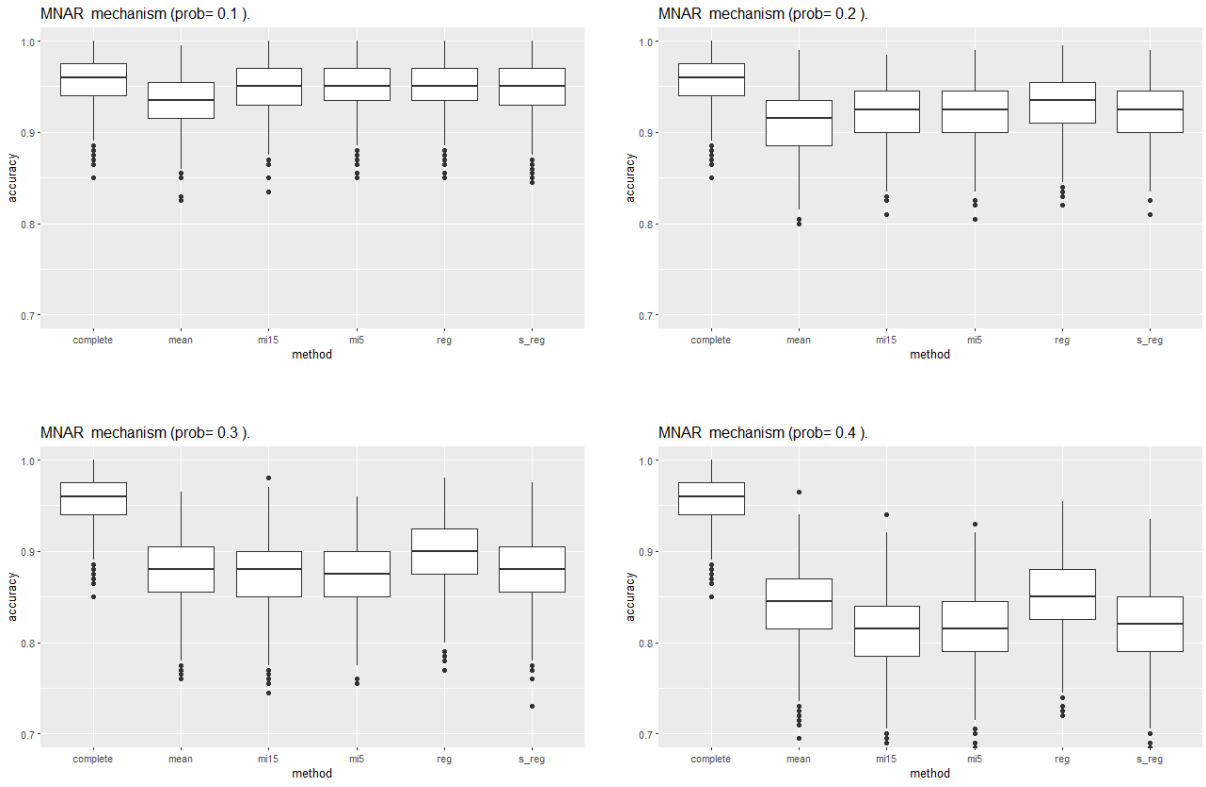


FIGURE 7. Boxplots comparison for MNAR data and missing probability equal to 0.1, 0.2, 0.3 and 0.4 (from left to right and top to bottom). "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. The $Y$ axis goes from 0.7 to 1.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.024 | -0.027 | -0.02 | 0 |
| mi15-complete | -0.008 | -0.012 | -0.005 | 0 |
| mi5-complete | -0.008 | -0.012 | -0.005 | 0 |
| reg-complete | -0.007 | -0.011 | -0.004 | 0 |
| s_reg-complete | -0.009 | -0.012 | -0.005 | 0 |
| mi15-mean | 0.015 | 0.012 | 0.019 | 0 |
| mi5-mean | 0.015 | 0.012 | 0.019 | 0 |
| reg-mean | 0.016 | 0.013 | 0.02 | 0 |
| s_reg-mean | 0.015 | 0.012 | 0.019 | 0 |
| mi5-mi15 | 0 | -0.003 | 0.004 | 1 |
| reg-mi15 | 0.001 | -0.002 | 0.004 | 0.97 |
| s_reg-mi15 | 0 | -0.004 | 0.003 | 1 |
| reg-mi5 | 0.001 | -0.003 | 0.004 | 0.983 |
| s_reg-mi5 | 0 | -0.004 | 0.003 | 1 |
| s_reg-reg | -0.001 | -0.004 | 0.002 | 0.955 |

TABLE 1. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MCAR and missing proportion is equal to 0.1. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.047 | -0.051 | -0.043 | 0 |
| mi15-complete | -0.034 | -0.038 | -0.031 | 0 |
| mi5-complete | -0.034 | -0.038 | -0.031 | 0 |
| reg-complete | -0.026 | -0.03 | -0.022 | 0 |
| s_reg-complete | -0.035 | -0.039 | -0.031 | 0 |
| mi15-mean | 0.013 | 0.009 | 0.017 | 0 |
| mi5-mean | 0.013 | 0.009 | 0.017 | 0 |
| reg-mean | 0.021 | 0.017 | 0.025 | 0 |
| s_reg-mean | 0.012 | 0.009 | 0.016 | 0 |
| mi5-mi15 | 0 | -0.004 | 0.004 | 1 |
| reg-mi15 | 0.008 | 0.004 | 0.012 | 0 |
| s_reg-mi15 | 0 | -0.004 | 0.003 | 1 |
| reg-mi5 | 0.008 | 0.004 | 0.012 | 0 |
| s_reg-mi5 | 0 | -0.004 | 0.004 | 1 |
| s_reg-reg | -0.009 | -0.012 | -0.005 | 0 |

TABLE 2. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MCAR and missing proportion is equal to 0.2. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.077 | -0.082 | -0.073 | 0 |
| mi15-complete | -0.076 | -0.081 | -0.072 | 0 |
| mi5-complete | -0.077 | -0.082 | -0.073 | 0 |
| reg-complete | -0.058 | -0.062 | -0.054 | 0 |
| s_reg-complete | -0.078 | -0.082 | -0.073 | 0 |
| mi15-mean | 0.001 | -0.003 | 0.006 | 0.983 |
| mi5-mean | 0 | -0.005 | 0.004 | 1 |
| reg-mean | 0.019 | 0.015 | 0.024 | 0 |
| s_reg-mean | -0.001 | -0.005 | 0.004 | 0.999 |
| mi5-mi15 | -0.001 | -0.006 | 0.003 | 0.965 |
| reg-mi15 | 0.018 | 0.014 | 0.023 | 0 |
| s_reg-mi15 | -0.002 | -0.006 | 0.003 | 0.895 |
| reg-mi5 | 0.019 | 0.015 | 0.024 | 0 |
| s_reg-mi5 | 0 | -0.005 | 0.004 | 1 |
| s_reg-reg | -0.02 | -0.024 | -0.015 | 0 |

TABLE 3. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MCAR and missing proportion is equal to 0.3. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.109 | -0.114 | -0.104 | 0 |
| mi15-complete | -0.132 | -0.137 | -0.127 | 0 |
| mi5-complete | -0.131 | -0.136 | -0.126 | 0 |
| reg-complete | -0.101 | -0.106 | -0.097 | 0 |
| s_reg-complete | -0.133 | -0.138 | -0.128 | 0 |
| mi15-mean | -0.023 | -0.028 | -0.018 | 0 |
| mi5-mean | -0.022 | -0.027 | -0.017 | 0 |
| reg-mean | 0.007 | 0.002 | 0.012 | 0 |
| s_reg-mean | -0.024 | -0.029 | -0.019 | 0 |
| mi5-mi15 | 0.001 | -0.004 | 0.006 | 0.994 |
| reg-mi15 | 0.03 | 0.025 | 0.035 | 0 |
| s_reg-mi15 | -0.001 | -0.006 | 0.004 | 0.997 |
| reg-mi5 | 0.029 | 0.024 | 0.034 | 0 |
| s_reg-mi5 | -0.002 | -0.007 | 0.003 | 0.913 |
| s_reg-reg | -0.031 | -0.036 | -0.026 | 0 |

TABLE 4. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MCAR and missing proportion is equal to 0.4. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.024 | -0.028 | -0.021 | 0 |
| mi15-complete | -0.01 | -0.013 | -0.007 | 0 |
| mi5-complete | -0.01 | -0.014 | -0.007 | 0 |
| reg-complete | -0.009 | -0.013 | -0.006 | 0 |
| s_reg-complete | -0.011 | -0.014 | -0.007 | 0 |
| mi15-mean | 0.014 | 0.011 | 0.018 | 0 |
| mi5-mean | 0.014 | 0.01 | 0.017 | 0 |
| reg-mean | 0.015 | 0.011 | 0.018 | 0 |
| s_reg-mean | 0.013 | 0.01 | 0.017 | 0 |
| mi5-mi15 | 0 | -0.004 | 0.003 | 1 |
| reg-mi15 | 0.001 | -0.003 | 0.004 | 0.996 |
| s_reg-mi15 | -0.001 | -0.004 | 0.003 | 0.992 |
| reg-mi5 | 0.001 | -0.002 | 0.004 | 0.967 |
| s_reg-mi5 | 0 | -0.004 | 0.003 | 1 |
| s_reg-reg | -0.001 | -0.005 | 0.002 | 0.887 |

TABLE 5. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MAR and missing proportion is equal to 0.1. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.049 | -0.053 | -0.045 | 0 |
| mi15-complete | -0.04 | -0.044 | -0.036 | 0 |
| mi5-complete | -0.04 | -0.044 | -0.036 | 0 |
| reg-complete | -0.032 | -0.036 | -0.028 | 0 |
| s_reg-complete | -0.041 | -0.045 | -0.037 | 0 |
| mi15-mean | 0.009 | 0.005 | 0.013 | 0 |
| mi5-mean | 0.009 | 0.005 | 0.013 | 0 |
| reg-mean | 0.017 | 0.013 | 0.021 | 0 |
| s_reg-mean | 0.008 | 0.004 | 0.012 | 0 |
| mi5-mi15 | 0 | -0.004 | 0.004 | 1 |
| reg-mi15 | 0.008 | 0.004 | 0.012 | 0 |
| s_reg-mi15 | -0.001 | -0.005 | 0.003 | 0.988 |
| reg-mi5 | 0.008 | 0.004 | 0.012 | 0 |
| s_reg-mi5 | -0.001 | -0.005 | 0.003 | 0.968 |
| s_reg-reg | -0.009 | -0.013 | -0.005 | 0 |

TABLE 6. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MAR and missing proportion is equal to 0.2. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.08 | -0.084 | -0.075 | 0 |
| mi15-complete | -0.085 | -0.089 | -0.081 | 0 |
| mi5-complete | -0.085 | -0.089 | -0.08 | 0 |
| reg-complete | -0.067 | -0.072 | -0.063 | 0 |
| s_reg-complete | -0.086 | -0.09 | -0.081 | 0 |
| mi15-mean | -0.006 | -0.01 | -0.001 | 0.005 |
| mi5-mean | -0.005 | -0.009 | -0.001 | 0.017 |
| reg-mean | 0.012 | 0.008 | 0.017 | 0 |
| s_reg-mean | -0.006 | -0.011 | -0.002 | 0.001 |
| mi5-mi15 | 0.001 | -0.004 | 0.005 | 0.999 |
| reg-mi15 | 0.018 | 0.013 | 0.022 | 0 |
| s_reg-mi15 | -0.001 | -0.005 | 0.004 | 0.998 |
| reg-mi5 | 0.017 | 0.013 | 0.022 | 0 |
| s_reg-mi5 | -0.001 | -0.006 | 0.003 | 0.967 |
| s_reg-reg | -0.018 | -0.023 | -0.014 | 0 |

TABLE 7. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MAR and missing proportion is equal to 0.3. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.118 | -0.123 | -0.113 | 0 |
| mi15-complete | -0.137 | -0.142 | -0.133 | 0 |
| mi5-complete | -0.138 | -0.143 | -0.134 | 0 |
| reg-complete | -0.113 | -0.118 | -0.109 | 0 |
| s_reg-complete | -0.14 | -0.145 | -0.135 | 0 |
| mi15-mean | -0.019 | -0.024 | -0.014 | 0 |
| mi5-mean | -0.02 | -0.025 | -0.016 | 0 |
| reg-mean | 0.005 | 0 | 0.01 | 0.05 |
| s_reg-mean | -0.022 | -0.027 | -0.017 | 0 |
| mi5-mi15 | -0.001 | -0.006 | 0.004 | 0.985 |
| reg-mi15 | 0.024 | 0.019 | 0.029 | 0 |
| s_reg-mi15 | -0.003 | -0.008 | 0.002 | 0.559 |
| reg-mi5 | 0.025 | 0.02 | 0.03 | 0 |
| s_reg-mi5 | -0.002 | -0.006 | 0.003 | 0.924 |
| s_reg-reg | -0.027 | -0.032 | -0.022 | 0 |

TABLE 8. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MAR and missing proportion is equal to 0.4. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.023 | -0.027 | -0.02 | 0 |
| mi15-complete | -0.008 | -0.012 | -0.005 | 0 |
| mi5-complete | -0.008 | -0.011 | -0.004 | 0 |
| reg-complete | -0.007 | -0.011 | -0.004 | 0 |
| s_reg-complete | -0.009 | -0.012 | -0.005 | 0 |
| mi15-mean | 0.015 | 0.011 | 0.018 | 0 |
| mi5-mean | 0.015 | 0.012 | 0.019 | 0 |
| reg-mean | 0.016 | 0.012 | 0.019 | 0 |
| s_reg-mean | 0.015 | 0.011 | 0.018 | 0 |
| mi5-mi15 | 0 | -0.003 | 0.004 | 0.999 |
| reg-mi15 | 0.001 | -0.002 | 0.004 | 0.945 |
| s_reg-mi15 | 0 | -0.004 | 0.003 | 1 |
| reg-mi5 | 0.001 | -0.003 | 0.004 | 0.995 |
| s_reg-mi5 | -0.001 | -0.004 | 0.003 | 0.994 |
| s_reg-reg | -0.001 | -0.005 | 0.002 | 0.889 |

TABLE 9. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MNAR and missing proportion is equal to 0.1. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.048 | -0.052 | -0.044 | 0 |
| mi15-complete | -0.036 | -0.039 | -0.032 | 0 |
| mi5-complete | -0.035 | -0.039 | -0.031 | 0 |
| reg-complete | -0.026 | -0.03 | -0.022 | 0 |
| s_reg-complete | -0.036 | -0.039 | -0.032 | 0 |
| mi15-mean | 0.012 | 0.009 | 0.016 | 0 |
| mi5-mean | 0.013 | 0.009 | 0.017 | 0 |
| reg-mean | 0.022 | 0.018 | 0.026 | 0 |
| s_reg-mean | 0.012 | 0.009 | 0.016 | 0 |
| mi5-mi15 | 0 | -0.004 | 0.004 | 1 |
| reg-mi15 | 0.009 | 0.005 | 0.013 | 0 |
| s_reg-mi15 | 0 | -0.004 | 0.004 | 1 |
| reg-mi5 | 0.009 | 0.005 | 0.013 | 0 |
| s_reg-mi5 | 0 | -0.004 | 0.004 | 1 |
| s_reg-reg | -0.009 | -0.013 | -0.005 | 0 |

TABLE 10. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MNAR and missing proportion is equal to 0.2. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.078 | -0.083 | -0.074 | 0 |
| mi15-complete | -0.082 | -0.086 | -0.077 | 0 |
| mi5-complete | -0.081 | -0.085 | -0.077 | 0 |
| reg-complete | -0.058 | -0.063 | -0.054 | 0 |
| s_reg-complete | -0.079 | -0.084 | -0.075 | 0 |
| mi15-mean | -0.004 | -0.008 | 0.001 | 0.215 |
| mi5-mean | -0.003 | -0.007 | 0.002 | 0.464 |
| reg-mean | 0.02 | 0.015 | 0.024 | 0 |
| s_reg-mean | -0.001 | -0.005 | 0.004 | 0.991 |
| mi5-mi15 | 0.001 | -0.004 | 0.005 | 0.998 |
| reg-mi15 | 0.023 | 0.019 | 0.028 | 0 |
| s_reg-mi15 | 0.003 | -0.002 | 0.007 | 0.569 |
| reg-mi5 | 0.023 | 0.018 | 0.027 | 0 |
| s_reg-mi5 | 0.002 | -0.003 | 0.006 | 0.839 |
| s_reg-reg | -0.021 | -0.025 | -0.016 | 0 |

TABLE 11. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MNAR and missing proportion is equal to 0.3. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

| | diff | lwr | upr | p adj |
|---|---|---|---|---|
| mean-complete | -0.115 | -0.12 | -0.11 | 0 |
| mi15-complete | -0.143 | -0.148 | -0.138 | 0 |
| mi5-complete | -0.142 | -0.147 | -0.137 | 0 |
| reg-complete | -0.106 | -0.111 | -0.101 | 0 |
| s_reg-complete | -0.138 | -0.143 | -0.133 | 0 |
| mi15-mean | -0.027 | -0.032 | -0.022 | 0 |
| mi5-mean | -0.026 | -0.031 | -0.021 | 0 |
| reg-mean | 0.009 | 0.004 | 0.014 | 0 |
| s_reg-mean | -0.023 | -0.028 | -0.018 | 0 |
| mi5-mi15 | 0.001 | -0.004 | 0.006 | 0.988 |
| reg-mi15 | 0.036 | 0.031 | 0.042 | 0 |
| s_reg-mi15 | 0.004 | -0.001 | 0.009 | 0.122 |
| reg-mi5 | 0.035 | 0.03 | 0.04 | 0 |
| s_reg-mi5 | 0.003 | -0.002 | 0.008 | 0.42 |
| s_reg-reg | -0.032 | -0.037 | -0.027 | 0 |

TABLE 12. The table shows the results of the Tukey test to compare mean performance of different imputation methods. Missing data mechanism is MNAR and missing proportion is equal to 0.4. "complete" represents the results for the complete dataset. "mean" represents the results for mean imputation. "reg" for regression imputation. "s_reg" for stochastic regression imputation. Finally, "mi5" and "mi15" represents multiple imputation with 5 and 15 imputed values. "lwr" and "upr" are respectively the lower and upper bounds of the 95% confidence interval for the mean difference "diff". "p adj" is the adjusted p-value.

**UNIVERSITÉ CATHOLIQUE DE LOUVAIN**
**Faculté des sciences**
Place des Sciences, 2 bte L6.06.01, 1348 Louvain-la-Neuve, Belgique | www.uclouvain.be/sc