Survey Methodology

Are deep learning models superior for missing data imputation in surveys? Evidence from an empirical comparison

by Zhenhua Wang, Olanrewaju Akande, Jason Poulos and Fan Li

Statistics Canada   Statistique Canada

Canada

How to obtain more information

For information about this product or the wide range of services and data available from Statistics Canada, visit our website, www.statcan.gc.ca.

You can also contact us by

Email at infostats@statcan.gc.ca

Telephone, from Monday to Friday, 8:30 a.m. to 4:30 p.m., at the following numbers:

- Statistical Information Service   1-800-263-1136
- National telecommunications device for the hearing impaired   1-800-363-7629
- Fax line   1-514-283-9350

Standards of service to the public

Statistics Canada is committed to serving its clients in a prompt, reliable and courteous manner. To this end, Statistics Canada has developed standards of service that its employees observe. To obtain a copy of these service standards, please contact Statistics Canada toll-free at 1-800-263-1136. The service standards are also published on www.statcan.gc.ca under "Contact us" > "Standards of service to the public."

Note of appreciation

Canada owes the success of its statistical system to a long-standing partnership between Statistics Canada, the citizens of Canada, its businesses, governments and other institutions. Accurate and timely statistical information could not be produced without their continued co-operation and goodwill.

Are deep learning models superior for missing data imputation in surveys? Evidence from an empirical comparison

Zhenhua Wang, Olanrewaju Akande, Jason Poulos and Fan Li[1]

Abstract

Multiple imputation (MI) is a popular approach for dealing with missing data arising from non-response in sample surveys. Multiple imputation by chained equations (MICE) is one of the most widely used MI algorithms for multivariate data, but it lacks theoretical foundation and is computationally intensive. Recently, missing data imputation methods based on deep learning models have been developed with encouraging results in small studies. However, there has been limited research on evaluating their performance in realistic settings compared to MICE, particularly in big surveys. We conduct extensive simulation studies based on a subsample of the American Community Survey to compare the repeated sampling properties of four machine learning based MI methods: MICE with classification trees, MICE with random forests, generative adversarial imputation networks, and multiple imputation using denoising autoencoders. We find the deep learning imputation methods are superior to MICE in terms of computational time. However, with the default choice of hyperparameters in the common software packages, MICE with classification trees consistently outperforms, often by a large margin, the deep learning imputation methods in terms of bias, mean squared error, and coverage under a range of realistic settings.

Key Words: Deep learning; Hyperparameter selection; Missing data; Multiple imputation by chained equations; Simulation studies; Survey data.

1. Introduction

Many sample surveys suffer from missing data, arising from unit nonresponse, where a subset of participants do not complete the survey, or item nonresponse, where missing values are concentrated on particular questions. In opinion polls, nonresponse may reflect either refusal to reveal a preference or lack of a preference (De Leeuw, Hox and Huisman, 2003). If not properly handled, missing data patterns can lead to biased statistical analyses, especially when there are systematic differences between the observed data and the missing data (Rubin, 1976; Little and Rubin, 2019). Complete case analysis on units with completely observed data is often infeasible and may lead to large bias in most situations (Little and Rubin, 2019). As a result, many analysts account for the missing data by imputing missing values and then proceeding as if the imputed values are true values.

Multiple imputation (MI) (Rubin, 1987) is a popular approach for handling missing values. In MI, an analyst creates $L > 1$ completed datasets by replacing the missing values in the sample data with plausible draws generated from the predictive distribution of probabilistic models based on the observed data. In each completed dataset, the analyst can then compute sample estimates for population estimands of interest, and combine the sample estimates across all L datasets using MI inference methods developed by Rubin (1987), and more recently, Rubin (1996), Barnard and Meng (1999), and Reiter and

Raghunathan (2007), and Harel and Zhou (2007). In MI, the estimated variance of an estimand consists of both within-imputation and between-imputation variances, and thus takes into account the inherent variability of the imputed values. Note that in survey studies, single imputation, e.g., via matching or regression, remains to be common for dealing with missing data, where the variance is estimated via the delta method or resampling methods (Chen and Haziza, 2019; Haziza and Vallée, 2020).

1.1   Model-based imputation

There are two general modeling strategies for MI. The first strategy, known as *joint modeling* (JM), is to specify a joint distribution for all variables in the data, and then generate imputations from the implied conditional (predictive) distributions of the variables with missing values (Schafer, 1997). The JM strategy aligns with the theoretical foundation of Rubin (1987), but it can be challenging to specify a joint model with high-dimensional variables of different types. Indeed, most popular JM approaches, such as "PROC MI" in SAS (Yuan, 2011), and "AMELIA" (Honaker, King and Blackwell, 2011) and "norm" in R (Schafer, 1997), make a simplifying assumption that the data follow multivariate Gaussian distributions, even for categorical variables, which can lead to bias (Horton, Lipsitz and Parzen, 2003). Recent research developed flexible JM models based on advanced Bayesian nonparametric models such as Dirichlet Process mixtures (Manrique-Vallier and Reiter, 2014; Murray and Reiter, 2016). However, these methods are computationally expensive, and often struggle to scale up to high-dimensional cases.

The second strategy is called *fully conditional specification* (FCS, van Buuren, Brand, Groothuis-Oudshoorn and Rubin (2006)), where one separately specifies a univariate conditional distribution for each variable with missing values given all the other variables and imputes the missing values variable-by-variable iteratively, akin to a Gibbs sampler. The most popular FCS method is multiple imputation by chained equations (MICE) (van Buuren and Groothuis-Oudshoorn, 2011), usually implemented with specifying generalized linear models (GLMs) for the univariate conditional distributions (Raghunathan, Lepkowski, Van Hoewyk and Solenberger, 2001; Royston and White, 2011; Su, Gelman, Hill and Yajima, 2011). Recent research indicates that specifying the conditional models by classification and regression trees (CART, Breiman, Friedman, Olshen and Stone (1984) and Burgette and Reiter (2010)) comprehensively outperforms MICE with GLM (Akande, Li and Reiter, 2017). A natural extension of MICE with CART is to use ensemble tree methods such as random forests, rather than a single tree (Breiman, 2001; Doove, Van Buuren and Dusseldorp, 2014).

MICE is appealing in large-scale survey data because it is simple and flexible in imputing different types of variables. However, MICE has a key theoretical drawback that the specified conditional distributions may be incompatible, that is, they do not correspond to a joint distribution (Arnold and Press, 1989; Gelman and Speed, 1993; Li, Yu and Rubin, 2012). Despite this drawback, MICE works remarkably well in real applications and numerous simulations have demonstrated it outperforms many theoretically sound JM-based methods; see van Buuren (2018) for case studies. However, MICE is also computationally intensive (White, Royston and Wood, 2011) and generally cannot be parallelized. Moreover, popular software packages for implementing MICE with GLMs, e.g., `mice` in R (van Buuren

and Groothuis-Oudshoorn, 2011), often crash in settings with high dimensional non-continuous variables, e.g., categorical variables with many categories (Akande et al., 2017).

1.2   Imputation with deep learning models

Recent advances in deep learning greatly expand the scope of complex models for high-dimensional data. This advancement brings the hope that a new generation of missing data imputation methods based on deep learning models may address the theoretical and computational limitations of existing statistical methods. For example, deep generative models such as generative adversarial networks (GANs, Goodfellow, Pouget-Abadie, Mirza, Xu, Warde-Farley, Ozair, Courville and Bengio (2014)) are naturally suitable for producing multiple imputations because they are designed to generate data that resemble the observed data as much as possible. A method in this stream is the generative adversarial imputation network (GAIN) of Yoon, Jordon and Schaar (2018). Multiple imputation using denoising autoencoders (MIDA, Gondara and Wang (2018) and Lu, Perrone and Unpingco (2020)), is another generative method based on deep neural networks trained on corrupted input data in order to force the networks to learn a useful low-dimensional representation of the input data, rather than its identity function (Vincent, Larochelle, Bengio and Manzagol, 2008; Vincent, Larochelle, Lajoie, Bengio, Manzagol and Bottou, 2010). Several methods have been proposed for missing value imputation in time-series data using variational autoencoders (Fortuin, Baranchuk, Rätsch and Mandt, 2020) or recurrent neural networks (Lipton, Kale and Wetzel, 2016; Monti, Bronstein and Bresson, 2017; Cao, Wang, Li, Zhou, Li and Li, 2018; Che, Purushotham, Cho, Sontag and Liu, 2018; Yoon, Zame and van der Schaar, 2018).

Deep learning based MI methods have several advantages, at least theoretically, over the traditional statistical models, including (i) they avoid making distributional assumptions; (ii) can readily handle mixed data types; (iii) can model nonlinear relationships between variables; (iv) are expected to perform well in high-dimensional settings; and (v) can leverage graphics processing unit (GPU) power for faster computation. Several papers report encouraging performance of deep learning based MI methods compared to MICE (e.g., Yoon, Jordon and Schaar, 2018). However, such conclusions are made based on limited evidence. First, the studies are usually based on small simulations or several well-studied public "benchmark" datasets, such as those described in Section 5, which do not resemble survey data. Second, the evaluations are usually based on a few overall performance metric, e.g., the overall predictive mean squared error or accuracy. Such metrics may not give a full picture of the comparisons and sometimes can be even misleading, as will be illustrated later. Third, given the uncertainty of the missing data process, it is crucial to examine the repeated sampling properties of imputation methods, but these have been rarely evaluated. Finally, hyperparameter tuning is crucial for machine learning models and different tuning can result in dramatically different results, but few details are provided on hyperparameter tuning and its consequences on the performance of imputation methods.

Motivated by these limitations, in this paper we carry out extensive simulations based on real survey data to evaluate MI methods with a range performance metrics. Specifically, we conduct simulations based on a subsample from the American Community Survey to compare repeated sampling properties of

four aforementioned MI methods: MICE with CART (MICE-CART), MICE with random forests (MICE-RF), GAIN, and MIDA. We find that deep learning based MI methods are superior to MICE in terms of computational time. However, MICE-CART consistently outperforms, often by a large margin, the deep learning methods in terms of bias, mean squared error, and coverage, under a range of realistic settings. This contradicts previous findings in the machine learning literature, and raises questions on the appropriate metrics for evaluating imputation methods. It also highlights the importance of assessing repeated-sampling properties of imputation methods. Though we focus on multiple imputation in this paper, we note that the aforementioned MI methods are readily applicable to generate single imputation when L is set to 1. Extensive empirical evidences suggest that the within-imputation variance usually dominates the between-imputation variance in MI. As such, we expect the patterns between different imputation methods observed here also stand if these methods are used for single imputation.

The remainder of this article is organized as follows. In Section 2, we review the four MI methods used in our evaluation. In Section 3, we describe a framework with several metrics for evaluating imputation methods. In Section 4, we describe the simulation design and results with large-scale survey data, and in Section 5 we summarize evaluation results on the benchmark datasets used in machine learning literature. Finally, in Section 6, we conclude with a practical guide for implementation in real applications.

2.   Missing data imputation methods

We first introduce notation. Consider a sample with n units, each of which is associated with p variables. Let Y_{ij} be the value of variable j for individual i, where $j = 1, \ldots, p$ and $i = 1, \ldots, n$. Here, Y can be continuous, binary, categorical or mixed binary-continuous. For each individual i, let $\mathbf{Y}_i = (Y_{i1}, \ldots, Y_{ip})$. For each variable j, let $\mathbf{Y}_j = (Y_{1j}, \ldots, Y_{nj})$. Let $\mathbf{Y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_n)$ be the $n \times p$ matrix comprising the data for all records included in the sample. We write $\mathbf{Y} = (\mathbf{Y}_{\text{obs}}, \mathbf{Y}_{\text{mis}})$, where $\mathbf{Y}_{\text{obs}}$ and $\mathbf{Y}_{\text{mis}}$ are respectively the observed and missing parts of $\mathbf{Y}$. We write $\mathbf{Y}_{\text{mis}} = (\mathbf{Y}_{\text{mis},1}, \ldots, \mathbf{Y}_{\text{mis},p})$, where $\mathbf{Y}_{\text{mis},j}$ represents all missing values for variable j, with $j = 1, \ldots, p$. Similarly, we write $\mathbf{Y}_{\text{obs}} = (\mathbf{Y}_{\text{obs},1}, \ldots, \mathbf{Y}_{\text{obs},p})$ for the corresponding observed data.

In MI, the analyst generates values of the missing data $\mathbf{Y}_{\text{mis}}$ using pre-specified models estimated with $\mathbf{Y}_{\text{obs}}$, resulting in a completed dataset. The analyst then repeats the process to generate L completed datasets, $\{\mathbf{Y}^{(l)}: l = 1, \ldots, L\}$, that are available for inference or dissemination. For inference, the analyst can compute sample estimates for population estimands in each completed dataset $\mathbf{Y}^{(l)}$, and combine them using MI inference rules developed by Rubin (1987), which will be reviewed in Section 3.

2.1   MICE with classification tree models

Under MICE, the analyst begins by specifying a separate univariate conditional model for each variable with missing values. The analyst then specifies an order to iterate through the sequence of the conditional models, when doing imputation. We write the ordered list of the variables as $(\mathbf{Y}_{(1)}, \ldots, \mathbf{Y}_{(p)})$. Next, the analyst initializes each $\mathbf{Y}_{\text{mis},(j)}$. The most popular options are to sample from (i) the marginal

distribution of the corresponding $\mathbf{Y}_{\mathrm{obs},(j)}$, or (ii) the conditional distribution of $\mathbf{Y}_{(j)}$ given all the other variables, constructed using only available cases.

After initialization, the MICE algorithm follows an iterative process that cycles through the sequence of univariate models. For each variable j at each iteration t, one fits the conditional model $(\mathbf{Y}_{(j)} \mid \mathbf{Y}_{\mathrm{obs},(j)}, \{\mathbf{Y}_{(k)}^{(t)}: k < j\}, \{\mathbf{Y}_{(k)}^{(t-1)}: k > j\})$. Next, one replaces $\mathbf{Y}_{\mathrm{mis},(j)}^{(t)}$ with draws from the implied model $(\mathbf{Y}_{\mathrm{mis},(j)}^{(t)} \mid \mathbf{Y}_{\mathrm{obs},(j)}, \{\mathbf{Y}_{(k)}^{(t)}: k < j\}, \{\mathbf{Y}_{(k)}^{(t-1)}: k > j\})$. The iterative process continues for T total iterations until convergence, and the values at the final iteration make up a completed dataset $\mathbf{Y}^{(l)} = (\mathbf{Y}_{\mathrm{obs}}, \mathbf{Y}_{\mathrm{mis}}^{(T)})$. The entire process is then repeated L times to create the L completed datasets. We provide pseudocode detailing each step of the MICE algorithm in the supplementary material.

Under MICE-CART, the analyst uses CART (Breiman et al., 1984) for the univariate conditional models in the MICE algorithm. CART follows a decision tree structure that uses recursive binary splits to partition the predictor space into distinct non-overlapping regions. The top of the tree often represents its root and each successive binary split divides the predictor space into two new branches as one moves down the tree. The splitting criterion at each leaf is usually chosen to minimize an information theoretic entropy measure. Splits that do not decrease the lack of fit by an reasonable amount based on a set threshold are pruned off. The tree is then built until a stopping criterion is met; e.g., minimum number of observations in each leaf.

Once the tree has been fully constructed, one generates $\mathbf{Y}_{\mathrm{mis},(j)}^{(t)}$ by traversing down the tree to the appropriate leaf using the combinations in $(\{\mathbf{Y}_{k}^{(t)}: k < j\}, \{\mathbf{Y}_{k}^{(t-1)}: k > j\})$, and then sampling from the $Y_{(j)}^{\mathrm{obs}}$ values in that leaf. That is, given any combination in $(\{\mathbf{Y}_{k}^{(t)}: k < j\}, \{\mathbf{Y}_{k}^{(t-1)}: k > j\})$, one uses the proportion of values of $\mathbf{Y}_{j}^{\mathrm{obs}}$ in the corresponding leaf to approximate the conditional distribution $(\mathbf{Y}_{(j)} \mid \mathbf{Y}_{\mathrm{obs},(j)}, \{\mathbf{Y}_{(k)}^{(t)}: k < j\}, \{\mathbf{Y}_{(k)}^{(t-1)}: k > j\})$. The iterative process again continues for T total iterations, and the values at the final iteration make up a completed dataset.

MICE-RF instead uses random forests for the univariate conditional models in MICE (e.g., Stekhoven and Bühlmann, 2012; Shah, Bartlett, Carpenter, Nicholas and Hemingway, 2014). Random forests (Ho, 1995; Breiman, 2001) is an ensemble tree method which builds multiple decision trees to the data, instead of a single tree like CART. Specifically, random forests constructs multiple decision trees using bootstrapped samples of the original, and only uses a sample of the predictors for the recursive partitions in each tree. This approach can reduce the prevalence of unstable trees as well as the correlation among individual trees significantly, since it prevents the same variables from dominating the partitioning process across all trees. Theoretically, this decorrelation should result in predictions with less variance (Hastie, Tibshirani and Friedman, 2009).

For imputation, the analyst first trains a random forests model for each $\mathbf{Y}_{(j)}$ using available cases, given all other variables. Next, the analyst generates predictions for $\mathbf{Y}_{\mathrm{mis},j}$ under that model. Specifically, for any categorical $\mathbf{Y}_{(j)}$, and given any particular combination in $(\{\mathbf{Y}_{k}^{(t)}: k < j\}, \{\mathbf{Y}_{k}^{(t-1)}: k > j\})$, the analyst first generates predictions for each tree based on the values $\mathbf{Y}_{j}^{\mathrm{obs}}$ in the corresponding leaf for that tree, and then uses the most commonly occurring majority level of among all predictions from all the

trees. For a continuous $\mathbf{Y}_{(j)}$, the analyst instead uses the average of all the predictions from all the trees. The iterative process again cycles through all the variables, for T total iterations, and the values at the final iteration make up a completed dataset. A particularly important hyperparameter in random forests is the maximum number of trees d.

For our evaluations, we use the `mice` R package to implement both MICE-CART and MICE-RF, and retain the default hyperparameter setting in the package to mimic the common practice in real world applications. Specifically, we set the minimum number of observations in each terminal leaf to 5 and the pruning threshold to 0.0001 in MICE-CART. In MICE-RF, the maximum number of trees d is set to be 10.

2.2   Generative Adversarial Imputation Network (GAIN)

GAIN (Yoon, Jordon and Schaar, 2018) is an imputation method based on GANs (Goodfellow et al., 2014), which consist of a generator function G and a discriminator function D. For any data matrix $\mathbf{Y} = (\mathbf{Y}_{\text{obs}}, \mathbf{Y}_{\text{mis}})$, we replace $\mathbf{Y}_{\text{mis}}$ with random noise, Z_{ij}, sampled from a uniform distribution. The generator G inputs this initialized data and a mask matrix $\mathbf{M}$, with $M_{ij} \in \{0,1\}$ indicating observed values of $\mathbf{Y}$, and outputs predicted values for both the observed data and missing data, $\hat{\mathbf{Y}}$. The discriminator D utilizes $\hat{\mathbf{Y}} = (\mathbf{Y}_{\text{obs}}, \hat{\mathbf{Y}}_{\text{mis}})$ and a hint matrix $\mathbf{H}$ of the same dimension to identify which values are observed or imputed by G, which results in a predicted mask matrix $\hat{\mathbf{M}}$. The hint matrix, sampled from the Bernoulli distribution with p equal to a "hint rate" hyperparameter, reveals to D partial information about $\mathbf{M}$ in order to help guide G to learn the underlying distribution of $\mathbf{Y}$.

We first train D to minimize the loss function, $L_D(\mathbf{M}, \hat{\mathbf{M}})$, for each mini-batch of size n_i:

$$L_D(\mathbf{M}, \hat{\mathbf{M}}) = \sum_{i=1}^{n_i}\sum_{j=1}^{J} M_{ij} \log(\hat{M}_{ij}) + (1 - M_{ij}) \log(1 - \hat{M}_{ij}). \tag{2.1}$$

Next, G is trained to minimize the loss function (2.2), which is composed of a generator loss, $L_G(\mathbf{M}, \hat{\mathbf{M}})$, and a reconstruction loss, $L_M(\mathbf{Y}, \hat{\mathbf{Y}}, \mathbf{M})$. The generator loss (2.3) is minimized when D incorrectly identifies imputed values as being observed. The reconstruction loss (2.4) is minimized when the predicted values are similar to the observed values, and is weighted by the hyperparameter β:

$$L(\mathbf{Y}, \hat{\mathbf{Y}}, \mathbf{M}, \hat{\mathbf{M}}) = L_G(\mathbf{M}, \hat{\mathbf{M}}) + \beta L_M(\mathbf{Y}, \hat{\mathbf{Y}}, \mathbf{M}), \tag{2.2}$$

$$L_G(\mathbf{M}, \hat{\mathbf{M}}) = \sum_{i=1}^{n_i}\sum_{j=1}^{J} M_{ij} \log(1 - \hat{M}_{ij}), \tag{2.3}$$

$$L_M(\mathbf{Y}, \hat{\mathbf{Y}}, \mathbf{M}) = \sum_{i=1}^{n_i}\sum_{j=1}^{J} (1 - M_{ij}) L_{\text{rec}}(Y_{ij}, \hat{Y}_{ij}), \tag{2.4}$$

where

$$L_{\text{rec}}(Y_{ij}, \hat{Y}_{ij}) = \begin{cases} (\hat{Y}_{ij} - Y_{ij})^2 & \text{if } Y_{ij} \text{ is continuous} \\ -Y_{ij} \log \hat{Y}_{ij} & \text{if } Y_{ij} \text{ is categorical.} \end{cases} \tag{2.5}$$

In our experiments, we model both G and D as fully-connected neural networks, each with three hidden layers, and θ hidden units per hidden layer. The hidden layer weights are initialized uniformly at random with the Xavier initialization method (Glorot and Bengio, 2010). We use leaky ReLU activation function (Maas, Hannun and Ng, 2013) for each hidden layer, and a softmax activation function for the output layer for G in the case of categorical variables, or a sigmoid activation function in the case of numerical variables and for the output of D. We facilitate this choice of output layer for numerical variables by transforming all continuous variables to be within range $(0, 1)$ using the MinMax normalization: $Y_{ij}^* = \{Y_{ij} - \min(Y_{\cdot j})\} / \{\max(Y_{\cdot j}) - \min(Y_{\cdot j})\}$, where $\min(Y_{\cdot j})$ and $\max(Y_{\cdot j})$ are the minimum and maximum of variable j, respectively. After imputation, we transform each value back to its original scale. We generate multiple imputations using several runs of the model with varying initial imputation of the missing values.

To implement GAIN in our evaluations, we use the same architecture as the one in Yoon, Jordon, and Schaar (2018). We set $\beta = 100$, θ equal to the number of features of the input data, and tune the hint rate on a single simulation. Following the common practice in the GAN literature (Berthelot, Schumm and Metz, 2017; Ham, Jun and Kim, 2020), we track the evolution of GAIN's generator and discriminator losses, and manually tune the hint rate so that the two losses are qualitatively similar. Specifically, we first coarsely select the hint rate among $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. Then we determine the final value by an additional fine tuning step. In the MAR scenario, for example, after observing that the optimal value is in the range $(0.1, 0.2)$, we perform a search among $\{0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19\}$. Finally, we set the optimal hint rate for MCAR and MAR scenarios to be 0.3 and 0.13, respectively. We train the networks for 200 epochs using stochastic gradient descent (SGD) and mini-batches of size 512 to learn the parameter weights. We use the Adam optimizer to adapt the learning rate, with an initial rate of 0.001 (Kingma and Ba, 2014).

2.3   Multiple Imputation using Denoising Autoencoders (MIDA)

MIDA (Gondara and Wang, 2018; Lu et al., 2020) extends a class of neural networks, denoising autoencoders, for MI. An autoencoder is a neural network model trained to learn the identity function of the input data. Denoising autoencoders intentionally corrupt the input data in order to prevent the networks from learning the identity function, but rather a useful low-dimensional representation of the input data. The MIDA architecture consists of an encoder and decoder, each modeled as a fully-connected neural network with three hidden layers, with θ hidden units per hidden layer. We first perform an initial imputation on missing values using the mean for continuous variables and the most frequent label for categorical variables, which results in a completed data $\mathbf{Y}_0$. The encoder inputs $\mathbf{Y}_0$, and corrupts the input data by randomly dropping out half of the variables. The corrupted input data is mapped to a higher dimensional representation by adding Θ hidden units to each successive hidden layer of the encoder. The decoder receives output from the encoder, and symmetrically scales the encoding back to the original input dimension. All hidden layers use a hyperbolic tangent (tanh) activation function, while the output layer of the decoder uses a softmax (sigmoid) activation function in the case of categorical (numerical)

variables. Multiple imputations are generated by using multiple runs with the hidden layer weights initialized as a Gaussian random variable.

Following Lu et al. (2020), we train MIDA in two phases: a primary phase and fine-tuning phase. In the primary phase, we feed the initially imputed data to MIDA and train for N_{prime} epochs. In the fine-tuning phase, MIDA is trained for N_{tune} epochs on the output in the primary phase, and produces the outcome. The loss function is used in both phases and closely resembles the reconstruction loss in GAIN:

$$L\left(Y_{ij_0}, \hat{Y}_{ij}, M_{ij}\right) = \begin{cases} \left(1 - M_{ij}\right)\left(Y_{ij_0} - \hat{Y}_{ij}\right)^2 & \text{if } Y_{ij} \text{ is continuous} \\ -\left(1 - M_{ij}\right) Y_{ij_0} \log \hat{\mathbf{Y}}_{ij} & \text{if } Y_{ij} \text{ is categorical.} \end{cases} \tag{2.6}$$

To implement MIDA in our evaluations, we use the same architecture and tune the hyperparameters in a single simulation as in Lu et al. (2020). We plot the evolution of loss function L, and select the number of additional units Θ among {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} to reduce the loss. In our experiments, we set θ equal to the number of features of the input data and add $\Theta = 7$ hidden units to each of the three hidden layers of the encoder. We train the model for $N_{\text{prime}} = 100$ epochs in the primary phase and $N_{\text{tune}} = 2$ epochs in the fine-tuning phase. Similar as in GAIN, we learn the model parameters using SGD with mini-batches of size 512, and use the Adam optimizer to adapt the learning rate with the initial rate being 0.001.

3. Simulation-based evaluation of imputation methods

Methods for missing data imputation are usually evaluated via real-data based simulations (van Buuren, 2018). Namely, one creates missing values from a complete dataset according to a missing data mechanism (Little and Rubin, 2014), imputes the missing values by a specific method, and then compares these imputed values with the original "true" values based on some metrics.

We first give a quick review of Rubin's MI combination rules. Let Q be the target estimand in the population, and $q^{(l)}$ and $u^{(l)}$ be the point and variance estimate of Q based on the lth imputed dataset, respectively. The MI point estimate of Q is $\bar{q}_L = \sum_{l=1}^{L} q^{(l)}/L$, and the corresponding estimate of the variance is equal to $T_L = (1 + 1/L) b_L + \bar{u}_L$, where $b_L = \sum_{l=1}^{L} \left(q^{(l)} - \bar{q}_L\right)^2 / (L-1)$, and $\bar{u}_L = \sum_{l=1}^{L} u^{(l)}/L$. The confidence interval of Q is constructed using $\left(\bar{q}_L - Q\right) \sim t_v \left(0, T_L\right)$, where t_v is a t-distribution with $v = (L-1)\left(1 + \bar{u}_L / \left[(1 + 1/L) b_L\right]\right)^2$ degrees of freedom.

The first step in our simulation-based evaluation procedure is choosing a dataset with all values observed, which is taken as the "population". We then choose a set of target estimands Q and compute their values from this population data, which are taken as the "ground truth". The estimands are usually summary statistics of the variables or parameters in a down-stream analysis model, e.g., a coefficient in a regression model (Tang, Song, Belin and Unützer, 2005; Huque, Carlin, Simpson and Lee, 2018). Second, we randomly draw without replacement H samples of size n from the population data, and in each of sample $(h = 1, \ldots, H)$ create missing data according to a specific missing data mechanism and pre-fixed proportion of missingness. Third, for each simulated sample with missing data, we create L imputed datasets using the imputation method under consideration and construct the point and interval estimate of

each estimand using Rubin's rules. Lastly, we compute performance metrics of each estimand from the quantities obtained in the previous step.

In the empirical application, we select a large complete subsample from the American Community Survey (ACS) – a national survey that bears the hallmarks of many big survey data – as our population. Since discrete variables are prevalent in the ACS, as well as in most survey data, we focus on the marginal probabilities of binary and categorical variables; e.g., a categorical variable with K categories has $K-1$ estimands. To evaluate how well the imputation methods preserve the multivariate distributional properties, similar to Akande et al. (2017), we also consider the bivariate probabilities of all two-way combinations of categories in binary and categorical variables. Another useful metric is the finite-sample pairwise correlations between continuous variables. For continuous variables, the common estimands are mean, median or variance. To facilitate meaningful comparisons of the results between the categorical and continuous variables, we propose to discretize each continuous variable into K categories based on the sample quantiles. We then evaluate these binned continuous variables as categorical variables based on the aforementioned estimands of marginal and bivariate probabilities.

For each estimand Q, we consider three metrics. The first metric focuses on bias. To accommodate close-to-zero estimands that are prevalent in probabilities of categorical variables, we consider the absolute standardized bias (ASB) of each estimand Q:

$$\text{ASB} = \sum_{h=1}^{H} \left| \bar{q}_L^{(h)} - Q \right| \Big/ (H \cdot Q), \tag{3.1}$$

where $\bar{q}_L^{(h)}$ is the MI point estimate of Q in simulation h.

The second metric is the relative mean squared error (Rel.MSE), which is the ratio between the MSE of estimating Q from the imputed data and that from the sampled data before introducing the missing data:

$$\text{Rel.MSE} = \frac{\sum_{h=1}^{H} \left(\bar{q}_L^{(h)} - Q \right)^2}{\sum_{h=1}^{H} \left(\tilde{Q}^{(h)} - Q \right)^2}, \tag{3.2}$$

where $\bar{q}_L^{(h)}$ is defined earlier, and $\tilde{Q}^{(h)}$ is the prototype estimator of Q, i.e., the point estimate from the complete sampled data in simulation h.

The third metric is coverage rate, which is the proportion of the $\alpha\%$ (e.g., 95%) confidence intervals, denoted by CI_h^α $(h=1,\ldots,H)$, in the H simulations that contain the true Q:

$$\text{Coverage} = \sum_{h=1}^{H} \mathbf{1} \left\{ Q \in \text{CI}_h^\alpha \right\} \Big/ H. \tag{3.3}$$

We recommend conducting a large number of simulations (e.g., $H \geq 100$) to obtain reliable estimates of MSE and coverage. This would not be a problem for deep learning algorithms, which can be typically completed in seconds even with large sample sizes. However, it can be computationally prohibitive for the

MICE algorithms when each of the simulated data is large (e.g., $n = 100,000$ in some of our simulations). In the situation that one has to rely on only a few or even a single simulation for evaluation, we propose a modified metric of bias. Specifically, for each categorical variable or binned continuous variable j, we define the weighted absolute bias (WAB) as the sum of the absolute bias weighted by the true marginal probability in each category:

$$\text{Weighted absolute bias } = \sum_{k=1}^{K} Q_{jk} \left| \bar{q}_{jk}^{(h)} - Q_{jk} \right|, \tag{3.4}$$

where K is the total number of categories, Q_{jk} is the population marginal probability of category k in variable j, and $\bar{q}_{jk}^{(h)}$ is its corresponding point estimate in simulation h. We can also average the weighted absolute bias over a number of repeatedly simulated samples.

The above procedure and metrics differ from the common practice in the machine learning literature. For example, many machine learning papers on missing data imputation conduct simulations on benchmark datasets, but these data often have vastly different structure and features from survey data and thus are less informative for the goal of this paper. One such dataset is the Breast Cancer dataset in the UCI Machine Learning Repository (Dua and Graff, 2017), which has only 569 sample units and no categorical variables. Also, these simulations are usually based on randomly creating missing values of a single dataset repeatedly rather than on drawing repeated samples from a population, and thus fails to account for the sampling mechanism. Moreover, these evaluations often use metrics focusing on accuracy of individual predictions rather than distributional features. Specifically, the most commonly used metrics are the root mean squared error (RMSE) and accuracy (Gondara and Wang, 2018; Yoon, Jordon and Schaar, 2018; Lu et al., 2020). Both metrics can be defined in an overall or variable-specific fashion, but the machine learning literature usually focuses on the overall version. The overall RMSE is defined as

$$\text{RMSE } = \sqrt{\frac{\sum_{i=1}^{n} \sum_{j} M_{ij} \left(\hat{Y}_{ij} - Y_{ij} \right)^2}{\sum_{i=1}^{n} \sum_{j} M_{ij}}}, \tag{3.5}$$

where Y_{ij} is the value of continuous variable j for individual i in the complete data before introducing missing data, and $\hat{Y}_{ij}$ is the corresponding imputed value. For non-missing values (i.e., $M_{ij} = 1$), $Y_{ij} = \hat{Y}_{ij}$. The (overall) accuracy is defined for categorical variables, namely it is the proportion of the imputed values being equal to the corresponding original "true" value:

$$\text{Accuracy } = \frac{\sum_{i=1}^{n} \sum_{j \in S_{\text{cat}}} M_{ij} \mathbf{1} \left(\hat{Y}_{ij} = Y_{ij} \right)}{\sum_{i=1}^{n} \sum_{j \in S_{\text{cat}}} M_{ij}}, \tag{3.6}$$

where S_{cat} is the set of categorical variables.

A number of caveats are in order for the RMSE and accuracy metrics. First, they are usually computed on a single imputed sample as an overall measure of an imputation method, but this ignores the

uncertainty of imputations. Second, both RMSE and accuracy are single value summaries and do not capture the multivariate distributional feature of data. Third, RMSE does not adjust for the different scale of variables and can be be easily dominated by a few outliers; also, it is often computed without differentiating between continuous and categorical variables. Lastly, when there are multiple (L) imputed data, a common way is to use the mean of the L imputed value as $\hat{Y}_{ij}$ in (3.5), but the statistical meaning of the resulting metrics is opaque. This is particularly problematic for categorical variables. For these reasons, we warn against using the overall RMSE and accuracy as the only metrics for comparing imputation methods, and one should exercise caution when interpreting them.

4. Evaluation based on ACS

In this section, we evaluate the four imputation methods described in Section 2 following the procedure and metrics described in Section 3. For simplicity, in the following discussions we use CART and RF to denote MICE-CART and MICE-RF, respectively.

4.1 The "population" data

We use the one-year Public Use Microdata Sample from the 2018 ACS to construct our population. The 2018 ACS data contains both household-level variables – for example, whether or not a house is owned or rented – and individual-level variables – for example, age, income and sex of the individuals within each household. Since individuals nested within a household are often dependent, and the imputation methods we evaluate generally assume independence across all observations, we set our unit of observation at the household-level, where independence is more likely to hold. We first remove units corresponding to vacant houses. Next, we delete units with any missing values, so that we only keep the complete cases. Within each household, we also retain individual-level data corresponding only to the household head and merge them with the household-level variables, resulting in a rich set of variables with potentially complex joint relationships.

It is often challenging to generate plausible imputations for ordinal variables with many levels when there is very low mass at the highest levels, as is the case for some variables in the ACS data. Following Li, Baccini, Mealli, Zell, Frangakis and Rubin (2014), we treat ordinal variables with more than 10 levels as continuous variables. We also follow the approach in Akande et al. (2017) to exclude binary variables where the marginal probabilities violate $np > 10$ or $n(1-p) > 10$; this eliminates estimands where the central limit theorem is not likely to hold. For each categorical variable with more than two levels but less than 10 levels where this might also be a problem, we merge the levels with a small number of observations in the population data. For example, for the household language variable, we recode the levels from five to three (English, Spanish, and other), because the probability of speaking neither English nor Spanish in the full population is less than 8.8%.

The final population data contains 1,257,501 units, with 18 binary variables, 20 categorical variables with 3 to 9 levels, and 8 continuous variables. We describe the variables in more detail in the supplementary material. We compute the population values of the estimands Q described in Section 3,

including all marginal and bivariate probabilities of discrete and binned continuous variables. We vary the size of the simulated samples from 10,000 to 100,000, and simulate missing data according to either missing completely at random (MCAR) or missing at random (MAR) mechanisms in each of these scenarios.

4.2   Simulations with $n = 10,000$

We first randomly draw $H = 100$ samples of size $n = 10,000$, and set 30% of each sample to be missing under either MCAR or MAR. CART or RF takes around 2.8 and 9.2 hours, respectively, to create $L = 10$ imputed datasets with default parameters on a standard desktop computer with a single central processing unit (CPU). The deep learning methods are much faster because they leverage GPU computing power when implemented on the GPU-enabled TensorFlow software framework (Abadi, Agarwal, Barham, Brevdo, Chen, Citro, Corrado, Davis, Dean, Devin, Ghemawat, Goodfellow, Harp, Irving, Isard, Jia, Jozefowicz, Kaiser, Kudlur, Levenberg, Mané, Monga, Moore, Murray, Olah, Schuster, Shlens, Steiner, Sutskever, Talwar, Tucker, Vanhoucke, Vasudevan, Viégas, Vinyals, Warden, Wattenberg, Wicke, Yu and Zheng, 2015). GAIN takes roughly 1.5 minutes and MIDA takes roughly 4 minutes to create $L = 10$ completed datasets using a GeForce GTX 1660 Ti GPU. Note that it is infeasible to manually tune the hyperparameter in each of the 100 simulations in each scenario for the deep learning models. So for each scenario, we have randomly selected one simulation, and tune the hyperparameters using the procedure described in Section 2. We then apply these selected hyperparameters to all simulations.

4.2.1   MCAR scenario

To create the MCAR scenario, we randomly set 30% of the values of each variable to be missing independently. Table 4.1 displays the distributions of the estimated ASB and relative MSE of all the marginal and bivariate probabilities in the imputed data by the four imputation methods.

Overall, for the estimands of marginal and bivariate probabilities of the categorical and binned continuous variables, MICE with CART significantly outperforms all other three methods, with consistently yielding the smallest ASB and relative MSE. RF is the second best, also consistently outperforming the deep learning methods. The advantage of the MICE algorithms is particularly pronounced in the upper (e.g., 75% and 90%) quantiles, indicating that GAIN and MIDA imputations have large variations over repeated samples and variables. Indeed, MIDA and GAIN lead to ultra long tails in estimating the summary statistics of the variables. For example, for bivariate probabilities of binned continuous variables, the 90% percentile of the ASB from MIDA and GAIN is approximately 20 and 27 times, respectively, of that from CART. The discrepancy is even bigger for relative MSE. There is no consistent pattern in comparing MIDA and GAIN. Specifically, for continuous variables, MIDA generally outperforms GAIN, but the difference is small except for the upper percentiles, where GAIN tends to produce very large bias and relative MSE. For categorical variables, GAIN outperforms MIDA half of the time, but again leads to the largest variation in imputations across the variables. Moreover, an interesting and somewhat surprising observation is that MICE with CART consistently outperforms RF – sometimes by a large magnitude – regardless of the choice of estimand or metric.

Table 4.1
Distributions of absolute standardized bias ($\times 100$) and relative mean squared error of all marginal and bivariate probabilities based on the imputations by the four MI methods, when $n = 10,000$ and 30% values MCAR

	Quantiles		Marginal				Bivariate			
			CART	RF	GAIN	MIDA	CART	RF	GAIN	MIDA
ASB ($\times 100$)	Cat.	10%	0.05	0.47	0.76	0.98	0.15	1.14	1.21	1.54
		25%	0.13	1.25	1.48	2.22	0.40	2.83	3.08	3.93
		50%	0.27	2.80	3.22	4.69	1.05	6.74	7.14	8.47
		75%	0.64	5.86	7.18	8.86	2.51	13.59	17.03	15.23
		90%	1.14	10.01	19.55	14.41	5.34	22.33	26.92	21.90
	B.Cont.	10%	0.06	0.24	7.25	2.73	0.19	1.30	6.05	4.80
		25%	0.10	1.05	12.86	8.36	0.43	3.24	17.61	12.01
		50%	0.21	3.59	27.30	18.51	1.02	6.61	34.29	24.07
		75%	0.43	5.43	30.21	26.84	1.90	11.76	49.38	39.54
		90%	0.81	8.49	46.41	31.36	3.42	20.79	90.90	64.65
Rel.MSE	Cat.	10%	1.05	1.67	2.50	3.38	0.96	1.11	2.75	2.98
		25%	1.16	2.40	4.97	9.03	1.08	1.61	4.33	4.75
		50%	1.37	5.99	10.37	14.89	1.25	3.35	7.40	8.16
		75%	1.49	10.25	27.73	26.16	1.48	9.07	14.87	15.80
		90%	1.62	16.22	97.33	40.16	1.89	23.91	36.37	27.92
	B.Cont.	10%	1.19	1.50	44.06	4.35	0.82	0.86	7.40	2.05
		25%	1.30	1.77	74.42	13.82	0.92	1.11	14.80	4.90
		50%	1.44	3.31	139.24	72.57	1.07	1.90	32.26	13.76
		75%	1.55	6.71	284.00	150.35	1.26	4.09	88.78	47.56
		90%	1.64	19.69	603.38	451.44	1.54	10.80	282.29	127.15

"Cat." means categorical variables and "B.Cont." means binned continuous variables.
CART = Classification and regression trees; RF = Random forests; GAIN = Generative adversarial imputation network;
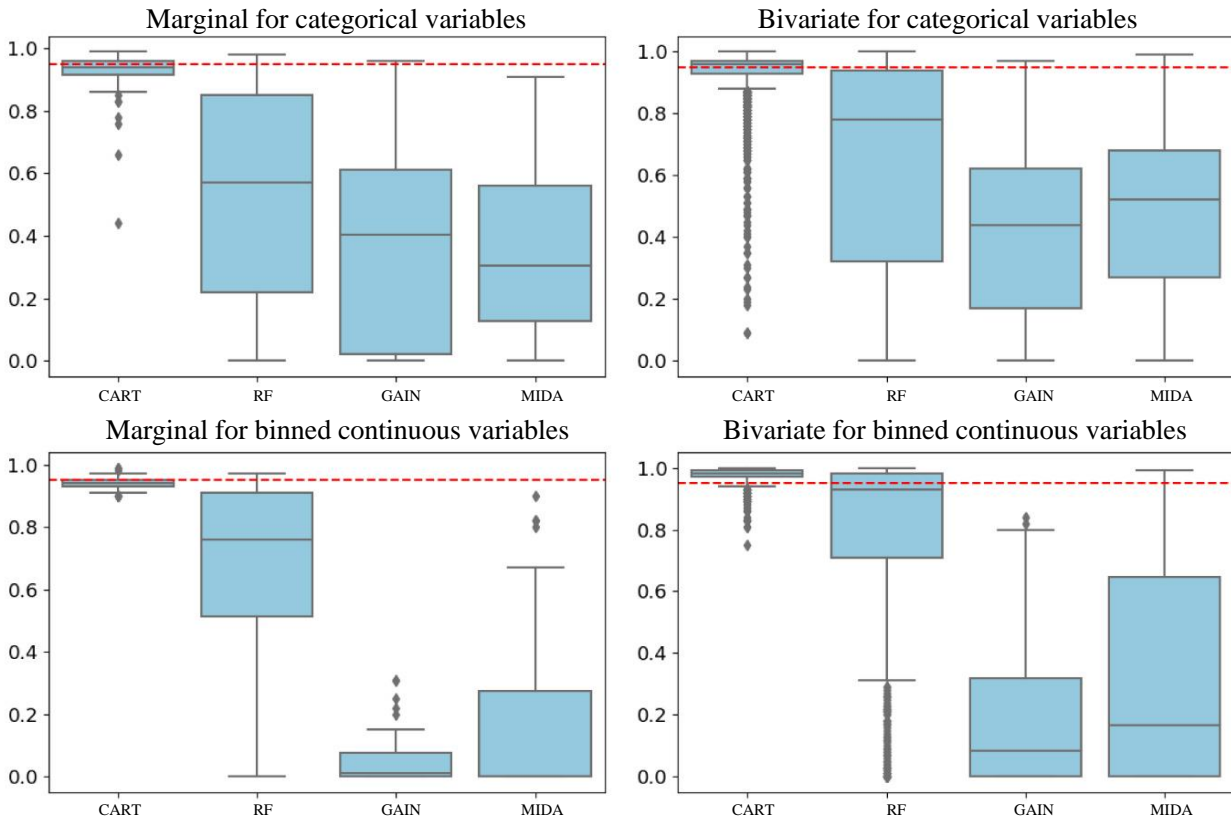MIDA = Multiple imputation using denoising autoencoders.

All methods generally yield less biased estimates (i.e., smaller ASB) of the marginal probabilities than the bivariate probabilities. This illustrates preserving multivariate distributional features is more challenging than univariate ones. The advantage of CART over the other methods is comparatively larger when estimating bivariate estimands than univariate ones. Interestingly, the relative MSE tends to be higher for the marginal probabilities than the bivariate probabilities. This is likely due to the fact that the denominator in the definition of relative MSE in (3.2) is the MSE from the sampled data before introducing missing data, which tends to be smaller for marginal probabilities than bivariate probabilities. CART yields MSEs that are very close to the corresponding MSEs from the sampled data before introducing missing data; i.e., the relative MSE is close to 1. On the contrary, both deep learning methods, and GAIN in particular, can result in exceedingly large relative MSE for many estimands.

Figures 4.1 displays the estimated coverage rates of the 95% confidence intervals for the marginal and bivariate probabilities. The patterns on coverage between different methods is similar to those on bias and MSE. Specifically, CART tends to result in coverage rates that are close to the nominal 95% level, with the median consistently being around 95% and tight interquartile range. In contrast, RF, GAIN and MIDA all result in coverage rates that are much farther off from the nominal 95% level. For example, the median coverage rates under both GAIN and MIDA are all under 0.60, and are even less than 0.30 for continuous variables. A closer look into the prediction accuracy of each variable reveals that GAIN and MIDA tend to generate imputations that are biased toward the most frequent levels, and GAIN in particular generally produces narrower intervals than the other methods. This once again provides evidence of significant bias under the deep learning methods. All methods tend to result in higher median coverage rates for the

bivariate probabilities than the marginal probabilities, although the left tails are generally longer for the former than the latter.

Figure 4.1 Coverage rate of the 95% confidence interval for all marginal and bivariate probabilities obtained from the four imputation methods in the simulations with *n* = 10,000 and 30% values MCAR.

The red dashed line is 0.95.
CART = Classification and regression trees; RF = Random forests; GAIN = Generative adversarial imputation network;
MIDA = Multiple imputation using denoising autoencoders.

4.2.2 MAR scenario

We also consider a MAR scenario, which is more plausible than MCAR in practice. We set six variables – age, gender, marital status, race, educational attainment and class of worker – to be fully observed. It would be cumbersome to specify different MAR mechanism for each of the remaining 40 variables, so we randomly divide them into three groups, consisting of 10, 15, and 15 variables. We then specify a separate nonresponse model by which to generate the missing data for the variables in each group. Specifically, we postulate a logistic model per group, conditional on the fully observed six variables, based on which we then generate binary missing data indicators for each variable in that group. This process results in approximately 30% missing rate for each of the 40 variables. We describe the models in more detail in the supplementary material.

Table 4.2 displays the distributions of the ASB and relative MSE of all the marginal and bivariate probabilities from the four methods. All methods yield larger ASB and relative MSE under the MAR scenario than the previous MCAR scenario. This is expected because MAR is a stronger assumption than

MCAR that requires conditioning on more information. Nonetheless, the overall patterns of relative performance between the methods remain the same as those under MCAR. Specifically, CART once again produces estimates with the least ASB and relative MSE – by an even larger margin then under MCAR – among the four methods, followed by RF, and then MIDA and GAIN. One notable observation is the deteriorating performance of the deep learning methods, particularly GAIN, in imputing continuous variables, sometimes resulting in several hundreds fold of relative MSE than CART. This indicates the huge uncertainties associated with GAIN in imputing continuous variables.

Table 4.2
Distributions of absolute standardized bias $(\times 100)$ and relative mean squared error for all methods, when $n = 10{,}000$ and 30% values MAR, over all possible marginal and bivariate probabilities

	Quantiles		Marginal				Bivariate			
			CART	RF	GAIN	MIDA	CART	RF	GAIN	MIDA
ASB $(\times 100)$	Cat.	10%	0.05	0.13	0.15	0.14	0.15	0.71	0.76	0.89
		25%	0.11	0.44	0.62	0.61	0.40	2.23	2.55	3.20
		50%	0.29	2.13	3.05	4.55	1.08	6.06	6.85	8.14
		75%	1.04	4.98	6.63	10.22	2.49	13.43	16.78	16.19
		90%	1.80	10.49	18.91	17.00	5.68	24.06	28.04	25.36
	B.Cont.	10%	0.07	0.29	0.33	0.33	0.27	1.17	10.87	6.18
		25%	0.17	1.07	9.64	3.13	0.69	3.49	23.67	16.26
		50%	0.67	3.14	32.86	23.85	1.58	7.83	38.52	31.17
		75%	1.20	6.95	39.57	36.09	3.40	15.20	53.59	47.34
		90%	3.40	12.39	63.45	41.99	5.94	25.16	97.47	85.44
Rel.MSE	Cat.	10%	1.00	1.00	1.00	1.00	0.97	1.00	1.53	1.93
		25%	1.08	1.82	2.56	4.75	1.04	1.39	3.78	4.03
		50%	1.33	4.33	19.03	15.13	1.25	3.00	10.42	8.38
		75%	1.72	13.08	55.07	33.36	1.59	9.56	27.45	16.95
		90%	2.27	18.70	101.91	48.44	2.23	27.44	64.01	32.85
	B.Cont.	10%	1.00	1.00	1.00	1.00	0.88	0.90	11.19	2.96
		25%	1.38	1.83	90.98	8.49	1.00	1.16	20.15	6.87
		50%	1.70	4.57	207.58	96.08	1.18	2.29	45.25	21.33
		75%	2.12	11.47	692.67	239.69	1.50	6.95	125.39	70.90
		90%	3.12	50.56	1342.23	806.43	2.12	18.07	459.78	205.14

"Cat." means categorical variables and "B.Cont." means binned continuous variables.
CART = Classification and regression trees; RF = Random forests; GAIN = Generative adversarial imputation network;
MIDA = Multiple imputation using denoising autoencoders.

Figures 4.2 displays the estimated coverage rates of the 95% confidence intervals for the marginal and bivariate probabilities, under each method. Similar as the case of bias and MSE, all methods generally result in lower coverage rates under MAR than MCAR, with visibly longer left tails in some cases, but the overall patterns comparing between the methods remain the same. Specifically, CART still tends to result in coverage rates that are above 90%, while the other three methods have consistently lower coverage rate. In particular, both GAIN and MIDA result in extremely low – below 7% – median coverage rates for continuous variables. This is closely related to the previous observation of the large uncertainty of the deep learning methods in imputing continuous variables.

Figure 4.2   Coverage rates of the 95% confidence intervals for all marginal and bivariate probabilities obtained from four methods in the simulations with *n* = 10,000 and 30% values MAR.
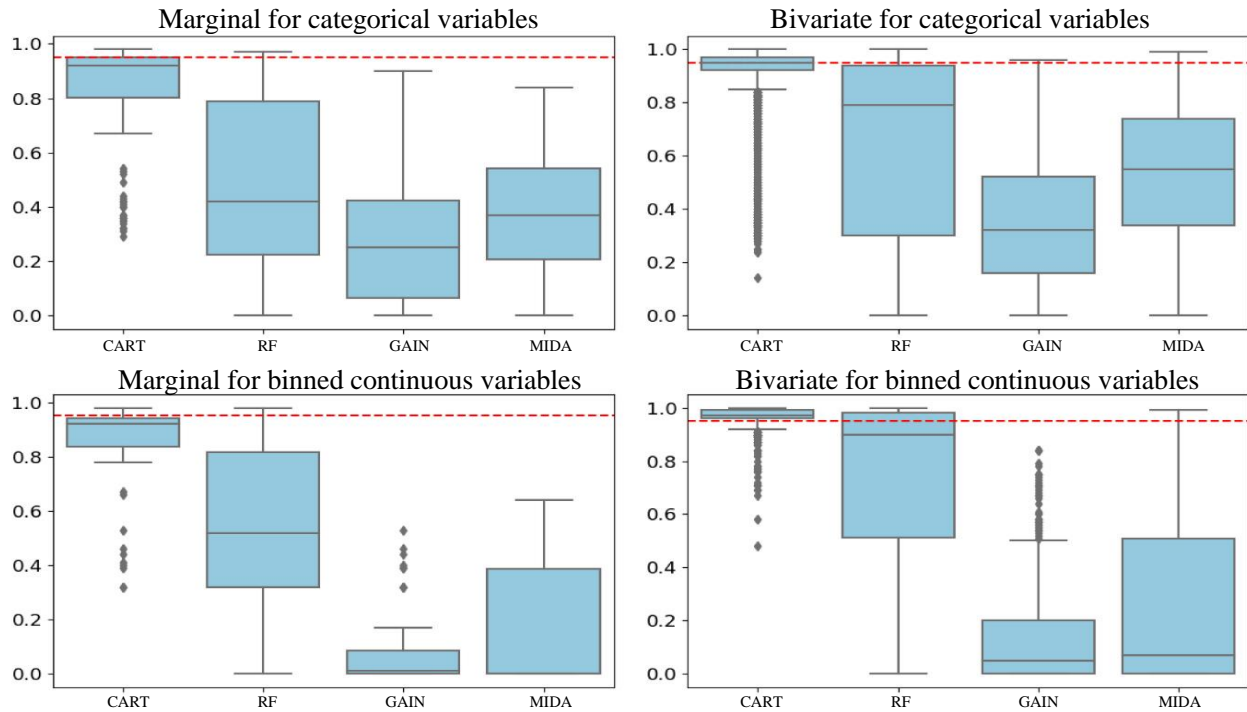
The red dashed line is 0.95.
CART = Classification and regression trees; RF = Random forests; GAIN = Generative adversarial imputation network;
MIDA = Multiple imputation using denoising autoencoders.

Finally, to illustrate that evaluating only the overall RMSE and accuracy metrics may be misleading, we display the mean and empirical standard errors of the overall RMSE and accuracy over the 100 simulations in Table 4.3, where MCAR is in the top panel and MAR is in the bottom panel. Under both missing data mechanisms, for the continuous variables, MIDA leads to the smallest overall RMSE, followed by CART, and with RF and GAIN being last. For the categorical variables, CART and GAIN lead to the highest overall accuracy, with MIDA being closely behind and RF last. These patterns, not surprisingly, differ from those reported earlier based on marginal and bivariate probabilities and different metrics. As discussed in Section 3, overall RMSE and accuracy do not capture the distributional features of multivariate data or the repeated sampling properties of the imputation methods.

Table 4.3
Overall RMSE on continuous variables and overall accuracy on categorical variables averaged over 100 simulations

Mechanism	Metric	CART	RF	GAIN	MIDA
MCAR	RMSE	0.128 (0.002)	0.159 (0.003)	0.161 (0.008)	0.112 (0.002)
	Accuracy	0.785 (0.001)	0.658 (0.003)	0.782 (0.002)	0.752 (0.004)
MAR	RMSE	0.130 (0.003)	0.154 (0.004)	0.145 (0.009)	0.110 (0.002)
	Accuracy	0.819 (0.001)	0.704 (0.003)	0.820 (0.002)	0.780 (0.007)

The empirical standard errors in the parenthesis.
The top panel is under MCAR and the bottom panel is under MAR, all with 30% missing data.
CART = Classification and regression trees; RF = Random forests; GAIN = Generative adversarial imputation network;
MIDA = Multiple imputation using denoising autoencoders.

4.3 Simulations with $n = 100,000$ and 30% MCAR

Deep learning models usually require a large sample size to train. Therefore, to give MIDA and GAIN a more favorable setting as well as to investigate the sensitivity of our results to variations in sample size, we generate a simulation scenario of $H = 10$ samples with $n = 100,000$ under MCAR. That is, we randomly set 30% of the values of each variable to be missing independently. Here we only generate 10 simulations due to the huge computational cost of MICE for samples with this size. In this scenario, we omit RF because the previous results in Section 4.2 have shown that RF is consistently inferior to CART in terms of performance and computation. We use CART, GAIN, and MIDA to create $L = 10$ completed datasets.

Because it usually requires a much larger number of simulations to reliably calculate MSE and coverage, here we focus on the weighted absolute bias metric (3.4). Table 4.4 displays the distributions of the estimated weighted absolute bias, averaged over 10 simulations, of the marginal probabilities of the categorical and binned continuous variables. Overall, the patterns comparing between the four methods remain consistent with those observed in Section 4.2. Specifically, CART again results in the smallest weighted absolute difference in both categorical and continuous variables, and the advantage is particularly pronounced with continuous variables. For example, for categorical variables, MIDA and GAIN result in a median of weighted absolute bias at least 9 and 11 times, respectively, larger than CART. The advantage of CART grows to about 30 and 60 times over MIDA and GAIN, respectively, for continuous variables. Moreover, CART performs robustly across variables, evident from the small variation in the weighted absolute bias, e.g., 0.07 for 10% percentile and 0.33 for 90% percentile among the categorical variables. In contrast, both deep learning models result in much larger variation across variables; e.g., 0.57 for 10% percentile and 2.92 for 90% percentile among the categorical variables under MIDA, and even larger for GAIN. In summary, other than computational time, MICE with CART significantly outperforms MIDA and GAIN in terms of bias and variance regardless of the sample size.

Table 4.4
Distributions of the weighted absolute bias $(\times 100)$ averaged over 10 simulated samples, each with $n = 100,000$ and 30% values MCAR

| Quantiles	Categorical			Binned Continuous		
	CART	GAIN	MIDA	CART	GAIN	MIDA
10%	0.07	0.43	0.57	0.10	5.52	1.98
25%	0.11	1.11	1.02	0.11	6.65	2.78
50%	0.15	1.74	1.40	0.12	7.36	4.04
75%	0.24	3.77	2.07	0.13	9.40	6.50
90%	0.33	4.63	2.92	0.15	11.31	7.72

CART = Classification and regression trees; GAIN = Generative adversarial imputation network;
MIDA = Multiple imputation using denoising autoencoders.

4.4 Role of hyperparameters in tree-based MICE

The pattern that CART outperforms RF is surprising, because the common knowledge is that ensemble methods are usually superior to single tree methods. But the same pattern was also observed in another

recent study (Wongkamthong and Akande, 2021). We investigate the role of the key hyperparameter in RF – the maximum number of trees d – in the simulations. We randomly selected a simulated data of size $n = 10,000$ and 30% of entries being MCAR. We use the `mice` package to fit RF with different number of trees: $d = 2, 5, 10, 15, 20$, where $d = 10$ is the default setting. The relative MSE of the imputed categorical variables fitted using each d value, as well as that using CART, is shown as trajectories in Figure 4.3, which reveals a consistent pattern: the upper quantiles – particularly those above 50% – of the relative MSE deteriorates rapidly as the maximum number of trees in RF increases, while the lower quantiles, e.g., 10%, 25%, remain stable. We found a similar pattern with the standardized bias metric and continuous variables, and thus the results are omitted here. This suggests that larger number of trees in RF – at least as implemented in the `mice` package – leads to much longer tail in the distribution of the bias and MSEs. This is likely due to overfitting. We cannot exclude the possibility that a more customized hyperparameter tuning of RF may outperform CART in some applications. However, such case-specific fine-tuning of the MICE algorithm is generally not available for the vast majority of MI consumers who relies on the default setting of popular packages like `mice`.

Figure 4.3   Quantiles of the relative mean squared error over all marginal and bivariate probabilities of categorical and binned continuous variables, under CART and RF with various number of trees, for a simulation sample with $n = 10,000$ and 30% values MCAR.
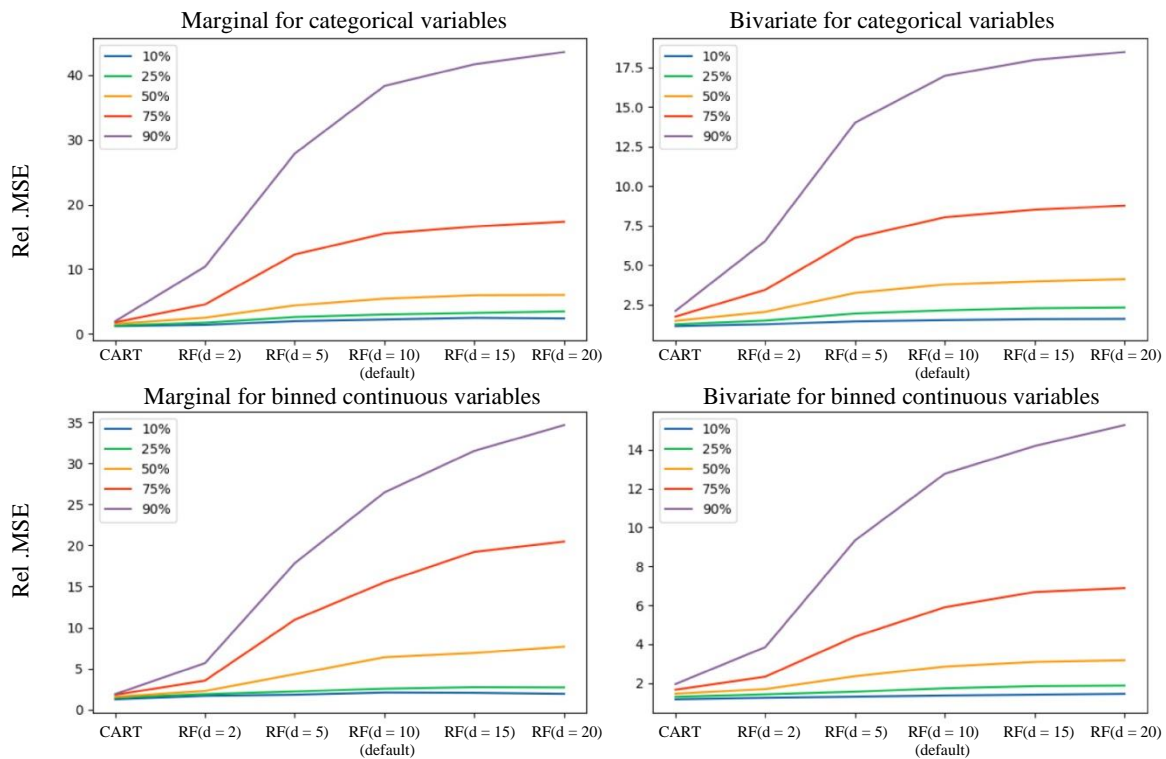
5.   Evaluation based on "benchmark" datasets

To verify the evaluations in the GAIN and MIDA papers (Gondara and Wang, 2018; Yoon, Jordon and Schaar, 2018; Lu et al., 2020), we also compared the two deep learning models with CART based on the

five benchmark datasets and simulation procedure (different from our proposed framework) used in these papers. Details of these datasets and simulations are presented in the supplementary material. The sample sizes of these data are generally not large enough to be considered as population data from which we can repeatedly sample from without replacement, so we are unable to evaluate them in a meaningful way using absolute standardized bias, relative MSE or coverage. We therefore evaluate the methods primarily on the weighted absolute bias metric. In summary, CART again consistently and significantly outperforms MIDA and GAIN in terms of weighted absolute bias for both categorical and continuous variables, across all five benchmark datasets. The difference in performance is particularly pronounced with continuous variables. We also calculated the overall MSE and accuracy as those papers did. Except for one dataset, we could not reproduce the results reported in these papers, even with the authors' code. One possible reason is that the process of tuning and selecting model hyperparameters may not be clearly documented, which is true in the present case. More details are provided in the online supplementary material.

6. Conclusion

Recent years have seen the development of many machine learning based methods for imputing missing data, raising the hope of improving over the more traditional imputation methods such as MICE. However, efforts in evaluating these methods in real world situations remain scarce. In this paper, we adopt an evaluation framework real-data-based simulations. We conduct extensive simulation studies based on the American Community Survey to compare repeated sampling properties of two MICE methods and two deep learning imputation methods based on GAN (GAIN) and denoising autoencoders (MIDA).

We find that the deep learning models hold a vast computational advantage over MICE methods, partially because they can leverage GPU power for high-performance computing. However, our simulations as well as evaluation on several "benchmark" data suggest that MICE with CART specification of the conditional models consistently outperforms, usually by a substantial margin, the deep learning models in terms of bias, mean squared error, and coverage under a wide range of realistic settings. In particular, GAIN and MIDA tend to generate unstable imputations with enormous variations over repeated samples compared with MICE. One possible explanation is that deep neural networks excel at detecting complex sub-structures of big data, but may not suit for data with simple structure, such as the simulated data used here. Another possibility is that the sample sizes in our simulations are not adequate to train deep neural networks, which usually required much more data compared to traditional statistical models.

These results contradict previous findings based on the single performance metric of overall mean squared error in the machine learning literature (e.g., Gondara and Wang, 2018; Yoon, Jordon and Schaar, 2018; Lu et al., 2020). This discrepancy highlights the pitfalls of the common practice in the machine learning literature of evaluating imputation methods. It also demonstrates the importance of assessing repeated-sampling properties on multiple estimands of MI methods. An interesting finding is that ensemble trees (e.g., RF) do not improve over a single tree (e.g., CART) in the context of MICE, which matches the findings in another recent study (Wongkamthong and Akande, 2021). Combined with the fact

that the former is more computationally intensive than the latter, we recommend using MICE with CART instead of RF in practice.

Our study has a few limitations. First, there are many deep learning methods that can be adapted to missing data imputation and all may have different operating characteristics. We choose GAIN and MIDA because both generative adversarial network and denoising autoencoders are immensely popular deep learning methods, and the imputation methods based on them have been advertised as superior to MICE. Nonetheless, it would be desirable to examine other deep learning based imputation methods in future research. Second, performance of machine learning methods is highly dependent on hyperparameter selection. So it can be argued that the inferior performance of GAIN and MIDA may be at least partially due to sub-optimal hyperparameter selection. However, practitioners would most likely rely on default hyperparameter values for any machine learning based imputation methods, which is indeed what we have adopted in our simulations and thus represents the real practice. Third, we did not consider the joint distribution between any categorical and continuous variables, but our evaluations within categorical and continuous variables have yielded consistent conclusions. Lastly, as any simulation study, one should exercise caution in generalizing the conclusions. By carefully selecting the data and metrics, we have attempted to closely mimic the settings representative of real survey data so that our conclusions are informative for practitioners who deal with similar situations. Additional evaluation studies based on different data are desired to shed more insights on the operating characteristics and comparative performances of different missing data imputation methods. Data, code, and supplementary material for the paper are available at: https://github.com/zhenhua-wang/MissingData_DL.

Acknowledgements

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y. and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems [Software available from tensorflow.org], https://www.tensorflow.org/.

Akande, O., Li, F., and Reiter, J. (2017). An empirical comparison of multiple imputation methods for categorical data. *The American Statistician*, 71(2), 162-170.

Arnold, B.C., and Press, S.J. (1989). Compatible conditional distributions. *Journal of the American Statistical Association*, 84, 152-156.

Barnard, J., and Meng, X.-L. (1999). Applications of multiple imputation in medical studies: From AIDS to NHANES. *Statistical Methods in Medical Research*, 8(1), 17-36.

Berthelot, D., Schumm, T. and Metz, L. (2017). *BEGAN: Boundary Equilibrium Generative Adversarial Networks*. CoRR, abs/1703.10717. http://arxiv.org/abs/1703.10717.

Breiman, L. (2001). Random forests. *Machine Learning*, 45, 5-32.

Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J. (1984). *Classification and Regression Trees*. Belmont, CA: Wadsworh, Inc.

Burgette, L., and Reiter, J.P. (2010). Multiple imputation via sequential regression trees. *American Journal of Epidemiology*, 172, 1070-1076.

Cao, W., Wang, D., Li, J., Zhou, H., Li, L. and Li, Y. (2018). BRITS: Bidirectional recurrent imputation for time series. *Advances in Neural Information Processing Systems*, 6775-6785.

Che, Z., Purushotham, S., Cho, K., Sontag, D. and Liu, Y. (2018). Recurrent neural networks for multivariate time series with missing values. *Scientific Reports*, 8(1), 1-12.

Chen, S., and Haziza, D. (2019). Recent developments in dealing with item nonresponse in surveys: A critical review. *International Statistical Review*, 87, S192-S218.

De Leeuw, E.D., Hox, J. and Huisman, M. (2003). Prevention and treatment of item nonresponse. *Journal of Official Statistics,* Stockholm, 19(2), 153-176.

Doove, L., Van Buuren, S. and Dusseldorp, E. (2014). Recursive partitioning for missing data imputation in the presence of interaction effects. *Computational Statistics & Data Analysis*, 72, 92-104.

Dua, D., and Graff, C. (2017). *UCI Machine Learning Repository*. http://archive.ics.uci.edu/ml.

Fortuin, V., Baranchuk, D., Rätsch, G. and Mandt, S. (2020). GP-VAE: Deep probabilistic time series imputation. *International Conference on Artificial Intelligence and Statistics*, 1651-1661.

Gelman, A., and Speed, T.P. (1993). Characterizing a joint probability distribution by conditionals. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 55, 185-188.

Glorot, X., and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Artificial Intelligence and Statistics*, 9, 249-256.

Gondara, L., and Wang, K. (2018). MIDA: Multiple imputation using denoising autoencoders. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 260-272.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, 2672-2680.

Ham, H., Jun, T.J. and Kim, D. (2020). *Unbalanced Gans: Pre-Training the Generator of Generative Adversarial Network Using Variational Autoencoder*. arXiv preprint arXiv:2002.02112.

Harel, O., and Zhou, X.-H. (2007). Multiple imputation: Review of theory, implementation and software. *Statistics in Medicine*, 26(16), 3057-3077.

Hastie, T., Tibshirani, R. and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference and Prediction* (2nd Ed.), Springer.

Haziza, D., and Vallée, A.-A. (2020). Variance estimation procedures in the presence of singly imputed survey data: A critical review. *Japanese Journal of Statistics and Data Science*, 3(2), 583-623.

Ho, T.K. (1995). Random decision forests. *Proceedings of 3rd International Conference on Document Analysis and Recognition*, 1, 278-282.

Honaker, J., King, G. and Blackwell, M. (2011). Amelia II: A program for missing data. *Journal of Statistical Software*, 45(7), 1-47.

Horton, N.J., Lipsitz, S.R. and Parzen, M. (2003). A potential for bias when rounding in multiple imputation. *The American Statistician*, 57(4), 229-232.

Huque, M.H., Carlin, J.B., Simpson, J.A. and Lee, K.J. (2018). A comparison of multiple imputation methods for missing data in longitudinal studies. *BMC Medical Research Methodology*, 18(1), 1-16.

Kingma, D., and Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. arXiv:1412.6980.

Li, F., Yu, Y. and Rubin, D. (2012). *Imputing Missing Data by Fully Conditional Models: Some Cautionary Examples and Guidelines*. Technical report, Duke University Department of Statistical Science Discussion Paper, 11-24.

Li, F., Baccini, M., Mealli, F., Zell, E.R., Frangakis, C.E. and Rubin, D.B. (2014). Multiple imputation by ordered monotone blocks with application to the anthrax vaccine research program. *Journal of Computational and Graphical Statistics*, 23(3), 877-892.

Lipton, Z.C., Kale, D.C. and Wetzel, R. (2016). Modeling missing data in clinical time series with RNNs. *Machine Learning for Healthcare*, 56.

Little, R.J., and Rubin, D.B. (2014). *Statistical Analysis with Missing Data*. Hoboken, NJ: John Wiley & Sons, Inc.

Little, R.J., and Rubin, D.B. (2019). *Statistical Analysis with Missing Data*, 3rd edition. New York: John Wiley & Sons, Inc.

Lu, H.-M., Perrone, G. and Unpingco, J. (2020). *Multiple Imputation with Denoising Autoencoder Using Metamorphic Truth and Imputation Feedback*. arXiv preprint arXiv:2002.08338.

Maas, A.L., Hannun, A.Y. and Ng, A.Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. ICML*, (1), 3.

Manrique-Vallier, D., and Reiter, J. (2014). Bayesian estimation of discrete multivariate truncated latent structure models. *Journal of Computational and Graphical Statistics*, 23, 1061-1079.

Monti, F., Bronstein, M. and Bresson, X. (2017). Geometric matrix completion with recurrent multi-graph neural networks. *Advances in Neural Information Processing Systems*, 3697-3707.

Murray, J.S., and Reiter, J.P. (2016). Multiple imputation of missing categorical and continuous values via bayesian mixture models with local dependence. *Journal of the American Statistical Association*, 111(516), 1466-1479.

Raghunathan, T.E., Lepkowski, J.M., Van Hoewyk, J. and Solenberger, P. (2001). A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey Methodology*, 27, 1, 85-95. Paper available at https://www150.statcan.gc.ca/n1/en/pub/12-001-x/2001001/article/5857-eng.pdf.

Reiter, J.P., and Raghunathan, T.E. (2007). The multiple adaptations of multiple imputation. *Journal of the American Statistical Association*, 102(480), 1462-1471.

Royston, P., and White, I.R. (2011). Multiple imputation by chained equations (mice): Implementation in Stata. *Journal of Statistical Software*, 45(4), 1-20.

Rubin, D.B. (1976). Inference and missing data (with discussion). *Biometrika*, 63, 581-592.

Rubin, D.B. (1987). *Multiple Imputation for Nonresponse in Surveys*. New York: John Wiley & Sons, Inc.

Rubin, D.B. (1996). Multiple imputation after 18+ years. *Journal of the American Statistical Association*, 91(434), 473-489.

Schafer, J.L. (1997). *Analysis of Incomplete Multivariate Data*. London: Chapman & Hall.

Shah, A., Bartlett, J., Carpenter, J., Nicholas, O. and Hemingway, H. (2014). Comparison of random forest and parametric imputation models for imputing missing data using mice: A caliber study. *American Journal of Epidemiology*, 179, 764-74.

Stekhoven, D.J., and Bühlmann, P. (2012). Missforest – non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1), 112-118.

Su, Y.-S., Gelman, A.E., Hill, J. and Yajima, M. (2011). Multiple imputation with diagnostics (mi) in r: Opening windows into the black box. *Journal of Statistical Software*, 45.

Tang, L., Song, J., Belin, T.R. and Unützer, J. (2005). A comparison of imputation methods in a longitudinal randomized clinical trial. *Statistics in Medicine*, 24(14), 2111-2128.

van Buuren, S. (2018). *Flexible Imputation of Missing Data*. CRC Press LLC.

van Buuren, S., Brand, J.P.L., Groothuis-Oudshoorn, C.G.M. and Rubin, D.B. (2006). Fully conditional specification in multivariate imputation. *Journal of Statistical Computation and Simulation*, 76(12), 1049-1064.

van Buuren, S., and Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3), 1-67.

Vincent, P., Larochelle, H., Bengio, Y. and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25ᵗʰ International Conference on Machine Learning*, 1096-1103.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A. and Bottou, L. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(12).

White, I.R., Royston, P. and Wood, A.M. (2011). Multiple imputation using chained equations: Issues and guidance for practice. *Statistics in Medicine*, 30(4), 377-399.

Wongkamthong, C., and Akande, O. (2021). A comparative study of imputation methods for multivariate ordinal data. *Journal of Survey Statistics and Methodology*, in press.

Yoon, J., Jordon, J. and Schaar, M. (2018). Gain: Missing data imputation using generative adversarial nets. *International Conference on Machine Learning*, 5689-5698.

Yoon, J., Zame, W.R. and van der Schaar, M. (2018). Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Transactions on Biomedical Engineering*, 66(5), 1477-1490.

Yuan, Y. (2011). Multiple imputation using SAS software. *Journal of Statistical Software*, 45(6), 1-25.