Amputation Method to Include Categorical Data

Varun Kamat
Eindhoven University of Technology
v.kamat@student.tue.nl

Niels Schelleman
Eindhoven University of Technology
n.v.schelleman@student.tue.nl

Tim van Zoggel
Eindhoven University of Technology
t.a.w.v.zoggel@student.tue.nl

Wouter van der Wee
Eindhoven University of Technology
w.j.v.d.wee@student.tue.nl

Yaron Heerkens
Eindhoven University of Technology
y.heerkens@student.tue.nl

ABSTRACT

Data missingness is a commonly occurring problem in data science and handling this appropriately is essential for reliable data analysis. To deal with such missingness amputation functions were created which remove observations to create a more realistic dataset for us to further process. This paper "describes an extension of the data amputation method developed by R. Schouten et al. [1], elaborating a multivariate amputation technique which allows for better amputation methods than any of its contemporaries, however, the method does not support the amputation of categorical variables. For this reason we sought to create a robust function which extends the use of the pymice method such that it can ampute categorical, more specifically ordinal variables. To test this method we put it through realistic simulations. From these simulations can be concluded that the new ordinal ampute function does better reflect expected characteristics than the old method.

1 INTRODUCTION

Missing data takes place when no data value is stored for the variable in an observation. It occurs in practically all studies, even in those that are well-designed and controlled. In his paper [8] Yiran Dong talks about how missing data might lower a study's statistical power and lead to inaccurate results due to skewed estimates. As a result, various studies have focused on how to deal with missing data, the problems that missing data might bring, and how to avoid or reduce missing data in all fields of science and engineering. Missing data is a typical issue in real-world data analysis.

Schafer J.L, [7] found that 36 percent of studies had no missing data, 48 percent had missing data, and about 16 percent cannot be determined. Among studies that showed evidence of missing data, 97 percent used the listwise deletion (LD) or the pairwise deletion (PD) method to deal with missing data. These two methods are ad hoc and notorious for biased and/or inefficient estimates in most situations.

Missing data can occur for a variety of reasons. In survey data, a respondent may choose not to answer a question or leave the survey before answering all of the questions. Some respondents in a panel poll may skip the second or subsequent waves. In the case of administrative data, records may be lost during the collection or recording process. We can better grasp the significance of missing data for our analysis and inferences if we understand the process by which data goes missing.

In recent years, an increase in the importance of solving missing data problems has led to a surge in new imputations methods being proposed. Most of these methods rely on the amputation of simulated, complete, datasets to generate a fair baseline over which different methods can be compared. The main focus of this paper is therefore to expand the generalizability of one of these amputation methods to include amputation options for ordinal data such that these kinds of data can also be considered for future research.

2 RELATED WORK

Python's scikit-learn package [9] has two major functionalities to deal with missing data, Univariate and multivariate imputations. Univariate imputation is a sort of imputation technique that imputes values in the i^{th} feature dimension using just non-missing data in that dimension (e.g. impute.SimpleImputer). Multivariate imputation techniques, on the other hand, estimate missing values using the whole set of accessible feature dimensions (e.g. impute.IterativeImputer).

Basic mechanisms for imputing missing values are provided by the SimpleImputer class. Missing values can be filled in using a constant value or the statistics (mean, median, or most common) of the columns in which the missing values are found. Different missing value encodings are also supported by this class.

Using the IterativeImputer class, which models each feature with missing values as a function of other characteristics and uses that estimate for imputation, is a more advanced technique. It does so in an iterated round-robin method, with one feature column designated as output y and the others handled as inputs X at each stage. For knowing y, a regressor is fitted on (X, y). The regressor is then used to forecast the missing y values. This is done iteratively for each feature, and then repeated for maxIteration imputation rounds. The last imputation round results are returned.

As important as Imputation is, amputation functions are equally important as they help us introduce missingness in complete datasets for us to to further test and process. Various studies around creation of amputation functions have been carried in the past years.

R. Schouten et al. [1] introduced us to an implementation of a multivariate amputation procedure for generating missing values. The approach was a a significant improvement compared to the contemporary methods at the time. Seeing as the method developed was both efficient and able to produce more reliable missing data as it allowed for proper regulation of missing data problems.

The paper also covers how missing values were previously amputed one variable at a time in older amputation experiments. It was difficult to manage the parameters of a missing data situation using this univariate amputation strategy. Another issue was that the missing data caused structural problems which were not a reflection of the researcher's original intent and hence could often be

assessed in the wrong way which lead to erroneous judgments regarding their inferential power. Hence it was vital to have a reliable amputation procedure which led to the creation of the multivariate amputation method.

One major drawback of the multivariate method is that it only works for numerical data which does not represent realistic datasets as a significant portion of them also contain categorical data. There are two types of categorical data. They are ordinal; the type of data with a scale or order to it(E.g. the education level of a person being either low, medium, high, etc.) and nominal; which is the type of data used to name variable(E.g. Name of a person, a color, etc.). Our paper hopes to expand upon the current field of amputation methods by creating a way to ampute using ordinal data.

A paper written Brand, J.P.L. [10] discusses a strategy to validate multiple imputation methods (MI). For this they propose a method which can ampute numerical data using odds-ratios. However, this method is quite obtuse and inflexible to use. For this reason we decided to implement the extension for ordinal data on the paper by Schouten R, et al. [1] as this provided more freedoms to create a better product.

3 THEORETICAL BACKGROUND

Within the literature three types of missing data mechanisms are defined; MCAR, MAR and MNAR as Missing Completely at Random, Missing At Random and Missing Not At Random respectively [4]. These three mechanisms together form the missing data model where each concept represents a situation where the probability for a datapoint to be missing differs for each mechanism.

In mathematical terms the missing data model looks like the following. Assume that D is a dataset of size $N \times P$, where N represents the number of independent rows and P the number of variables. R is defined as a missing data indicator, represented as a matrix of the same size ($N \times P$) where R equals 1 if a certain element D_{ij} is missing and 0 if the variable is observed. All the missing elements are combined as D_{mis} and together with the set D_{obs} it forms the dataset D as a whole. The missing data model, which denotes the probability that a certain value is missing, can be observed in Equation (1). Where α contains the parameters of the missing data. The model is based on the original findings by Rubin [4].

$$Pr(R|D_{mis}, D_{obs}, \alpha) \tag{1}$$

3.1 MCAR

MCAR assumes that the probability to be missing for a certain datapoint is in any way unrelated to either observed or unobserved data. The probability is thus (as the name says) completely at random. This makes the probability to be missing the same for all cases. The missing data mechanism of a dataset, containing individuals with missing data drawn from a random subset of a population, is denoted as MCAR [2]. This implies that the missing data is the same as the observed data. In Equation (2) it can be observed that the probability that R equals 1 (a specific element is missing) for MCAR indeed does not depend on D_{mis} or D_{obs}, but only on α.

$$Pr(R = 1|D_{mis}, D_{obs}, \alpha) = Pr(R = 1|\alpha) \tag{2}$$

3.2 MAR

MAR consists of the situation where the information about missingness in a specific feature depends on the observed data in the dataset. The assumption of MCAR does not hold anymore; the probability to be missing is not equal among all cases within a dataset. MAR indicates that there are significant differences in missing and observed features which are related to other observed features. An epidemiological article uses a clear example within their field [3]. In a dataset which consist of blood pressure datapoints missing at random conditional on age and sex, the probability to be missing is not the same among all cases. However, these probabilities will be similar among individuals of the same age and sex.

In MAR the observed data is different from the unobserved data. However, the observed data can be used to infer about the missing values, which allows for valid statistical hypothesis testing if the missingness is modeled correctly. In Equation (3) it can be observed that the probability to be missing depends again on α, but also on D_{obs} in the MAR case.

$$Pr(R = 1|D_{mis}, D_{obs}, \alpha) = Pr(R = 1|D_{obs}, \alpha) \tag{3}$$

3.3 MNAR

In short, the third mechanism (MNAR) occurs when the first two do not. The probability of missing data is related to the unobserved data. The reason for the missing value to be missing is because of the missing value itself. Examples of missing data missing not at random within the existing literature are often related to public opinion surveys. Individuals with weaker opinions will respond less often [5]. And individuals currently dealing with a depression can be more/less likely to answer questions about depression [2]. From the mathematical point of view, the probability to be missing does depend on all three variables: D_{mis},D_{obs}, and α (as can be observed in Equation (4).

$$Pr(R = 1|D_{mis}, D_{obs}, \alpha) = Pr(R = 1|D_{mis}, D_{obs}, \alpha) \tag{4}$$

4 CATEGORICAL MISSINGNESS SCENARIOS

For the implementation of categorical variables in data amputation, we can distinguish between 6 main scenarios, based on the feature types: nominal, ordinal, numerical and binary.

Nominal features are considered to be categorical features (i.e. no numerical value) in which there is no natural order (e.g. color of a car).

Ordinal features are also categorical, but there is an order in the values (e.g. lower, medium and higher education). This order does not necessarily have to be linear or easily converted into numerical (e.g. higher education is not 3 times higher than lower education).

Numerical features are measured on a numerical scale in which there is a clear natural order (e.g. height of a person). Within numerical features we can distinguish between continuous numerical features, for which the values are uncountable (e.g. height in centimeters can be expanded with infinitely many decimals), and discrete numerical features, which have countable values (e.g. number of children a person has).

Binary features also take on numerical values, but can only take on a value of 0 or 1 (e.g. to encode gender of a person), and can therefore be processed in the same way as numerical features.

In addition to those main feature types, we often speak of dummy variables to encode nominal features: these are obtained when we split a nominal feature into binary features for each unique value indicating whether that value is present (1) or not (0) for an observation (e.g. color of a car is encoded as 3 binary features 'red', 'blue', 'green').

These feature types result in the following scenarios in terms of categorical features in data amputation:

(1) MNAR: nominal
(2) MNAR: ordinal
(3) MAR: nominal → numerical
(4) MAR: ordinal → numerical
(5) MAR: numerical → nominal
(6) MAR: numerical → ordinal

For MAR missing data the missingness may be affected by multiple observed variables at once, but for clearness, in the last 4 cases we focus on the implementation of one particular type of data at a time. The step of generalizing this into multiple observed variables is trivial, as we can process each feature type separately and then compute the weighted sum scores on the combination of them.

4.0.1 MNAR: nominal. In the MNAR case for nominal data, a nominal variable has missing data for which the likelihood of being missing depends on the value itself (e.g. in a car dataset, the color variable of red cars is more often missing than that of differently colored cars).

As there is generally no natural order to be derived in nominal features (besides manually specifying their effect on data missingness), we propose to stay with the current approach of the ampute function to encode nominal features using dummy variables, and thus provide no alternative method for this case.

4.0.2 MNAR: ordinal. The MNAR case for ordinal data is similar to the nominal one: the likelihood of being missing depends on the variable itself, but in this case for ordinal data (e.g. the energy label variable is more often missing in cars with inefficient labels than in those with more efficient labels).

In Section 5.2 we present a method to derive numerical values for ordinal features that can be used to determine the probability of going missing.

4.0.3 MAR: nominal → numerical. In this case, it is of interest how an ordinal variable can affect the missingness of a numerical variable (e.g. the mileage of red cars is more often missing than that of differently colored cars).

Similar to the first case, we provide no alternative method of dealing with the effect of nominal features on data missingness.

4.0.4 MAR: ordinal → numerical. The MAR missingness of numerical variables can also be the result of ordinal variables (e.g. the mileage of cars with more efficient energy labels is more often missing than that of cars with inefficient labels).

As with case 2, our alternative method of dealing with ordinal features is described in Section 5.2.

4.0.5 MAR: numerical → nominal. For this case, we study how the likelihood of a nominal variable going missing can be affected by numerical variables (e.g. the car brand is more often missing for cars with a high mileage).

4.0.6 MAR: numerical → ordinal. This case is similar to the above, but focuses on missingness in ordinal variables determined by numerical variables (e.g. the energy label of cars with a high mileage is more likely to go missing).

For the last two cases we provide no alternative method as the current approach to numerical features remains applicable in determining which cases should have missing values. The focus of this paper will thus be on cases 2 and 4.

5 METHODS

In this section we will discuss both the original amputation method provided by the pymice package which can handle both numerical and nominal data as well as the extension we propose which would allow for the amputation of ordinal data using the pymice amputation method.

5.1 The original Ampute

The ordinal amputation function is an extension of the multivariate amputation procedure proposed in a paper by Schouten, Lugtig & Vink [1]. The function adds a way for ordinal data to be properly processed by the multivariate amputation procedure. The original amputation method takes a fully numerical dataset, D, of size $n * p$ along with a set of patterns, frequencies and weights. The patterns indicate all possible combinations of missing values that could be present in one of the n rows (e.g. a pattern 0,1,1 would mean the value in the first column will be missing whilst the other two are not when this pattern is applied);

The frequencies dictate in what proportion these patterns should be applied. For a target missingness of, for instance, 40% of rows containing missing data, the frequencies [0.2;0.4;0.4] would indicate that out of the 40% missing rows, 20% of them would be missing according to the first pattern, with the remaining 80% of missing rows being divided over the last two patterns;

The weights indicate, for each pattern, how important each variable is for calculating a weighted sum score. This weighted sum score, which is calculated for each row, can be defined as:

$$wss = \sum_{i=1}^{m} w_i x_i$$

Where w_i is the weight for the i^{th} column given some missingness pattern p and x_i is the numerical data value in that row at the i^{th} column, with x being a vector containing all the values in the x^{th} row of the dataset.

A logistic distribution is subsequently fit on the set of these weighted sum scores in order to calculate whether a row will contain a missing value or not. The shape of this logistic distribution depends on the missingness proportion, with higher proportions transforming the logistic distribution such that a larger portion of the wss will result in having missing data. along with the kind of missingness. This can be either: left; where low values for the wss are more likely to be missing, right; where high values for the wss

are more likely to be missing, tail; where extreme values for the wss are more likely to be missing or mid; where values for the wss close to the average are more likely to be missing [1].

The pymice implementation for the multivariate amputation method also proposes a solution for nominal data which involves creating dummy variables for each of the nominal variables and subsequently keeping the patterns and weights for the dummy components of the variable consistent. This makes it possible to ampute nominal data using the multivariate amputation method. However, for ordinal data creating dummy variables may not be a good idea as doing this results in a loss of the ordering information present in ordinal data. Therefore, we propose a function which can process the ordinal variables without loss of the ordering information.

5.2 The Ordinal Extension

The ordinal extension for the multivariate amputation works by first generating a numerical "dummy" interval: an interval where the true numerical values (unknown) for the data are within the interval of the level they were assigned to, for some constant transformation on the data $D * c$: where the entire dataset is multiplied by some, non-zero constant. This means that the relations between the levels should be the same as in the true population. Unfortunately, the relation between the levels cannot simply be obtained from the data itself and must therefore be provided (or approximated) by the user.

The user provides this information using two inputs. The first input is a binary value u which dictates whether the relation between the levels is related to the amount of observations for each level or not. If it is not, the function assumes that all levels have the same amount of observations in the true population. Otherwise it assumes that the amount of observations in the true population for each level are proportional to those for each level within the dataset. The second input is a function $f(x)$ which dictates how the levels are related to each other. Setting $f(x)$ to be equal to $2x$ would, for instance, imply that when the values in the true population are ordered, each value would be twice the size of the previous one in the data, and therefore, each subset of the true population would, on average be twice as big as the previous, same size subset, in terms of the sum of their values. If nothing is known about the relation between the levels, such that an appropriate $f(x)$ cannot be chosen it may be best to either choose the default $f(x) = x$, when the user at least knows that one level contains higher values than the previous or, when this is not the case, to interpret the ordinal variable as some nominal variable and use the already present methods for nominal variables.

If both the u and $f(x)$ are specified, the intervals are calculated as follows:

$$interval_i = (low; high)$$
$$low = f(\sum_{j=1}^{x_{i-1}} |x_j|^u)$$
$$high = low + f(|x_i|^u)$$

Where x_i denotes a vector, representing some level within the ordinal variable x, with $|x_j|$ being the size of the vector x at level j; $f(x)$ denotes the input function that defines the relation between the variables; u denotes whether or not the size of each of the levels needs to be taken into account. As an example, to calculate the lower bounds with $f(x) = 2x$ and $u = 0$ and an input with four levels (their size does not matter as $u = 0$), the results will be [0,2,4,6]. If $u = 1$ and the sizes were [3,15,6,7], Then the lower bounds would become: [0,6,36,48].

These intervals can then be subsequently used to calculate a wss for the rows within the dataset. This is done by calculating two different scores, one for the ordinal part of the dataset (wss_o), consisting of o variables and one for the numerical part of the dataset (wss_n), which is the same procedure as the one in the original amputation method. Adding these two would then give the complete wss.

$$wss = wss_o + wss_n$$
$$wss_n = \sum_{i=1}^{p-o} w_i x_i$$

The wss_o is calculated by first calculating a partial wss for every possible combination of the high and low bounds of the interval. Since there are o ordinal variables and two bounds for each of the variables, this leads to calculating 2^o scores for each row. These scores are subsequently averaged by calculating some L^α norm over a vector of these scores, with the norm value denoted by L. This norm is subsequently multiplied by $2^{-\frac{o}{L}}$ to normalize the results to the domain of the smallest partial wss and largest partial wss. For $L = 1$ this just gives the average over all these combinations but for higher values of L this tends towards the maximum whilst for lower values it tends towards the minimum. This can be used to create biases towards either low or high partial wss values. This may be useful to make the entire set of ordinal variables more or less important towards the entire wss or when the user perceives either the low or high bound to be more important than the other.

An example of this can be seen in Figure 1 where several models with two ordinal variables (thus $2^2 = 4$ combinations or 4 partial wss) are shown for a range of values for L, note that $L = 0$ does not exist due to division by 0.

Mathematically the wss_o is calculated as follows ($wss_p(i)$ indicates one of the partial wss):

$$wss_o = (2^{-\frac{o}{L}} \left(\sum_{i=1}^{2^o} [wss_p(i)^L] \right)^{1/L}$$

$$wss_p(i) = \sum_{j=1}^{o} w_j * low_j^{\lceil \frac{i}{2^{(j-1)}} \rceil \mod 2} * high_j^{1-(\lceil \frac{i}{2^{(j-1)}} \rceil \mod 2)}$$

The first function computes the wss_o by applying the transformation related to L to all the wss_p giving some wss score for the ordinal variables. The second function calculates a wss for one specific combination of high and low bounds for the all the ordinal variables. Where i denotes the combination number and j denotes a specific variable in the ordinal columns. Note that when the sum of the partial wss is negative, instead of taking a fractional power over a negative number, the fractional power is computed on the absolute value of the sum and then multiplied by -1.

As an example, to recreate the first line in Figure 1 can be found in the appendix.

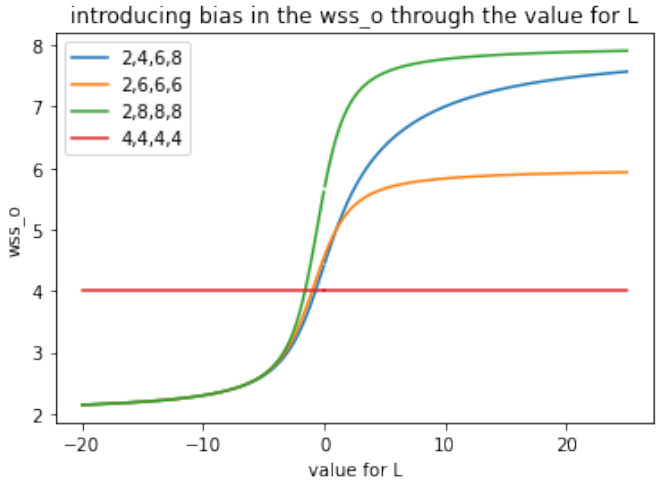

Figure 1: This shows four distinct sets of partial wss for a range of values for the variable L. The sets show: a balanced set of wss; an unbalanced set with the same mean as the balanced set; an unbalanced set with a different mean and a set containing only one value for all wss.

This method can be used to ampute both numerical and ordinal variables (nominal variables are converted to numerical dummies in the original method) using any combination of numerical and ordinal variables, both for MAR and MNAR cases. For the MNAR case however, the size of o will always be one. This gives comparatively little advantage to using this method as the bias for all the ordinal variables L can be replaced by just tweaking the weight for this ordinal variable. For this reason our extension also has a purpose built function specifically for the MNAR on an ordinal variable. Here, instead of taking both the bounds, the mean of the bounds is calculated and used as a numerical substitute for the ordinal data, which, is then converted back into the ordinal data. This, in addition to tweaking the weights (with weights being w_o) can give the same or similar results to the previous method but is a lot less computationally expensive so may be preferred over the previous method when data samples are very large.

$$mean = f(\frac{low + high}{2})$$
$$wss_o = w_o * mean$$

5.3 The Ordinal Extension In Practice

The function which handles the ordinal data for the multivariate amputation, takes several parameters:

- data: The entire dataset on which the amputation needs to be performed.
- ord-cols: A list of the names of the ordinal columns to identify which of the columns are ordinal and which are not.
- orders: A list containing ordered lists where each list maps to one of the ordinal variables and the items within the list denote the order of the levels from low to high.
- proportion-based: A list, where for each ordinal column a statement of either True or False is given denoting whether the proportions need to be taken into account (written as u in Section 5.2).
- relations: A list, where for each ordinal column a lambda function is given which transforms the values for x in accordance with the true relations of the ordinal variable (written as $f(x)$ in Section 5.2)
- norm: A list, for each pattern of nonzero numerical values which can include both infinity and negative infinity. This can give some bias or weight for the entire ordinal part of the wss (written as L in Section 5.2)

On top of these parameters, since this function is part of the larger multivariate amputation method [1], the parameters belonging to that also have to be taken into account:

- patterns: A list of patterns which denote which column will be missing.
- weights: The weights belonging to each pattern.
- proportion: The proportion of the rows that will contain amputed data when the procedure is finished.
- mechanisms: A list, for each pattern which missing data mechanism (MNAR, MAR or MCAR) will be used.
- freqs: A list, for each pattern how frequently it occurs compared to the other patterns.
- types: A list of the type of amputation that will be used for each pattern (right, left, mid or tail).

An example for what the function might look like for a dataset (D) with three columns: a numerical column (n) and two ordinal columns ($o1$ and $o2$) each with four levels ($r1, r2, r3, r4$).

```
from ampute import MultivariateAmputation

MultivariateAmputation(
        patterns=[[0,1,1],[1,0,1]],
        prop=0.4,
        weights=[[0,4,2],[0,1,0]],
        mechanisms=['MAR','MNAR'],
        freqs=[0.3,0.7],
        types=['LEFT','MID'])
.do_ordinal_ampute(ord_cols=['o1', 'o2'],
        data=D,
        orders=[['r1', 'r2', 'r3', 'r4'],
        ['r1', 'r2', 'r3', 'r4']],
        proportion_based=[True, False],
        relations=[lambda x: x^2, lambda x: x],
        norm = [-4,1])
```

6 EVALUATION

In this section we discuss the setup for the simulations that are performed in section 7 as well as the hypotheses associated with them and how to quantify their correctness.

6.1 Simulation setup

Evaluation of the ordinal ampute function is done using two different methods. The first evaluation method consists of a Monte

Carlo simulation study where missing data is generated based on either the MAR or MNAR mechanism by using both the old ampute function and the new ordinal ampute function for MAR. In the case of MNAR only the ordinal ampute function using L1 and L2 norms are evaluated, due to the fact that MNAR amputation for categorical variables was not yet implemented in a straightforward manner in python. The goal of this first simulation is to compare the performance of both ampute functions. This is done by comparing amputed datasets to the characteristics belonging to the missingness mechanism used. These characteristics have been analysed in a paper by Schouten, R and Vink, Gerko(2021)[6] and will be further explained in subsection 6.2. The Monte Carlo consists of 1000 simulations where in each simulation data is generated from a multivariate normal distribution, where x_2 or y_1 corresponds to the ordinal variable in respectively the MAR and MNAR simulations, with:

$$\mu = \begin{pmatrix} 5 \\ 5 \\ 10 \end{pmatrix} \tag{5}$$

and

$$\Sigma = \begin{pmatrix} 1 & \rho & \rho \\ \rho & 1 & \rho \\ \rho & \rho & 1 \end{pmatrix} \tag{6}$$

Where ρ is the correlation coefficient which varies between $\rho \in \{0.1, 0.2, ..., 0.9\}$, $\mu = (y_1, x_1, x_2)$. The relation between groups for the ordinal data is exponential, this is chosen to better reflect the effect of the new ordinal ampute function. After data generation, the data is amputed using the patterns and weights found in Table 1, all using 20% missingness. Data is amputed using three different methods: the old ampute function, ordinal amputer using L1 norm and ordinal amputer using L2 norm.

	Pattern			Weight		
Mechanism	y_1	x_1	x_2	y_1	x_1	x_2
MAR	0	1	1	0	0	1
MNAR	0	1	1	1	0	0

Table 1: Patterns and weights used for simulation. Where 0 in pattern indicates a value to be amputed and where a 0 in weight indicates the influence of that variable on the amputation

Table 1 shows that MAR missingness is generated on y_1 due to the dependence on ordinal variable x_1, with categories low, med and high. The type of missingness generated is RIGHT missingness, meaning that probability of being missing for y_1 increases with the ordinal value of x_1. Evaluation of the ordinal amputation method is done by comparing the results to the expected characteristics of the missingness mechanism that is applied and by comparing results to the old ampute function. This comparison happens for the MAR on the mean, confidence interval width and confidence interval coverage of the mean of y_1 by both complete case analysis(CCA) and MI, using bayesian regression as impute function. For the MNAR mechanism the proportion of classes is used to compare the expected characteristics, due to y_1 being ordinal data. This implies that investigating the mean of y_1 is not feasible. Therefore, comparison is subject to the histograms for the original data, CCA data and MI data. Furthermore, since using bayesian regression as an imputer function is not viable for imputing data, it is decided to use a k nearest neighbour impute function with k=1.

The second evaluation studies whether the modified ampute function makes a noticeable difference for the evaluation of imputation techniques in practice. The ampute function is used primarily as a tool here (and less as the end goal), and we study the effect of the amputation method for the evaluation of 3 imputation techniques. Here, the MSE's achieved by the imputation techniques on data amputed by the modified ampute function is compared to the original ampute function for a quantitative comparison. The main goal for this comparison is to illustrate whether or not our alternative approach to ordinal features makes a fundamental difference to the evaluation of imputation techniques in practice, rather than to directly demonstrate the added value of our approach (since there is no ground truth in the performance of imputation techniques that we aim to reproduce).

The experiment is performed on the Contraceptive Method Choice (CMC) dataset from the UCI Machine Learning Repository [11]. This dataset consists of 10 features, of which 3 are ordinal, 2 are numerical, 2 are nominal, and 3 are binary. In line with the approach to nominal features in this paper, the nominal features are encoded as dummy variables. The numerical and binary variables are not modified, and the ordinal variables are processed as specified in the ordinal ampute function. The setup is as follows: the CMC dataset is split into a train (70%) and test set (30%). On the train set, we train 2 imputers: simple imputer and KNN-imputer. Both imputers are implemented as in scikit-learn, but adapted to handle categorical variables as implemented in the experiments of [12]. The test set is then amputed, either according to the original ampute function or the modified ordinal ampute, with a proportion of 0.5, following the MAR mechanism with RIGHT missingness. Ordinal variables are assumed to follow the default relation $f(x) = x$ using the L1 norm. On the amputed test set, we impute with the 2 imputers trained before, as well as using MI as implemented in the R package mice. For the evaluation, we compare the imputed values to the original values by means of mean squared errors. This procedure is repeated 100 times to demonstrate the certainty of our findings.

6.2 Hypotheses

The first simulation is used to test the effects of the new ordinal amputer on exponential related groups within an ordinal variable compared to the old ampute function on this same data. Therefore, hypotheses are subject to the characteristics belonging to the missingness pattern[6]. Hence we expect that in case of RIGHT MAR the estimates of the mean of y_1 are biased when using CCA and estimates are unbiased when using MI. Consequently, the coverage rate for CCA is expected to be below 95%, especially when data correlation increases, in contrast the coverage rate for MI is expected to stay around 0.95, since information about missingness is in the observed data. The confidence interval width is expected to not change too much, due to the low percentage of missingness, however, for CCA the ciw should stay around the same value and for MI the ciw should see a small decrease when correlation grows. In all comparisons the expectation is that the old ampute and ordinal

amputer using L1 norm behave approximately the same. For the ordinal amputer using L2 norm, however, the results are expected to behave more like the theoretical expected behaviour, because this ampute functions takes into account the exponential relation in the ordinal variable. This exponential relation is expected to increase bias in MAR RIGHT missingness, thus also decreasing coverage rate and confidence interval width, in the case of CCA. In case of MI it is expected that the differences between old and new amputation do not differ as much, since the MI learns from observed data, which is not altered.

In the case of right MNAR the only evaluated metric is the proportion of each class. The expectation is that estimated proportions are biased for all correlation settings, due to the lack of knowledge in the observed data. For CCA it is expected that this bias increases with increasing correlation, due to more and more *high* categories being amputed. Furthermore, it is expected that the generated bias when using ordinal ampute L1 functions is less than when using ordinal amputer L2. This is based on the fact that the weights for the third group in the weighted sum scores is exponential bigger than when using the ordinal amputer L1 function.

For the second simulation, it is not expected that the amputation techniques will have a significant effect on the differences between imputation techniques, since this is strongly influenced by the imputation techniques themselves. However, as the effect of the ordinal feature on the weight sum scores is demonstrated to be different in Section 5.2, it is hypothesized that the overall errors of the imputation techniques differ in the datasets amputed with the modified ampute function, as compared to the original one.

7 RESULTS

In this section the results of the simulations, which experimental setup was explained in the previous section, will be elaborated. First simulation 1 is explained by use of visualizations as well as explanations whereafter the same holds for the second simulation. In the next section the results presented in the current section will be discussed whereafter conclusion could be drawn.

7.1 Simulation 1

This section consists of two visualizations with textual explanations. First the simulation on MAR generated data is elaborated whereafter also the MNAR generated data is discussed.

In Figure 2 the values of bias, coverage rate, and CI width (from top to bottom), and their behavior with respect to different correlations used within the dataset can be observed. The MI biases do not differentiate significantly using higher correlations. The CCA biases on average do differentiate; the bias increases together with the correlation in the dataset. The old ampute function has a smaller bias for both values. Within the MI this difference is not significant, while with a CCA the difference in bias grows as the old amputation line has a smaller slope compared with the L1 and L2 methods.

The coverage rate shows the same results. L1 and L2 CCA show the same pattern where L2 drops a bit faster. The old ampute CCA coverage rate has a smaller slope and thus has a smaller effect of an increasing correlation in the begin. With MI this time all the three

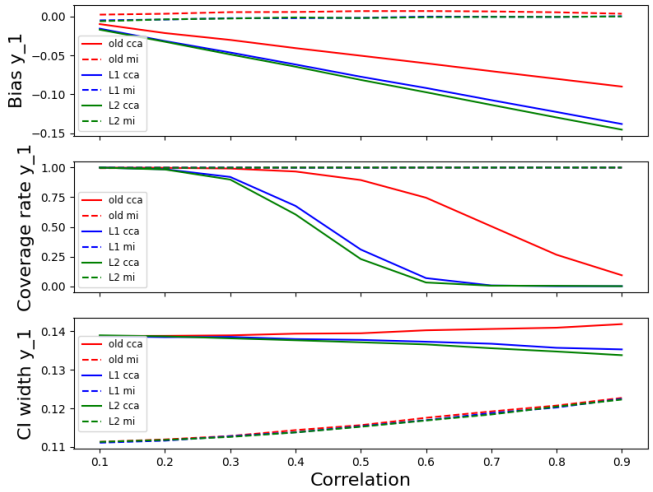

Figure 2: results simulation 1 RIGHT MAR generated data

methods tests are identical in behavior, where again the coverage rate has no effect on increasing the correlation.

The CI width differentiates for both the MI and CCA when the correlation is increased. For the L1 and L2 CCA the lines are close to each other having both a small decrease, while the old ampute CCA has a small increase in CI width instead. The MI CI width has the same behavior for all three methods tested: the CI widths increase together with an increase in the correlation.

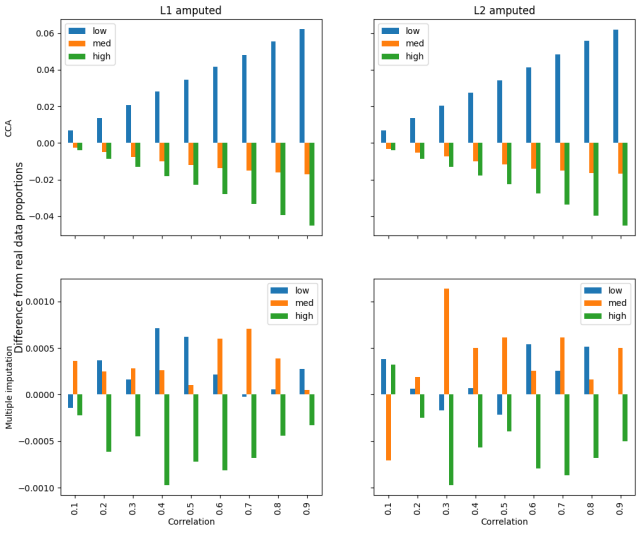

Figure 3: Results simulation 1 MNAR missingness

In Figure 3 the difference between estimated and real proportions of classes is visualized as barplots of the CCA data and the MI data, for both the L1 and L2 amputed methods. For the CCA data there is a pattern observable as the difference in proportions increase together with the increase in correlation within the data. For the L1 amputed data the category *high* ends up underestimated in proportion quite a bit while for the L2 amputed data the *low* category proportion is highly overestimated. In regards to the *med* category can be seen that for L1 amputed data the proportion is estimated correctly for lower correlations , but start to be overestimated a tiny bit when correlation increase. When using L2, the *med* proportions are underestimated for every correlation.

The MI data shows different patterns, there is no average increasing pattern observable. Within the L1 amputed the category *high* has the biggest underestimated proportion for each correlation. Within the L2 amputed data for all, except 0.1, correlations the *high* category proportion is below the real values. Most of this bias is introduced by classifying more cases as *med*.

7.2 Simulation 2

In this section the results of the second simulation are presented where experiments cover the different f(x)'s, comparisons between the old and new amputation method, and the use of different norms.

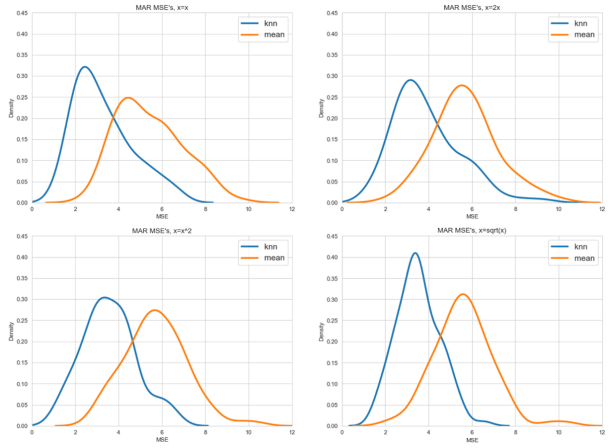

Figure 4: Results simulation 2 MAR on new ampute, using different f(x)'s

In Figure 4 the MSE's of the KNN and mean imputation of the MAR dataset are visualized as density plots. It can be observed that the average MSE as well as the variance (width of the densities) varies across the different f(x)'s tested. The relations tested are $x = x$, $x = 2x$, $x = x^2$, and $x = \sqrt{(x)}$. In general, the MSE of mean imputation is higher than KNN imputation, regardless of the ampute function. For KNN imputation the average MSEs are approximately the same. However, the width of the densities differs. For KNN the average MSEs are also approximately the same, except for the relation $x = x$ which is somewhat lower. Also here the width of the densities differs.

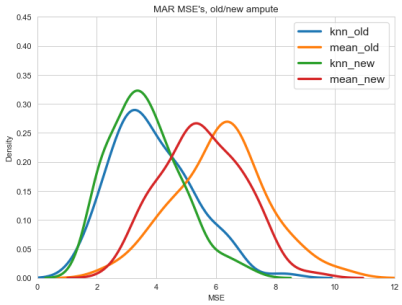

Figure 5: Results simulation 2 MAR on old/new ampute

In Figure 5 the MSE's of the KNN and mean imputation on the MAR dataset for the old and the new amputation (using relation $f(x) = x$) functions can be visualized. In general, the MSE's look approximately the same. Also using the old amputation the MSE of the KNN imputation is lower than the mean imputation. The density of the MSE of the new amputation is somewhat higher compared with the old one. On the other hand, the MSE of the mean imputation using the new amputation method is somewhat lower than using the original amputation method.

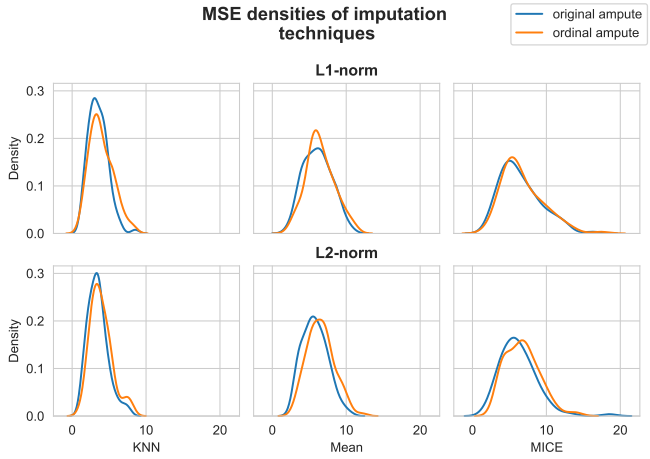

Figure 6: Densities of MSEs of imputation techniques on CMC dataset (50% MAR missingness ($f(x) = x$, for L1 and L2 norm)

Figure 6 shows densities of the MSE's obtained by 3 imputation technique on datasets amputed with both ampute functions, for the L1 and L2 norm. The densities show no significant differences between the two ampute functions for either of the norms, nor does it show differences in the overall errors between the two norms. The evaluation of imputation techniques on this dataset thus does not seem to be affected by the chosen norm.

8 CONCLUSIONS

In this section we will reflect on our hypotheses by comparing them to the found results. First results of simulation 1 are discussed, second the results of simulation 2.

8.1 Simulation 1

The expectations for simulations were divided into 2 parts, a MAR RIGHT amputation part and a MNAR missingness part. The expectations for the MAR RIGHT were that the amputated data followed the theoretical characteristics of MAR RIGHT missing data as explained by Schouten, R and Vink, Gerko(2021)[6]. This entails that, it was expected that: The amputed data in case of CCA showed a negative bias, larger for the ordinal amputer than for the old ampute function; The coverage rate would drop below 95% in case of CCA and, again, using ordinal amputing this drop happens faster and that the CI width would not change to much.

By examining the results for the MAR RIGHT in Figure 2, we saw that indeed when using CCA the bias of the mean of y_1 increased negatively. Especially, it increased more when using the ordinal amputer, with a small difference between L1 and L2 norm. For the coverage rate we also see what was expected. Again, for the new function can be seen that coverage rate drops faster than when using the old method, in the case of CCA. Both of these findings show that using the new function better reflects the exponential relation in the ordinal data, by adding more missingness to the rows in the higher groups. Due to this missingness, rows belonging in a high quantile are missing and thus mean estimates are below the real estimates of the mean. This effect should be bigger when using exponential related ordinal groups, since this also means that being in the highest group should give a exponential change to the wss, differentiating the change to be missing. For the ciw we see that, in CCA, the new function shows a small decrease in width, whereas the old function shows a small increase. The decrease is as expected since higher values have more chance to be missing, thus decreasing the variance. For all MI cases, the differences between old and new ampute functions do not differ much. The imputation model looks at complete rows to fill in missing values, thus whether the underlying relationship is exponential or linear does not matter leading to unbiased estimates either way.

In the MNAR simulation it was expected that bias would increase with the correlation. We see this happening very clear in the CCA model, this happens due to the increase of missing rows within the *high* category. This suggests the function to work as desired. In the MI case can be seen that the imputation technique has trouble estimating the right proportions. *high* categories are underestimated at almost every correlation and most of the times estimated as belonging to *med* category. Meaning that negative bias remains for the MNAR missingness even when using imputation.

8.2 Simulation 2

The hypothesized differences for simulation 2 were not convincingly confirmed by the results. While specifying a different relation between the ordinal values does seem to lead to a slight change in error distributions, the differences are too small to impact the evaluation of the imputation techniques on the CMC dataset. The experiments on the effect of the norm similarly show no change in errors or evaluation of the imputation techniques on the CMC dataset. The added value of ordinal ampute may therefore be limited to datasets in which the relation between ordinal values is estimated or known to be non-linear.

9 DISCUSSION

While the ordinal extension does seem to allow for better amputation of ordinal variables, within the multivariate amputation method, the function still has some limitations associated with it.

First and foremost, in order to properly be able to use the function, the user needs to have some idea about how the underlying distribution of each of the ordinal variables is shaped and how the values are distributed within the individual levels. This may not always be realistic to expect from the user, in which case the user would need to make some assumptions which may or not be correct. Ideally the function would include some proven optimal algorithm for amputing the ordinal variables when the user has only minimal or no information about the underlying true distribution.

Second, Since the ordinal function computed a partial wss for each combination of the high and low bounds for each of the ordinal variables. The running time of this would be non-polynomial with respect to the amount of ordinal variables. This means that for datasets which contain a large amount of ordinal variables the amputation process could become very slow. A possible solution to this might be to include an approximation function which would approximate the values generated for the ordinal wss when there are a large number of ordinal variables in a more computationally efficient way.

Third, the experiment on a practical dataset aimed to demonstrate the added value of the modified amputation function for the evaluation of imputation techniques. While the chosen CMC dataset contains relatively many ordinal features (3 out of 10), there was no known relationship between the ordinal values of any of the features, which is the main contribution of the method discussed in this paper. For a more conclusive result on the value of the modified ampute function, it might have been worthwhile to test it on a dataset for which this relation is known to be non-linear (e.g. age intervals with unequal interval widths and frequencies).

REFERENCES

[1] Schouten, R. and Lugtig, P. and Vink, G. (2018) Generating missing values for simulation purposes: a multivariate amputation procedure, Journal of Statistical Computation and Simulation, 88:15, 2909-2930, DOI: 10.1080/00949655.2018.1491577

[2] Pedersen, A. and Mikkelsen, E. and Cronin-Fenton, D. and Kristensen, N. and Pham, T. and Pedersen, L. and Petersen, I. (2017) Missing data and MI in clinical epidemiological research, Clinical epidemiology 9, p. 157.

[3] Bhaskaran, K. and Smeeth, K. (2014) What is the difference between missing completely at random and missing at random? International journal of epidemiology 43, p.1336-1339

[4] Rubin and Donald, B. (1974) Inference and missing data, Biometrika, v.63, p.581-592, doi:10.1093/biomet/63.3.581

[5] Van Buuren, S. (2018). Flexible imputation of missing data. CRC press.

[6] Schouten, R. and Vink, G. (2021) The Dance of the Mechanisms: How Observed Information Influences the Validity of Missingness Assumptions, Sociological Methods and Research, p. 1243-1258

[7] Schafer J.L. Analysis of incomplete multivariate data. London: Chapman & Hall/CRC; 1997. [Google Scholar]

[8] Principled missing data methods for researchers Dong, Y. and Peng, C.Y. Springerplus. 2013; 2: 222. Published online 2013 May 14. doi: 10.1186/2193-1801-2-222 PMCID: PMC3701793

[9] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V.,Thirion, B., Grisel, O., et al. Duchesnay, E. (2011). Scikit-learn: Machine learning in Python.Journal of MachineLearning Research,12, 2825–2830.

[10] Brand, J.P., van Buuren, S., Groothuis-Oudshoorn, K. and Gelsema, E.S. (2003), A toolkit in SAS for the evaluation of multiple imputation methods. Statistica Neerlandica, 57: 36-45.

[11] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of

Information and Computer Science.
[12] Correia, A. (2021). Generative Forests in Python. GitHub. Retrieved October 27, 2021, from https://github.com/AlCorreia/GeFs.

APPENDIX

Example calculating the wss_o

For this example we have two ordinal variables, which at this specific row have bounds: (1,3) and (1,5). We will also set all the weights to be 1. This would give us four wss_p. For the sake of convenience we will write mod 2 as %2. The calculation of the wss_o will then go as follows:

The wss_o can be calculated with:

$$wss_o = 2^{-\frac{2}{L}}(wss_p(1)^L + wss_p(2)^L + wss_p(3)^L + wss_p(4)^L)^{\frac{1}{L}}$$

We want to calculate the wss_p:

$$wss_p(1) = 1 * 1^{\lceil \frac{1}{1} \rceil \%2} * 3^{1-\lceil \frac{1}{1} \rceil \%2} + 1 * 1^{\lceil \frac{1}{2} \rceil \%2} * 5^{1-\lceil \frac{1}{2} \rceil \%2}$$

$$wss_p(1) = 1^1 * 3^0 + 1^1 * 3^0 = 2$$

$$wss_p(2) = 1 * 1^{\lceil \frac{2}{1} \rceil \%2} * 3^{1-\lceil \frac{2}{1} \rceil \%2} + 1 * 1^{\lceil \frac{2}{2} \rceil \%2} * 5^{1-\lceil \frac{2}{2} \rceil \%2}$$

$$wss_p(2) = 1^0 * 3^1 + 1^1 * 5^0 = 4$$

$$wss_p(3) = 1 * 1^{\lceil \frac{3}{1} \rceil \%2} * 3^{1-\lceil \frac{3}{1} \rceil \%2} + 1 * 1^{\lceil \frac{3}{2} \rceil \%2} * 5^{1-\lceil \frac{3}{2} \rceil \%2}$$

$$wss_p(3) = 1^1 * 3^0 + 1^0 * 5^1 = 6$$

$$wss_p(4) = 1 * 1^{\lceil \frac{4}{1} \rceil \%2} * 3^{1-\lceil \frac{4}{1} \rceil \%2} + 1 * 1^{\lceil \frac{4}{2} \rceil \%2} * 5^{1-\lceil \frac{4}{2} \rceil \%2}$$

$$wss_p(4) = 1^0 * 3^1 + 1^0 * 5^1 = 8$$

Plugging them in gives:

$$wss_o = 2^{-\frac{2}{L}}(2^L + 4^L + 6^L + 8^L)^{\frac{1}{L}}$$

Let L=2

$$wss_o = 2^{-\frac{2}{2}}(2^2 + 4^2 + 6^2 + 8^2)^{\frac{1}{2}}$$

$$wss_o = \frac{1}{2}\sqrt{4 + 16 + 36 + 64}$$

$$wss_o = \frac{1}{2}\sqrt{120} \approx 5.7$$