Data Analytics as a Service (DAaaS): Automated & Intelligent Imputation Methods for Supervised Machine Learning

By

Shahid Barkat, Joseph Kearney

Supervisor: Dr. Arnab Bose

A Capstone Project

Submitted to the University of Chicago in partial fulfillment
of the requirements for the degree of

Master of Science in Analytics

Graham School of Continuing Liberal and Professional Studies

June, 2019

The Capstone Project committee for Shahid Barkat, Joseph Kearney
Certifies that this is the approved version of the following capstone project report:

Data Analytics as a Service (DAaaS): Automated & Intelligent Imputation Methods for Supervised Machine Learning

Approved by Supervising Committee:

Dr. Arnab Bose	Dr. Sema Barlas

Abstract

The researchers develop and maintain an open-source, Python-based package named Autoimpute to address one of the most common nuisances in machine learning – datasets with missing values. The package implements numerous imputation algorithms and extends common supervised machine learning methods to handle multiply imputed datasets. Autoimpute treats missing data as a first-class citizen in the Python world, making imputation familiar to Python developers and easy to use for those new to the language. Ultimately, the package provides users an end-to-end framework that covers missing data exploration to imputation analysis.

Keywords: Missing Data; Imputation; Python; Autoimpute Package; Machine Learning; Bias/Variance Analysis; Fractional Factorial Design, SciKit Learn; Pandas

Executive Summary

This research develops an end-to-end methodology to address one of the most common nuisances in machine learning – datasets with missing values. It approaches incomplete datasets with four objectives: describe and visualize the extent of the missing value problem; examine factors related to missingness; develop methods to impute missing data; and measure the impact of imputation on inference derived from supervised learning, specifically linear and logistic regression.

To meet these objectives, the researchers develop and maintain an open-source, Python-based package named Autoimpute. The package works with pandas DataFrames and implements numerous imputation algorithms, which the authors cover later in this report. Autoimpute also extends common supervised machine learning methods to handle imputed datasets. The package includes linear and logistic regression specifically designed for multiply imputed data as well as methods to assess the impact of imputation on parameter inference from these analysis models. Further, Autoimpute follows the API design of popular Python machine learning package, scikit-learn. Its "Imputers" integrate nicely with scikit-learn machine learning pipelines.

Autoimpute treats missing data as a first-class citizen in the Python world, making imputation familiar to Python developers and easy to use for those new to the language. The package incorporates the four objectives outlined above and provides users an end-to-end framework that covers missing data exploration to imputation analysis. In doing so, Autoimpute remains flexible, yielding users as much control and complexity as they would like while they grapple with missing data.

Table of Contents

List of Figures

List of Tables

Introduction

The researchers develop and maintain an open-source, Python-based package named `Autoimpute` to address one of the most common nuisances in machine learning – datasets with missing values. The package implements numerous imputation algorithms and extends common supervised machine learning methods to handle multiply imputed datasets. `Autoimpute` treats missing data as a first-class citizen in the Python world, making imputation familiar to Python developers and easy to use for those new to the language. Ultimately, the package provides users an end-to-end framework that covers missing data exploration to imputation analysis.

Problem Statement

Machine learning models rely entirely on the data they are provided, and most require that the underlying dataset be complete. In reality, however, many real-world datasets are incomplete, containing missing values for the response variable and one or more of the features collected. As a result, the machine learning practitioner must decide what to do about missing data. The way in which the practitioner handles missing data greatly affects the interpretability of and results from the machine learning models the practitioner builds and deploys.

The challenge of handling missing data has inspired numerous imputation methods, each of which has its advantages and disadvantages depending on the nature of the missing data and the machine learning task at hand. Unfortunately, the existence of multiple imputation methods does not get the machine learning practitioner any closer to handling missing data. First, imputation methods can be challenging to understand and computationally expensive to implement. Next, no global criteria or metric exists to select the optimal imputation method given a dataset with missing values. Even if a practitioner successfully implements imputation methods, he or she has no structured way to evaluate how well imputation performs or how imputation affects supervised models downstream built upon imputed data.

Because handling missing data is quite complex, most statistical packages simply remove records with missing data so that machine learning models can execute. While this option is simple to implement and enables models to run, it generally comes with numerous undesirable side effects if data is not missing completely at random (MCAR). This subject is explored further in the background section of this paper. However, in practice, real-world

data is rarely MCAR, so models should generally avoid discarding missing records. Extensions in software packages do exist to implement imputation methods automatically. That being said, the practitioner still must decide which method to implement, explain why an imputation method is optimal, and evaluate how the optimal imputation method affects models trained on imputed data.

Research Purpose

This research aids the machine learning practitioner by bringing more clarity to the imputation process, making imputation methods more accessible and comparable, and measuring the impact imputation methods have in supervised regression and classification models. This research strives to not just automate imputation but also develop an open-source framework to structure and evaluate imputation methods within a supervised machine learning process.

To address this purpose, this research specifies four objectives:

1. Assess the extent of the missing data problem with descriptive and visual measures
2. Examine the factors related to the missingness of data
3. Deploy imputation methods and select the most appropriate methodology
4. Measure the impact of imputation to the fit, stability, bias, and variance of parameters derived from supervised models built on imputed data

The researchers meet these objectives by developing an open-source Python package that can generalize across cross-sectional and time-series datasets. Any data science professional can deploy or reuse components of the package itself. Eventually, the researchers will accept contributions from the open source community as well.

Background

One must understand what missingness is and why it exists before deciding how to handle it. The authors explore the nature of missingness using a motivating example.

For an experiment, assume that a random sample of measurements is collected from thousands of weigh scales. Also assume each weigh scale in the experiment may fail to produce measurements for three separate reasons. First, a scale might run out of batteries, in which case all measurements from that scale cease for a given period of time. Next, a scale may fail more frequently for heavier objects or items over a certain weight threshhold. And finally, a scale may stop reporting measurements when placed on a soft surface instead of a hard one (Van Buuren, 2018, ch. 1.2).

As a result of these scenarios, the random sample collected likely has many instances in which weight measurements are unobserved. How missingness in this sample is handled moving forward will depend on characteristics of the missing data itself and the process that generated missingess. The authors devote the rest of this section to concepts related to the nature of missingness and how it is handled. The authors refer back to this example to make concepts more concrete.

Basic Terminology and Notation

Before diving into concepts related to missingness, the authors establish notation used throughout this text. In doing so, the authors also introduce some basic terminology. This research follows the notation Van Buuren uses in the second edition of his book, Flexible Imputation of Missing Data. For more information, refer to Appendix B, which collects and summarizes the notation seen throughout this research.

In the introductory example, weigh scales produce a set of n measurements, some of which may be missing weight. Y denotes the n x p matrix that contains the n measurements and the p variables recorded along with the measurements. The surface of the scale (hard or soft) is an example of a discrete variable in p, while the weight measurement itself is a continuous variable in p. We can retrieve an observation within Y by specifying the row and column index corresponding to the observation's cell. To do so, we use the notation y_{ij}, where i represents the row and j represents the column associated with a scalar y in matrix Y.

Y itself represents the hypothetically complete data (Van Buuren, 2018, ch. 2.2.3), which

is comprised of Y_{obs} and Y_{mis}. Y_{obs} represents each row Y_i out of n records that have known values for each column Y_j in p. Y_{mis}, on the other hand, contains records with missing observations across any of the columns in p. Note that $Y = (Y_{obs}, Y_{mis})$. The hypothetically complete data equals the conjoined observed and missing data matrices.

It is convenient to store whether or not a cell in Y is missing in a separate matrix R. Therefore, R is an n by p matrix where all entries $r_{ij} \in 0, 1$. Any observation $r_{ij} = 1$ corresponds to a known value for y_{ij}. On the contrary, any observation $r_{ij} = 0$ corresponds to a missing value for y_{ij}. R is commonly referred to as the **missing indicator matrix**, while Y is the **complete data matrix** (Van Buuren, 2018, ch. 2.2.3).

Missing Data Mechanism

Each of the three scenarios discussed above causes a given weigh scale to produce missing data, but the underlying reason for missingness is quite different in each case. Donald Rubin describes the ways in which data can be missing (Rubin, 1976). According to Rubin (as cited in Van Buuren, 2018, ch. 1.2), every observation in a dataset has some probability of being missing. The process that governs these probabilities is called the **missing data mechanism** (Rubin, as cited in Van Buuren, 2018, ch. 1.2). The missing data mechanism generates a statistical relationship between observations and the probability of missing data (Nakagawa, 2015, pg. 83). The **missing data model** is the function that formalizes that statistical relationship (Van Buuren, 2018, ch. 2.2.4). These statistical relationships fall into one of three categories, each of which represents a different missing data mechanism (Rubin, as cited in Van Buuren, 2018, ch. 1.2).

The general expression for the missing data model is defined as:

$$P(R|Y_{obs}, Y_{mis}, \psi)$$

ψ cointains the parameters of the missing data model. This expression notes that the probability of missingness $P(R = 0)$ within a dataset depends on the observed data, the missing data, and the missing data model's parameters. (Van Buuren, 2018, ch. 2.2.4). From another lense, this expession states that the missing data model is the distribution of the missingness indicator conditional on the observed data, the missing data, and the parameters of the missing data model.

The missing data model is the function that formalizes the statistical relationship governed by a missing data mechanism. The next section examines the manifestation of the missing data model under each mechanism.

MCAR, MAR, and MNAR

Rubin popularized names for the three categories that represent the main missing data mechanisms:

- **Missing Completely at Random (MCAR)**

- **Missing at Random (MAR)**
- **Missing Not at Random (MNAR)**

In this research, the authors refer to each category by its abbreviated label. The subsections below examine each category in more detail.

Missing Completely at Random (MCAR)

MCAR is the first of the three missing data mechanisms. MCAR assumes missing values in the underlying dataset have the same probability of missingness for all cases (Gelman & Hill, 2017, pg. 530). Thus, MCAR implies that the probability of missingness within a dataset is completely unrelated to the data in question or any other observed or unobserved data. The missing data model associated with MCAR is defined as:

$$P(R = 0|Y_{obs}, Y_{mis}, \psi) = P(R = 0|\psi)$$

Note that the general expression of the missing data model reduces to a much simpler form. Under MCAR, the probability of data being missing depends on the parameters of the missing data only and not on the values of missing data itself or the value of any of the observed data (Van Buuren, 2018, ch. 2.2.4).

From the weigh scale example, MCAR governs the probability of values being missing from a scale that runs out of batteries at some point in time (Van Buuren, 2018, ch. 1.2). For simplicity, assume time is not a latent variable nor of interest in the data collection process. In this case then, the missing values produced from the scale do not depend on the weight of the object in question nor the surface used for the scale, so the probability a value is missing does not depend on any of the missing or observed data. Although MCAR is convenient, it is very restrictive and generally unrealistic, so datasets with missing values are often not MCAR in the real world (Van Buuren, 2018, ch. 2.2.4).

Missing at Random (MAR)

MAR is the second missing data mechanism. MAR occurs when the probability a given variable is missing depends on available and observed information only (Gelman & Hill, 2017, pg. 530). MAR is a weaker and more general classification of missingness than MCAR (Allison, 2012). The missing data model associated with MAR is defined as:

$$P(R = 0|Y_{obs}, Y_{mis}, \psi) = P(R = 0|Y_{obs}, \psi)$$

For MAR, the missing data model reduces because the probability of data being missing depends on the missing data model and the observed data only. Therefore, Y_{mis} does not affect the probability of missingness under MAR.

In the case of the weigh scale, MAR describes missing data that arises from the scale's placement on hard or soft surfaces. If information about the surface (hard or soft) is fully observed for each attempted weight measurement, the probability of missing measurements

then depends on available data - surface type - and thus falls under MAR (Van Buuren, 2018, ch. 1.2). Since MAR is a more general assumption than MCAR, it is more realistic to encounter in real-world datasets.

Missing Not at Random (MNAR)

Missing not at random (MNAR) is the final missing data mechanism. MNAR suggests data's "probability of being missing varies for reasons that are unknown to us" (Van Buuren, 2018, ch. 1.2). As a result, the missing data model for MNAR does not reduce:

$$P(R = 0|Y_{obs}, Y_{mis}, \psi)$$

Missing data under MNAR may depend on observed and unobserved data, so the missing data model does not simplify at all. There are two major sub-categories that capture "unobserved" within MNAR. First, missingness may depend on the actual missing values themselves (Gelman & Hill, 2017, pg. 530). When the weight of an object itself is to blame for a scale's failure to report measurements, the underlying process falls under this sub-category of MNAR because the probability of weight being missing is related to weight itself (Van Buuren, 2018, ch. 1.2). The second type of MNAR covers missingness that depends on unobserved measurements or latent variables (Gelman & Hill, 2017, pg. 530). The weigh scale example proposes three reasons a scale may generate missing measurements. These reasons do not cover countless other possibilities for why missing data may occur, such as the brand or age of a given scale. If the brand or age of a scale contributes to the probability of missing measurements but data is not available regarding a scale's brand or age, then the missing data mechanism falls under the second sub-category of MNAR.

Ignorability

The missing data mechanism underpins the assumptions one can make when handling missing data. The most important assumption is that of **ignorability**. Missingness is ignorable "if it is missing at random and the probability of a missingness does not depend on the missing information itself" (Introduction to SAS, 2017). Therefore, the missingness within a dataset is said to be ignorable if the underlying missing data mechanism is MCAR or MAR. MNAR, on the other hand, constitutes missingness that is non-ignorable.

As Van Buuren notes, "the concept of ignorability plays an important role in the construction of imputation models" (2018, ch. 2.2.6). Specifically, ignorability determines whether one can ignore the way in which data are missing prior to imputing missing data through an imputation model (Nakagawa, 2015, Pg. 85). As stated in Introduction to SAS, "the assumption of ignorability is needed for optimal estimation of missing information and is a required assumption" (2017).

To formalize the statements above, consider the general expression for an imputation model:

$$P(Y_{mis}|Y_{obs},R)$$

This expression means that the distribution of the missing data depends on the distribution of the observed data Y_{obs} and the process that generated the missing data, R (Van Buuren, 2018, ch 2.2.6). In the context of imputation, this expression suggests that the imputed values are generated conditional on the observed data and missing data mechanism.

If the missingness is ingorable, then:

$$P(Y|Y_{obs},R=1) = P(Y|Y_{obs},R=0)$$

This equality states that the distribution of the data Y is the same for both the response (observed) and non-response (missing) groups (Van Buuren, 2018, ch 2.2.6). Essentially, this equation suggests that imputation models need not consider the missing data model when creating imputations for Y_{mis}. As a result, the parameters of the missing data model, ψ, are not important if the underlying missing data mechanism is ignorable.

The importance of this equality becomes clear through an example. Take the weigh scale experiment introduced in the beginning of the Background section. Assume Y_{weight} contains weight measurements in Y, and $Y_{surface}$ specifies the surface (hard or soft) of the scale generating weight measurements.

The researcher can model weight using the observed data only within each surface:

$$P(Y_{weight}|Y_{surface} = hard, R=1); P(Y_{weight}|Y_{surface} = soft, R=1)$$

If the missing data mechanism is ignorable, then:

$$P(Y_{weight}|Y_{surface} = hard, R=1) = P(Y_{weight}|Y_{surface} = hard, R=0)$$
$$P(Y_{weight}|Y_{surface} = soft, R=1) = P(Y_{weight}|Y_{surface} = soft, R=0)$$

Therefore, the researcher can draw imputations for weight conditional on observed data only; he or she does not need to consider the missing data model within each subgroup, because the distribution for the observed and the missing data is the same (Van Buuren, 2018, ch. 2.2.6). Thus, the researcher can focus on the impact of the scale surface alone, as missing values for weight come from the same distribution as the observed.

If within each surface group the distribution for missing weights depends further on the actual mass of the object weighed, then the researcher cannot ignore the missing data model, because imputed values drawn from the observed data would be systematically different than the distribution of imputed values drawn when taking the missing data model into consideration. In this example, if weight were missing for objects that had a higher weight, imputations drawn from the observed data would systematically understate the weight of

the missing objects regardless of the surface of the scale if the missing data model is ignored. The resulting imputations would be (potentially severly) biased depending on how influential the missing data model actually is.

Therefore, the implications of ignorability are of utmost importance when building an imputation model. Most imputation models in literature assume that the missing data mechanism is ignorable. Imputation models relying on this assumption produced unbiased imputations if the assumption of ignorability holds. In the event that it does not hold, imputations may be systematically biased.

The imputation models introduced in subsequent sections operate under the assumption that the missing data mechanism is ignorable (MCAR or MAR). The important takeaway, however, is that the researcher considers the missing data mechanism responsible for generating missing data and understands the effect of the missing data model on an imputation model's ability to generate unbiased imputations. If the practicioner is aware of the potential consequences that stem from the missing data mechansim at hand, he or she is prepared to assess the benefits and drawbacks of each imputation model applied to a given dataset.

This section establishes a firm understanding of the concepts related to missing data and their implications. For an extended examination, consult Appendix C. Next, the authors review popular imputation models in literature and those supported in the `Autoimpute` package. All the models assume ignorability, so methods for non-ignorable missingness are out of scope. For interested readers, many studies cover non-ignorable missingness and its impact on the models presented in this research. **(ADD SOURCES HERE TO REFER TO)**.

Missing Data Methods

Missing data mechanisms that satisfy ignorability provide the foundation for the missing data methods explored throughout the rest of this research. With an understanding of these concepts, one can now examine the imputation methods built upon these assumptions as well as the effect of those methods on supervised analysis.

Before the authors examine missing data methods, they clarify some terminology upfront to avoid confusion later on:

- A **missing data method** is a strategy to handle missing data. The two types of methods are deletion and imputation.
- An **imputation model** is a strategy used to impute missing data. Imputation models are missing data methods within the imputation category.
- An **analysis model** is a machine learning model that a researcher wants to apply to an underlying dataset. Because analysis models require complete data, the practicioner must perform either deletion or imputation to enable an analyis model to run.

The sections below define and explore deletion and imputation models. In doing so, the authors reference the effect a given missing data method may have when subsequent analysis

is performed. That being said, the researchers devote an entire section to analysis models after a full review of missing data methods.

Deletion

One of the most popular missing data methods that data practitioners use when dealing with missing data is listwise deletion or complete-case analysis (CCA). Listwise deletion discards all rows in which at least one value in the columnspace of Y is missing for a given record (Van Buuren, 2018, ch. 1.3.1). Complete-case analysis (CCA) is relatively easy to implement and enables analysis models to run without the need for imputation because missing data has been removed. However, CCA also has its flaws. As Gelman & Hill (2017) describe, two problems arise with CCA:

1. If the units with missing values differ systematically from the completely observed cases, this could bias the complete-case analysis.
2. If many variables are included in a model, there may be very few complete cases, so that most of the data would be discarded for the sake of a simple analysis. (p. 531)

The first hypothetical results from the strong assumptions listwise deletion makes regarding the missing data mechansim. CCA requies missing data to not only be ignorable but also MCAR in most cases. Therefore, listwise deletion can result in biased estimates even if ignorability holds but MAR is not met (Introduction to SAS, 2017).

The second hypothetical addresses the impact of removing observations from a dataset. Removing records from a dataset can drastically reduce the sample size if enough records have missing data. Small sample sizes can cause problems in parameter inference from supervised machine learning model analysis. Smaller samples reduce statistical power and, as a result, inflate the standard error of analysis model coefficients. If standard error increases enough, coefficients may become statistically insignificant in an analysis model (Introduction to SAS, 2017).

Imputation

Because of these problems, researchers are cautious with listwise deletion and employ CCA as a benchmark or in specific cases where the effect from deletion is negligible. Instead, researchers turn to imputation in most cases to handle missing data. UNECE provides the following definition: “Imputation is a procedure for entering a value for a specific data item where the response is missing or unusable” (2000). Instead of discarding data, imputation retains potentially important information in the data by substituting missing values with plausible ones produced from an imputation model.

While this process seems straightforward, numerous imputation models exist and range from quite simple methods to very complex models. Furthermore, no universal evaluation metric exists to judge the accuracy or success of an imputation technique because the performance of an imputation model is often contextual, depending on the nature of the missing data beyond the missing data mechanism. Because of these reasons, practitioners must fully understand the different imputation options available to them and perform

imputation analysis to discern which method serves their use case to best meet their objectives. The next section explores the fundamentals behind different imputation methods and examines their respective strengths and weaknesses.

In general, there are two major categories of imputation methods - **univariate** and **multivariate**.

Univariate

Univariate imputation techniques focus on a single incomplete variable known as the target variable (Van Buuren, 2018, ch. 3). Univariate methods utilize observed values in the target variable to determine how to fill in missing values in the same target. Mean imputation is a popular example of a univariate method. When applied, mean imputation takes the mean of the observed features within a target variable and imputes missing values with the mean. This imputation method extends to any descriptive statistic that one can calculate from a single target variable's observed data. Additional examples include median and mode imputation, which follow a similar process but use a different statistic for imputation.

Univariate measures do not have to impute a single static value. For example, linear interpolation employs linear curve fitting to construct imputations for missing values between two observed data points. Therefore, imputations from linear interpolation depend upon the closest observed values, so the value of an imputation will differ from one section of the data to another.

The only requirement for univariate methods is that they use information contained within the observed values of the same variable they are designed to impute. Appendix D goes into greater detail of all the univariate methods that the researchers support in `Autoimpute`.

Multivariate

The second major category of imputation methods is multivariate imputation. Multivariate imputation methods rely on one or more available features to predict plausible imputations for the target variable. When an imputation model has access to multiple features within a dataset, the method can preserve the relationships between the features and the target variable (Van Buuren, 2018, ch. 4.1). While this preservation is beneficial, it is not always clear which features one should use in a multivariate imputation model. In this case, the **missing data pattern** becomes useful to know. Van Buuren (2018) states:

> The missing data pattern influences the amount of information that can be transferred between variables. Imputation can be more precise if other variables are non-missing for those cases that are to be imputed. The reverse is also true. Predictors are potentially more powerful if they have are non-missing in rows that are vastly incomplete. (ch. 4.1.2)

Since the missing data pattern shows how information can be transferred between variables, we can now calculate quantitative statistics to determine further how each variable connects

to one another. Van Buuren (2018) names the first of these statistics **Influx**. The influx coefficient I_j is defined as:

$$I_j = \frac{\sum_j^p \sum_k^p \sum_i^n (1 - r_{ij}) r_{ik}}{\sum_k^p \sum_i^n r_{ik}}$$

Influx represents the number of variable pairs (Y_j,Y_k) with Y_j missing and Y_k observed, divided by the total number of observed data points. Its value depends on the proportion of missing data of the variable, where $I_j = 0$ when a variable is completely observed and $I_j = 1$ when a variable is completely missing (Van Buuren, 2018, ch. 4.1.3). As Van Buuren notes, "for two variables with the same proportion of missing data, the variable with higher influx is better connected to the observed data, and might thus be easier to impute" (2018, ch. 4.1.3). Thus, influx is an important statistic that suggests which variables in the dataset are good candidates to be imputed using the other variables as predictors.

Van Buuren (2018) coins a similar coefficient of interest, named **Outflux**. The outflux coefficient O_j is defined as:

$$O_j = \frac{\sum_j^p \sum_k^p \sum_i^n r_{ij} (1 - r_{ik})}{\sum_k^p \sum_i^n 1 - r_{ij}}$$

The outflux coefficient O_j is the number of variable pairs with (Y_j,Y_k) with Y_j observed and Y_k missing, divided by the total number of incomplete data points. Its value indicates the potential usefulness of Y_j for imputing other variables. As with influx, outflux depends on the proportion of missing data of the variable. Unlike influx, $O_j = 1$ when a variable is completely observed, and $O_j = 0$ when a variable is completely missing (Van Buuren, 2018, ch. 4.1.3). Van Buuren describes outflux in a similar manner to influx: "For two variables having the same proportion of missing data, the variable with higher outflux is better connected to the missing data, and thus potentially more useful for imputing other variables" (ch. 4.1.3). Accordingly, outflux recommends variables that are potentially more useful as predictors when imputing missing value variables in a multivariate missing dataset.

Practitioners use the above statistics to understand the importance of and relationships between variables that contain missing values. Once the set of variables is identified, a multivariate imputation model can be specified, and predictions from that model impute missing values in a target variable. A few examples of multivariate imputation methods are:

- Linear and Logistic Regression Imputation
- Stochastic Regression Imputation
- Bayesian Regression Imputation
- Predictive Mean Matching (PMM)
- Local Residual Draws (LRD)

Appendix E provides more information regarding multivariate imputation methods available in `Autoimpute` and detail behind how each method works under the hood. `Autoimpute` implements regressions as seen in Van Buuren and implements PMM and LRD as seen in Morris et. al.

Single Imputation

Each of the imputation methods presented above replaces missing values in dataset with imputed ones. As a result, the imputed dataset is complete, containing no missing values for any observation Y_{ij} within Y. As Gelman & Hill note, "these methods keep the full sample size, which can be advantageous for bias and precision" (2017, pg. 532). In comparison to list-wise deletion, imputation methods do not discard any data; rather, they impute missing records so that no information is lost within the original dataset. As a result, statistical power is not reduced and standard errors are not inflated because the dataset retains the full sample size. That being said, these imputation methods can yield a different set of biases on their own when used via **single imputation**.

Single imputation is a process by which the researcher deploys imputation methods columnwise for each column with missing data and imputes respective missing values one time for each column. While the dataset is now complete, its use within analysis models can yield standard errors for parameter coefficients that tend to be too low (Gelman & Hill, 2017, pg. 532). Gelman expands upon the issues with single imputation:

> The intuition here is that we have substantial uncertainty about the missing values, but by choosing a single imputation we in essence pretend that we know the true value with certainty.

Each imputed value is actually a random variable that comes from a distribution of possible imputed values produced by the imputation model. When a researcher imputes a dataset using one draw from each imputed value's distribution, then the researcher replaces missing values in a dataset with point estimates. Those point estimates become, in essence, true values - indistinguishable from the originally observed variables. As a result, the imputations stemming from single imputation ignore the variability and uncertainty regarding what the true value of the missing value actually is (Gelman & Hill, 2017, pg. 532). An analysis model will treat observed and imputed values the same - as true values with no inherent variability.

Donald Rubin (as cited in Van Buuren, 2018, ch. 2) notes:

> Imputing one value for a missing datum cannot be correct in general, because we don't know what value to impute with certainty (if we did, it wouldn't be missing).

Therefore, producing a single imputation for each missing value places too much trust in the imputed values themselves and disregards the variability stemming from the imputation process. As a result, practioners turn to another framework with which they impute missing data - **multiple imputation**

Multiple Imputation

As noted in the Introduction to SAS (2017):

> Multiple imputation is essentially an iterative form of stochastic imputation. However, instead of filling in a single value, the distribution of the observed data is used to estimate multiple values that reflect the uncertainty around the true value. These values are then used in the analysis of interest, such as in a OLS model, and the results combined. Each imputed value includes a random component whose magnitude reflects the extent to which other variables in the imputation model cannot predict it's true values (Johnson and Young, 2011; White et al, 2010). Thus, building into the imputed values a level of uncertainty around the "truthfulness" of the imputed values.

Specifically, multiple imputation includes three major steps in developing a multiply imputed datasets (Allison, 2012):

1. Introduce random variation into the process of imputing missing values, and generate several data sets, each with slightly different imputed values.
2. Perform an analysis on each of the data sets using the analysis model one would have used had the dataset been complete.
3. Combine the results into a single set of parameter estimates, standard errors, and test statistics using parameter pooling techniques.

Figure 1 below visualizes the workflow described in the three steps above (Van Buuren, 2018, ch. 1.4.1).

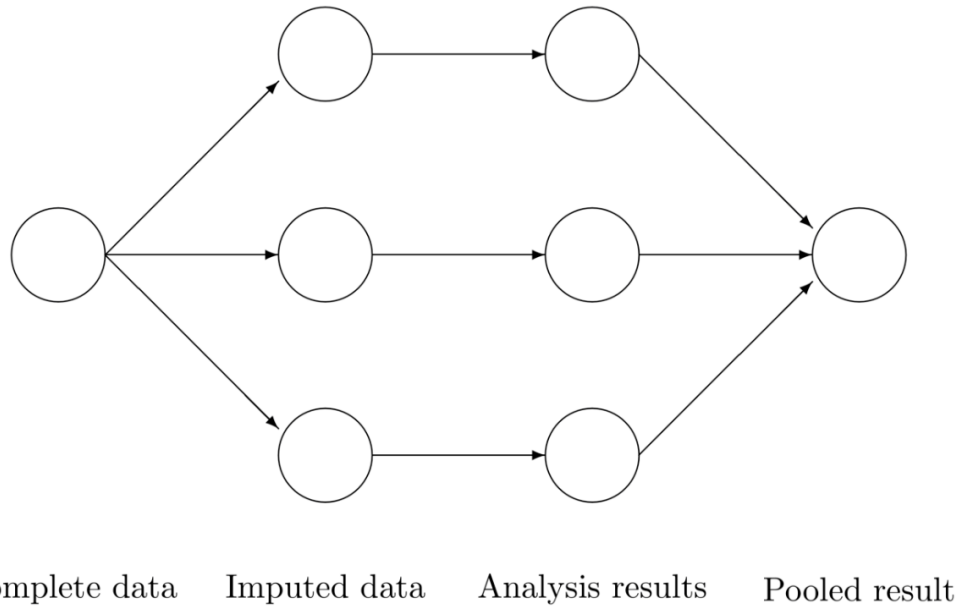

Figure 1: Multiple Imputation Workflow

In multiple imputation, practitioners should choose an imputation method that can produce sufficient variation for the imputed value. Then multiple imputation results in multiple copies of imputed datasets with different imputed values. When performing analysis on multiply imputed data, each imputed dataset is analyzed separately and then parameters

from those analyses are pooled together to get combined diagnostics on the performance of the multiple imputation process and the specified imputation model. This method "solves the standard error problem by calculating the variance of each parameter estimate across the several data sets" (Allison, 2012). The pooled parameters replace those from the supervised machine learning model of interest. The pooled parameters not only produce the coefficient estimate but also the properly account for the increase in standard error due to uncertainty introduced from imputing missing data.

The authors cover the analysis on multiply imputed data at the end of this section. Before then, the authors focus on the differences between single and multiple imputation in the context of imputation alone.

Single Imputation vs. Multiple Imputation Revisited

The example below helps visualize the inherent differences between the single and multiple imputation procedure. Imagine a subset of the complete data matrix Y from the weigh scale experiment:

$$Y_{subset} = \begin{bmatrix} soft & 60 \\ soft & NaN \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & NaN \end{bmatrix}$$

Further, assume the underlying missing data mechanism is ignorable, so imputation focuses on the observed dataset only. From the matrix above, one can discern that the distibution of weight depends on the surface of the scale. In this example, $P(Y_{weight}|Y_{surface} = soft) \sim N(60,5)$, and $P(Y_{weight}|Y_{surface} = soft) \sim N(90,5)$. The figure below visualizes the differences in the distribution of weight, conditional on the surface of the scale.

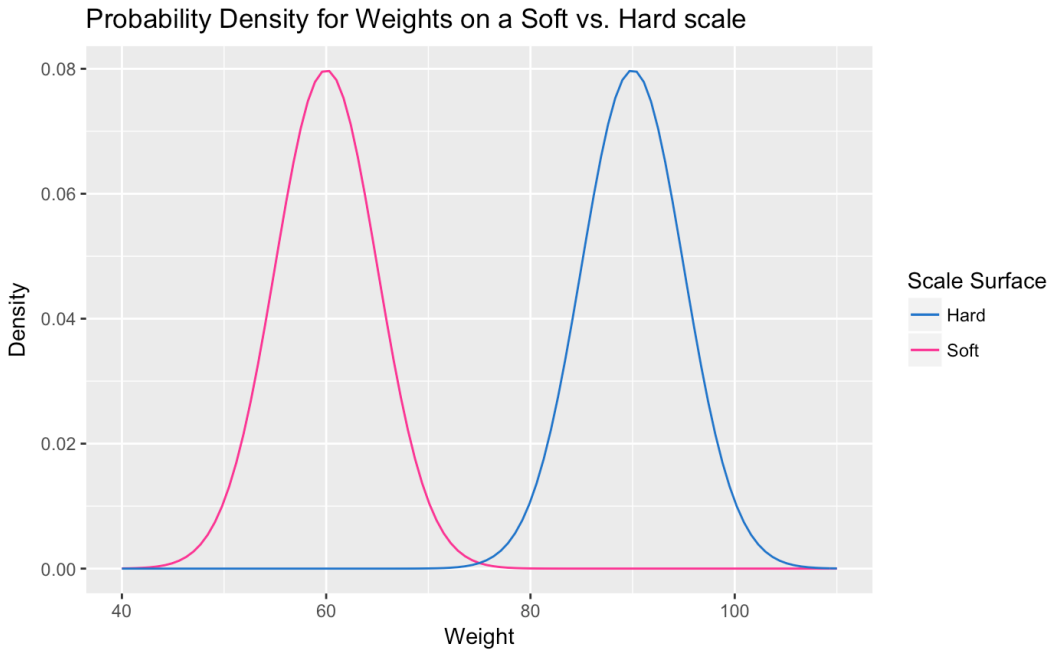

Figure 2: Distribution of Weight by Scale Surface

Now proceed with imputation analysis using single imputation. Assume that the imputation models takes a random draw from each respective surface's weight distribution to impute missing data. Then the complete matrix becomes:

$$Y_{mis} = \begin{bmatrix} soft & 60 \\ soft & NaN \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & NaN \end{bmatrix} \rightarrow Y_{complete} = \begin{bmatrix} soft & 60 \\ soft & 63 \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & 91 \end{bmatrix}$$

The imputation model imputed 63 for $Y_{2,2}$ and 91 for $Y_{6,2}$. Under single imputation, the matrix is now complete, as values for weight have been imputed. But an analysis model has access to the complete data matrix only, so it would not be able to distinguish which values were imputed and which were originally observed. Therefore, an analysis model treats the imputed values (63, 91) the same as observed values. The imputed values carry no uncertainty with them into the analysis phase.

Under multiple imputation, where $m = 3$ representing 3 imputations, the matrices may look like the below:

$$Y_{mis} = \begin{bmatrix} soft & 60 \\ soft & NaN \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & NaN \end{bmatrix} \rightarrow Y_{complete} = \begin{bmatrix} soft & 60 \\ soft & 63 \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & 91 \end{bmatrix}, \begin{bmatrix} soft & 60 \\ soft & 66 \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & 89 \end{bmatrix}, \begin{bmatrix} soft & 60 \\ soft & 59 \\ soft & 65 \\ hard & 85 \\ hard & 90 \\ hard & 94 \end{bmatrix}$$

Multiple imputation produced 3 datasets. Across each imputation, the observed have the same values, which makes sense given that the true value of these weights is known. The imputed values, however, are different. The differences reflect the uncertainty about the nature of the true value imputed, given that the true value is actually missing. Single imputation disregards this inherent variability, which tricks analysis models into believing that the imputed values are true observations. Multiple imputation retains the variability introduced by the imputation process. This research covers analysis of multiply imputed data in the next section, as there is some additional work supervised learning methods must perform under multiple imputation. That being said, it's important to take away the purpose of multiple imputation. It not only retains the full sample size but also retains the variability produced from the process of imputing missing data.

Analysis Models

The primary purpose of handling missing data is to enable analysis models to run. As previously noted, most supervised maching learning models require datasets to be complete in order to fit an analysis model capable of making predictions or classifications. The previous sections in the background address the main points a researcher must consider when dealing with missing data. All of these considerations factor into the effect that the handling of missing data has on the results produced from an analysis model. One must remember that listwise deletion and imputation methods fundamentally alter a dataset. If the structure of the data is distorted in the process, the consequences affect inference derived from analysis models fit on imputed data. This section addresses the impact of imputation and listwise deletion from the perspective of analysis. It also covers the additional work necessary to fit analysis models on multiply imputed data.

In this research, an analysis model is nothing more than a supervised machine learning methods. This text restricts focus to linear and logistic regression. Both techniques operate under the assumption that the underlying dataset is complete, containing no missing records. As a result, analysis models do not have any information regarding what happened to the dataset before the model is fit. They cannot separate imputed values from true observations, and they do not know if records were removed through list-wise deletion prior to analysis. Therefore, the inference from the analysis model depends on the way in which missing data is handled.

Analysis after Listwise Deletion

In the Deletion section, the authors explain the potential pitfalls associated with listwise deletion. Depending on the proportion of missing data, listwise deletion may significantly deplete the sample size on which an analysis model is trained. In doing so, an analysis model generates larger standard errors for parameters estimates than it would had the dataset retained every record in othe original sample. As a result, the significance of coefficients may be called into question. Additionally, if the missing data mechanism is not MCAR, listwise deletion may discard records which contain important information. Doing so may lead to biased coefficient estimates in addition to increased standard errors of those estimates.

Analysis after Single Imputation

In the Single Imputation section, the authors note that single imputation retains the full dataset but ignores any uncertainty or variation introduced when imputing missing values. Therefore, an analysis model such as linear regression treats the single-imputed dataset as if each observation is a true value. The analysis model fits the dataset as it would any other dataset.

In the best case, if the imputation model used to impute data is not misspecified, then the the coefficients produced from the analysis model should at least be unbiased estimates of the true parameters had the dataset been fully observed. In the worst case, the imputation model used to impute data is misspecified, and the imputed values do not accurately reflect the distribution of the missing data. In this case, the analysis model produces coefficients that may be biased depending on the severity of the imputation model's impact. In either situation, the standard error of the estimates produced from the analysis model will be too low (Gelman & Hill, 2017, pg. 532). When the standard error of a parameter estimate is too low, the test-statistic for that parameter will be too high, and the statistical significance of the parameter may be inflated.

Underestimated standard errors may lead a researcher to erroneously conclude that statistical relationships exist. Such inference is dangerous and undermines the reason why missing data must be handled in the first place. Even if the researcher understands the missing data mechanism and applies a proper imputation model to handle missing data, he or she may not have done enough to properly account for the uncertainty introduced from imputation. That lack of consideration appears downstream, affecting the ability for the researcher to trust the significance of analysis model parameters.

Analysis after Multiple Imputation

To handle the problems incurred from single imputation, this research suggests researchers employ imputation methods using multiple imputation. Multiple imputation creates multiple instances of the same dataset, depending on the number of imputations specified. Each imputed dataset has the same values for the observed but different values for the imputations. These differences highlight the variability introduced by imputation methods since

the true value of a missing observation is unknown.

While multiple imputation retains sample size and imputation variability, it complicates the process of analysis. Because multiple imputed datasets exist, the researcher cannot simply apply one model to the complete dataset. Insetad, the practicioner must apply the same supervised model independently to each imputed dataset and then pool the results of each analysis model to produce proper coefficient estimates and standard errors.

The authors introduce new notation to explain the pooling process used in multiple imputation. Again, the authors follow the notation outlined in Van Buuren. The notation in this section also appears in Appendix B.

Assume Q is the population parameter of interest. In this context, Q is a scalar for the β coefficient from simple linear regression or a vector $\vec{Q}$ containing the $\beta_1, ..., \beta_p$ coefficients from multiple linear regression. (Note the same notation applies to coefficients from other regression models, such as logistic regression). In either case, $\hat{Q}$ is the estimate of Q. U represents the variance of scalar Q or the covariance matrix of $\vec{Q}$ (Van Buuren, 2018, ch. 2.3.).

In the case of complete data or single imputation, the researcher fits an analysis model and produces the estimate $\hat{Q}$. But when the researcher uses multiple imputation, the researcher must apply the analysis model to each imputed datset independently. Therefore, the researcher ends up with m x $\hat{Q}$ estimates and m x U covariance matrices, one set from each analysis model applied to m imputed datasets (Van Buuren, 2018, ch. 2.3). The researcher must pool these parameters together to build one analysis model that properly accounts for the estimates from each m imputed dataset.

Pooling the coefficient estimate is the most straightforward. The equation is as follows:

$$\bar{Q} = \frac{1}{m} \sum_{\ell=1}^{m} \hat{Q}_\ell$$

In this equation, $\bar{Q}$ is the average of the $\hat{Q}$ estimates from each of the m imputed datasets. This equation reduces the coefficients from each imputed dataset down to one estimate (or vector of coefficients) by taking the average across the m imputed datasets. The resulting value is the pooled parameter estimate for the analysis model on multiply imputed data.

Pooling variance is more involved. The researcher must account for the variance from the analysis model of each imputed dataset. Additionally, the researcher must account for the variance that occurs between U from each analysis model. Lastly, the researcher must add extra variance to account for the fact that only m imputations were performed.

The first formula is the variance within:

$$\bar{U} = \frac{1}{m} \sum_{\ell=1}^{m} \bar{U}_\ell$$

This formula is called **variance within**. As with the coefficient estimate, variance within is the average of the variance within each of the analysis models fit on the m imputed datasets. It is the traditional variance metric one would expect from a linear regression, expect in the case of multiple imputation, it is the average across the m imputed datasets (Van Buuren, 2018, ch 2.3).

The next formula is the variance between:

$$B = \frac{1}{m-1} \sum_{\ell=1}^{m} (\hat{Q}_\ell - \bar{Q})(\hat{Q}_\ell - \bar{Q})'$$

This equation is called **variance between**. Because each of the m imputed datasets produces a separate set of estimates for $\hat{Q}$, variance exists between the estimates of each imputed dataset. Therefore, the analysis must account for the variance that occurs between the different estimates each imputation procedure produces (Van Buuren, 2018, ch 2.3).

The final formula represents the total variance:

$$T = \bar{U} + B + B/m$$

This equation is the **total variance** for the pooled parameter estimate of $\bar{Q}$. It adds the variance within to the variance beween to the additional variance from a finite number of imputations, m. Note that as $m \to \infty$, the last part of this equation goes to 0. But because one cannot perform an infinite number of imputations, additional variance must be added to account for the number of imputations the researcher conducts (Van Buuren, 2018, ch. 2.3).

Therefore, the analysis model for multiply imputed data is parameterized by $\bar{Q}$ and T. These parameters pool each individual analysis model applied to m imputed datasets and produce one analysis model that accurately measures the coefficient estimate and accounts for the variance that results from imputing data.

While the process is more involved, analysis of multiply imputed data fully captures the impact of missingness and the effect of imputation. The pooled parameters also provide insights into the efficiency and performance of the imputation process. For example, a large variance between metric indicates that the imputation model produces wildly different imputations for each dataset. Additional metrics and diagnostics to assess the efficiency of the imputation process appear in Appendix F. The authors also cover these diagnostics in the Methodology and Findings sections.

Missing Data and Autoimpute

This background explores the main concepts related to missing data and discusses what a researcher must consider before selecting which missing data method to use. The section then introduces numerous imputation methods, each of which operates with its own set of assumptions but all of which assume that the missing data mechanism is at least ignorable.

This text then briefly examines the impact of imputation on supervised analysis and demonstrates how concepts from missing data can leak into the inference of an analysis model if missingness is not handled with care.

The `Autoimpute` package gives the end user the tools to step through the entire process of handling missing data to performing supervised analysis. The package includes methods to explore and visualize missing data to discern what missing data mechanisms may be present. The package also includes numerous imputation methods, which the practicioner can use within the package's implementation of the Single Imputation and Multiple Imputation frameworks. Finally, `Autoimpute` extends linear and logistic regression to mulitply imputed datasets. In doing so, `Autoimpute` handles parameter pooling for the user and includes additional metrics and diagnostics to assess the impact of imputation on downstream analysis.

In the Methodology section, the authors return to the `Autoimpute` package itself and demonstrate how it is used to step through everything covered in the background section. In addition, the authors design experiments to dig deeper into the effect of specific imputation models and how they affect analysis depending on the nature of the missing data.

Methodology

In this section, the researchers use `Autoimpute` to demonstrate its capabilities as an end-to-end framework for analyzing datasets with missing values. The researchers showcase the package's features on two datasets with different types of missingness.

The researchers begin by simulating a dataset with no missingness. A simple linear regression is performed on this simulated dataset and its coefficients are stored as benchmarks for comparison of all future analysis models. Then, in each example, the researchers introduce missingness within the given dataset using a predefined missingness mechanism. This dataset with missing values becomes the source of truth for deletion and imputation methods performed on the simulated missing data.

In each example, the researchers then explore missingness patterns within the dataset. After exploration, they employ complete-case analysis or listwise deletion. Following this procedure, they use the missing value dataset to create imputations based on a number of imputation methods. They use univariate imputation methods including mean imputation and multivariate methods including least squares and predictive mean matching. They then run the processed missing value datasets through a simple linear regression and gather the respective coefficients and feature variance for comparison. Lastly, they compare the results to see the impact of deletion and imputation on the analytical model under the circumstances described in each example.

The Full Dataset

The full dataset contains 500 observations for feature x and response y. Both datasets come from a joint multivariate normal distribution, and the correlation between x and y is 0.5. The mean of each distribution is zero.

The figure below describes each feature within the distribution:

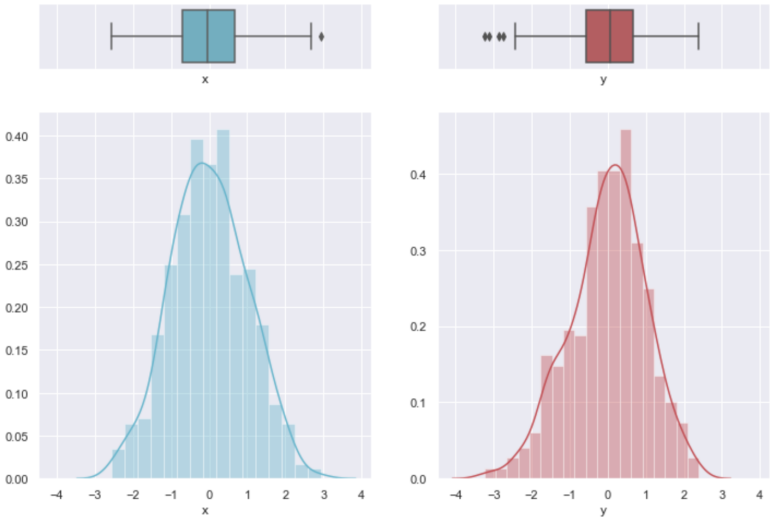

Figure 3: Distribution of Full x and y

The next figure demonstrates the joint relationship between each feature, with the marginals plotted as well:

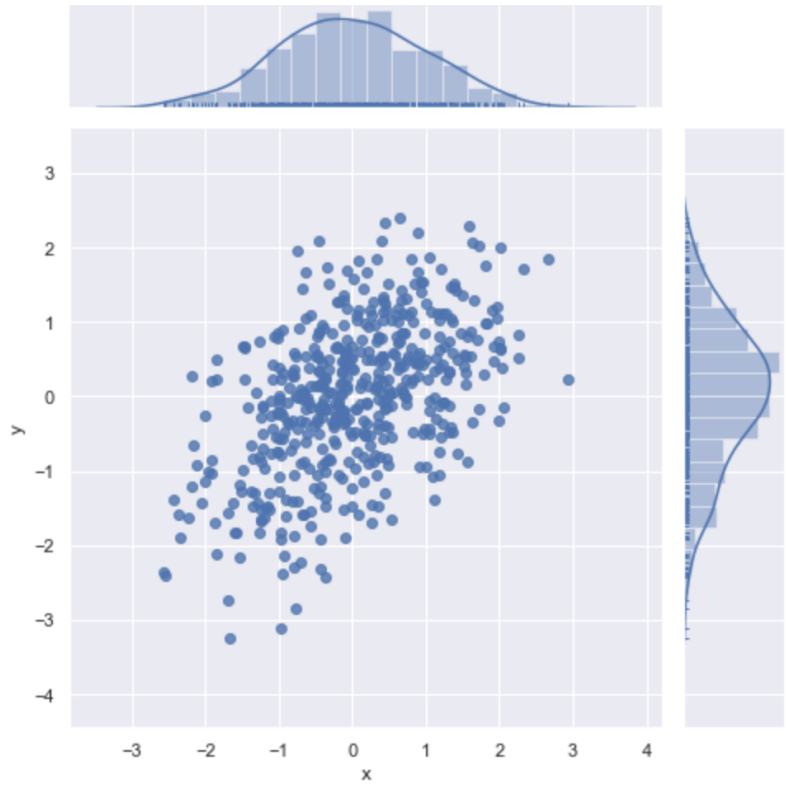

Figure 4: Joint and Marginals of Full x and y

The full dataset does not contain any missing values. Therefore, we do not have to perform

any imputation before we conduct analysis. Using `Autoimpute`, the researchers perform linear regression on the full dataset. The results appear below:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	0.0	0.03877	0.00150	0.0	0.00150	498.0	0.0	0.0	0.0
x	0.5	0.03881	0.00151	0.0	0.00151	498.0	0.0	0.0	0.0

Figure 5: Linear Regression Results: Full

The results from linear regression on the full dataset serve as the golden source.

The output above displays for each variable the following:

- `coefs` - Linear regression coefficients for each variable
- `std` - The standard error of the coefficient estimate
- `vw` - The variance of the coefficient within each complete-data sample
- `vb` - The variance between each complete-data sample
- `vt` - The total variance including vw, vb & extra simulation variance
- `dfcom` - The degrees of freedom for the hypothetically complete dataset
- `dfadj` - Adapted degrees of freedom for smaller sample sizes
- `lambda` - The proportion of variance due to nonresponse
- `riv` - The relative increase in variance

Observe that `vb`, `lambda`, and `riv` are all equal to 0. This result occurs because no imputation is necessary and no multiple imputation takes place. Therefore, we focus mainly on the `coefs` and `std` results from above as benchmarks for comparison in the analysis from each example below. These additional metrics become important when we observe examples that contain multiple imputations.

Example 1: MCAR with Missingness in the Response

In the first example, the researchers generate MCAR missingness within the response, y. Response y contains 40% missing values. Feature x remains fully observed. The two plots below showcase how to use `Autoimpute` to explore missingness within a given dataset. The plots are quite simple in this case, but they can help detect patterns in missing data when multiple features are present with different levels of missingness.

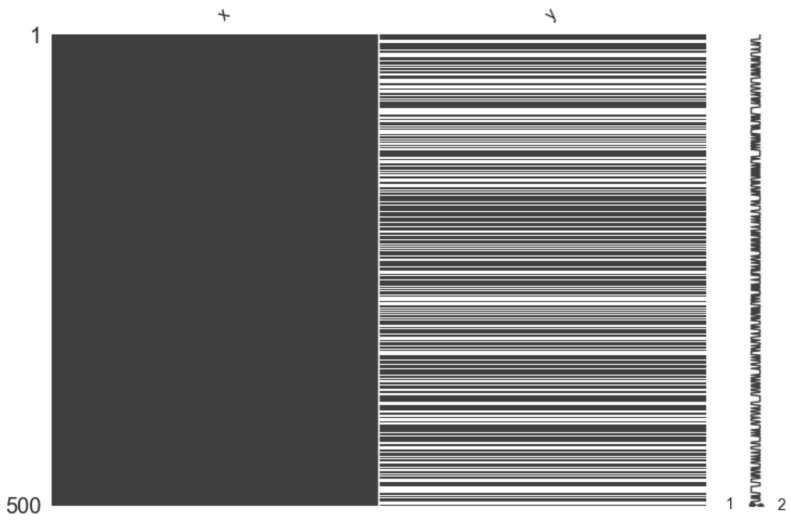

Figure 6: Missingness Locations

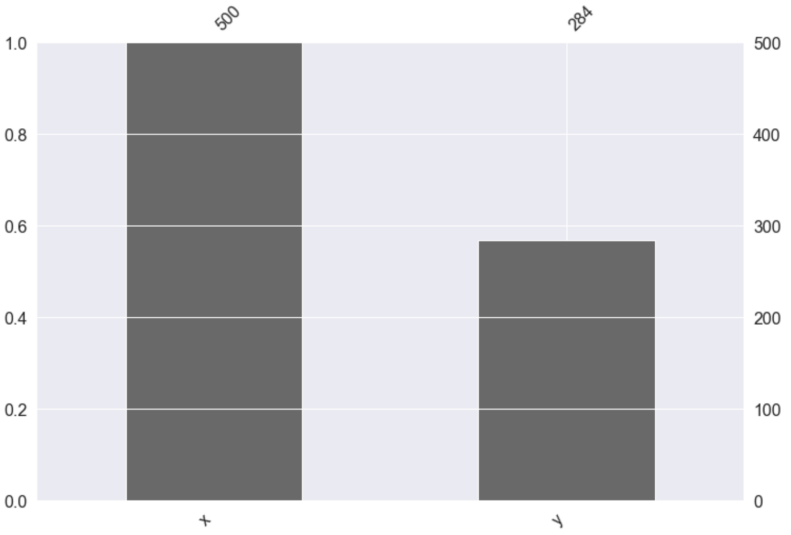

Figure 7: Missingness Percentage

Next, the researchers employ missing data methods to handle missing data. In this case, missing data methods must find plausible imputations for the 40% of y that is missing. The imputation methods used include mean, linear regression, and pmm. The researchers also use listwise deletion, although there is no visualization within `Autoimpute` for complete-case analysis because no imputations are performed. For imputation methods, the researchers deploy each strategy within the multiple imputation framework. The number of imputations performed for each method is 5.

The visualizations below show the impact of mean, linear regression, and pmm. For each strategy, there are two respective plots. The first plot is the new multivariate distribution between x and y after imputation. The second plot is a swarm plot that shows the imputations for y against other, observed values for y for each of the 5 imputations performed.

Let's start with mean imputation:

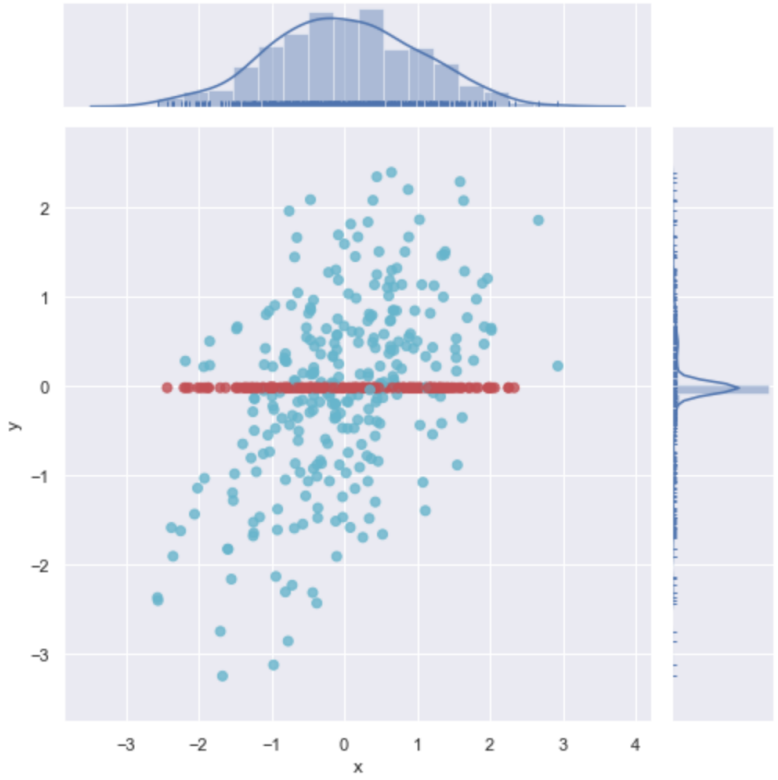

Figure 8: Joint and Marginals with Mean Imputation

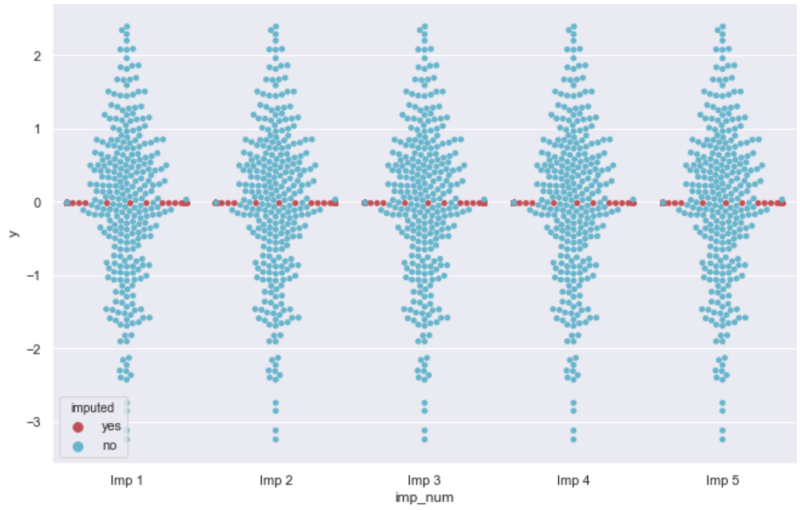

Figure 9: Swarm Plot: Mean Imputation

Note that for mean imputation, the imputations do not depend on the value of x, and the relationship between y and x is ignored. This result makes sense, as mean imputation is a univariate method. Also note that the imputation values in the swarm plot are the same for

each imputation. This occurs because the mean of the observed values of y do not change from imputation to imputation within the multiple imputation framework. Therefore, the imputation values within each imputed dataset are the same, and the imputation values accross each imputed dataset are the same.

Next, observe imputation via least squares:

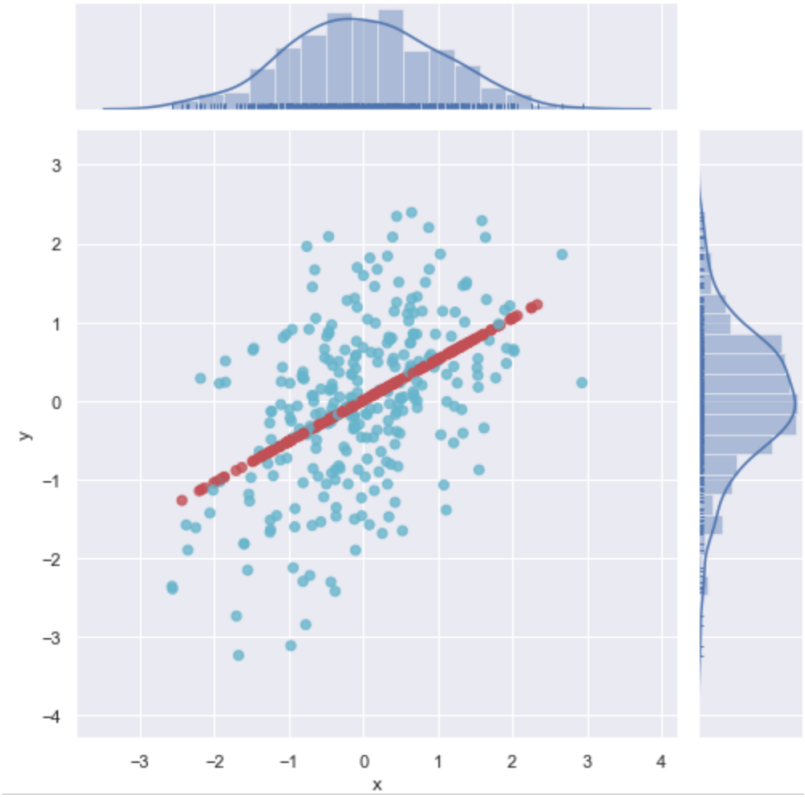

Figure 10: Joint and Marginals with Least Squares Imputation

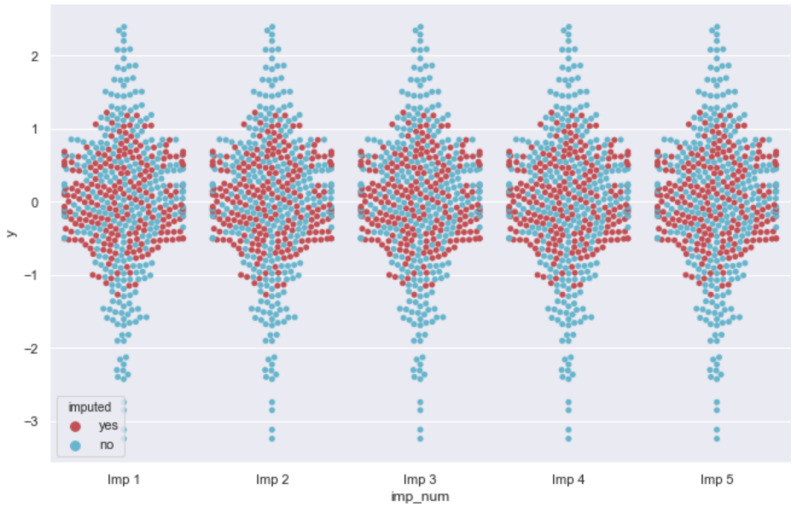

Figure 11: Swarm Plot: Least Squares Imputation

The plots for linear regression now take into account the relationship between y and x. As a result, the imputed values are different within imputations but the same across imputations. Within imputations, the linear model produces different results, which depend on the value of x. But across imputations, the linear model is the same because it is fit to the same data and no random error is added to imputations.

Finally, observe pmm imputation:

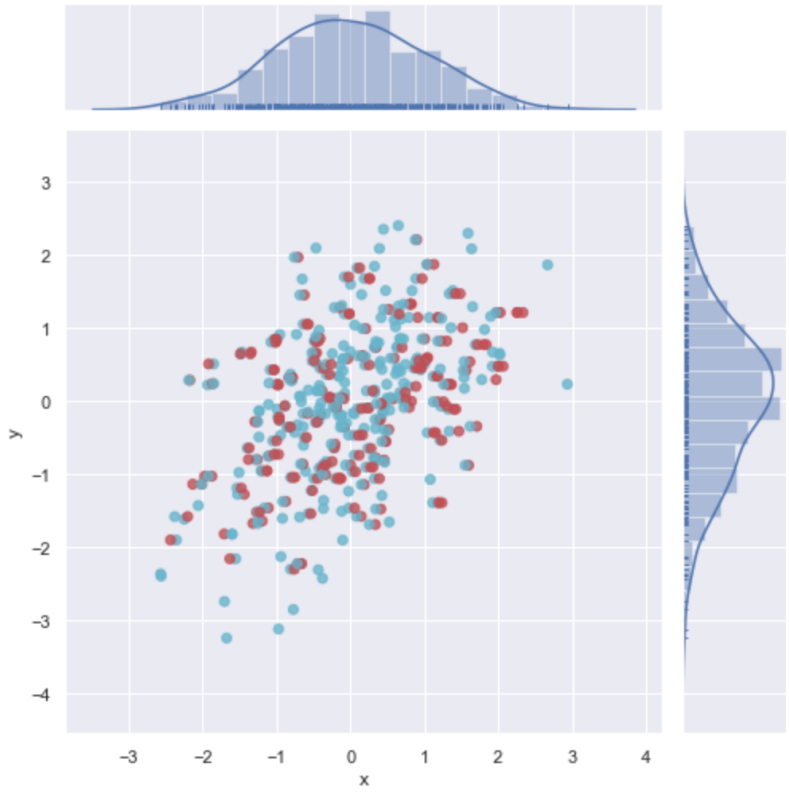

Figure 12: Joint and Marginals with PMM Imputation

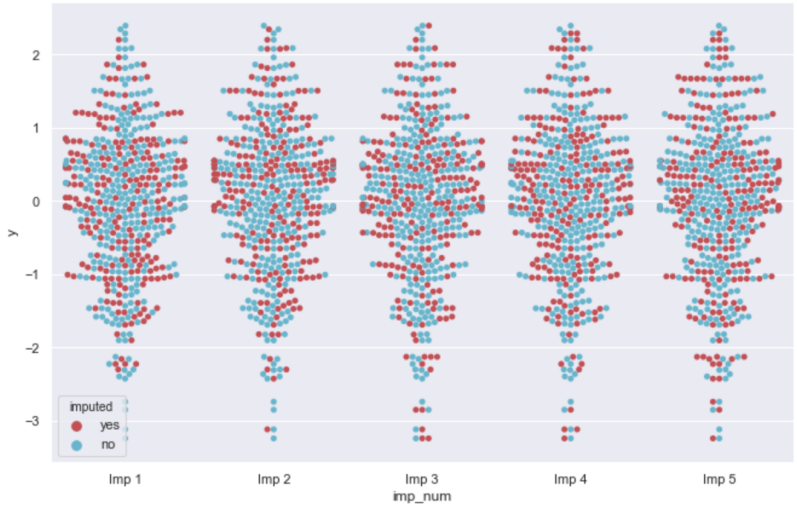

Figure 13: Swarm Plot: PMM Imputation

PMM Imputation respects not only the relationship between y and x but also the variance between the features. Imputations are no longer from the "line of best fit" as they are with linear regression. Additionally, imputations are different within and across imputations. Therefore, PMM does the best job at respecting the structure of the data and adding variance between / across imputed datasets.

Example 2: MAR with Missingness in the Predictor

In the second example, the researchers generate MAR missingness within the predictor, x. Predictor x contains 40% missing values. Response y remains fully observed. To keep this section concise, we will not generate the same plots that we did above, but we will apply the exact same methods. `Autoimpute` can impute both features and predictors, and it can examine missingness anywhere it exists within a dataset.

The researchers take the same approach to Example 2 as they do to Example 1. Multiple imputation is performed, with the number of imputations equal to 5. The same methods are applied as well (listwise delete, mean, least squares, and pmm).

Analysis Models on each Example

The sections above take the user through the data exploration phase and multiple imputation phase of `Autoimpute`. The next section, Findings, demonstrates how the nature of missingness affected the results of linear regression on our mulitply imputed data.

Findings

Full Model Recap

We begin this section with a recap of linear regression on the full dataset. The results below serve as the benchmark for comparison. They are considered the golden standard. Therefore, we seek to examine how our imputation methods perform and if they produce a dataset that significantly affects the results from linear regression. If the results are skewed, we know that the imputation model used is significantly affected by the nature of missingness within the data, and we may have to do more to account for the missingness.

The results for the full model are displayed below:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	0.0	0.03877	0.00150	0.0	0.00150	498.0	0.0	0.0	0.0
x	0.5	0.03881	0.00151	0.0	0.00151	498.0	0.0	0.0	0.0

Figure 14: Linear Regression Results: Full

Results from Example 1

Next, we examine the results from a linear regression on the imputed dataset from Example 1. Recall that Example 1 contains 40% missingness in the response y, and the missingness mechanism is MCAR.

First, listwise delete:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.00127	0.05522	0.00305	0.0	0.00305	282.0	0.0	0.0	0.0
x	0.52300	0.05737	0.00329	0.0	0.00329	282.0	0.0	0.0	0.0

Figure 15: Linear Regression Results: Listwise Delete

Next, mean imputation:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.01795	0.03341	0.00112	0.0	0.00112	498.0	495.96176	0.0	0.0
x	0.27543	0.03344	0.00112	0.0	0.00112	498.0	495.96176	0.0	0.0

Figure 16: Linear Regression Results: Mean Imputation

Next, least squares imputation:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.00127	0.03130	0.00098	0.0	0.00098	498.0	495.96176	0.0	0.0
x	0.52300	0.03133	0.00098	0.0	0.00098	498.0	495.96176	0.0	0.0

Figure 17: Linear Regression Results: Least Squares Imputation

Finally, pmm imputation:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.01811	0.05038	0.00165	0.00074	0.00254	498.0	29.84352	0.34878	0.53558
x	0.52484	0.05032	0.00166	0.00073	0.00253	498.0	30.31064	0.34589	0.52879

Figure 18: Linear Regression Results: PMM Imputation

Results from Example 2

Next, we examine the results from a linear regression on the imputed dataset from Example 2. Recall that Example 2 contains 40% missingness in the predictor x, and the missingness mechanism is MAR.

First, listwise delete:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.224408	0.046041	0.002120	0.0	0.002120	298.0	0.0	0.0	0.0
x	0.484521	0.044931	0.002019	0.0	0.002019	298.0	0.0	0.0	0.0

Figure 19: Linear Regression Results: Listwise Delete

Next, mean imputation:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	0.069300	0.042066	0.001770	0.0	0.001770	498.0	495.96176	0.0	0.0
x	0.484521	0.052657	0.002773	0.0	0.002773	498.0	495.96176	0.0	0.0

Figure 20: Linear Regression Results: Mean Imputation

Next, least squares imputation:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.020121	0.033820	0.001144	0.0	0.001144	498.0	495.96176	0.0	0.0
x	0.741911	0.038293	0.001466	0.0	0.001466	498.0	495.96176	0.0	0.0

Figure 21: Linear Regression Results: Least Squares Imputation

Finally, pmm imputation:

	coefs	std	vw	vb	vt	dfcom	dfadj	lambda	riv
const	-0.016113	0.039877	0.001426	0.000136	0.001590	498.0	204.128430	0.102982	0.114805
x	0.532586	0.044844	0.001411	0.000500	0.002011	498.0	39.788325	0.298392	0.425298

Figure 22: Linear Regression Results: PMM Imputation

Analysis of Results

We see that regardless of the missing data mechanism, mean and least squares imputation significantly reduce the variance in the coefficient estimates. Therefore, these methods should be avoided regardless because they significantly overstate the confidence we have in an estimate's coefficient. When missingness is in the predictor, least squares is quite biased. When missingness is in the response, mean is biased downward as well. When data is MCAR, listwise deletion is the most efficient and therefore preferred. When data is MAR, listwise deletion becomes slightly biased, and PMM is the most robust.

Conclusion

In summary, this project examines the nature of missingness and provides an end-to-end methodology for missing data & imputation analysis. It focuses on four key objectives: describe and visualize the extent of the missing value problem; examine factors related to missingness; develop methods to impute missing data; and measure the impact of imputation on inference derived from supervised learning, specifically linear regression. The researchers create a open-source, Python-based package called Autoimpute to accomplish these objectives. The Autoimpute package provides methods to explore missingness, implement imputation methods, and analyze their impact on analytical models in a flexible way. This report specifically focuses on establishing a solid foundation around the concepts missing data & imputation and demonstrates examples of missing data analysis with different types of missing data. Ultimately, this research provides python-focused data practitioners a tool to use to explore missingness in their datasets, perform imputation methods, and assess the impact of imputation on analytical models downstream.

Recommendations

This research explores missingness and outlines a framework through which a data practitioner can conduct missing data and imputation analysis. The researchers recommend data practitioners use the flexibility, simplicity, and granularity of the Autoimpute package to conduct their own missing data analyses. To start, data practitioners should review the Autoimpute package tutorials (See Appendix A.0) and get a better understanding of how the package works. This will help data practitioners follow best practices when utilizing the package. Once that is clear, data practitioners should clone the publicly available package from GitHub (See Appendix A.0) and start their analyses. If issues arise during your analyses, please contact the researchers. Lastly, the researchers ask that you share your feedback about the Autoimpute package or any interesting results you find in your analyses.

Appendix A

Autoimpute References

Appendix B

Notation Cheatsheet

Notation	Definition
n	The number of records or observations in a matrix
p	The number of columns or features in a matrix
Y	The complete data matrix of observed and missing records
Y_{obs}	Records within Y that are completely observed for all columns p
Y_{mis}	Records within Y with a missing value in any column in p
y_{ij}	A scalar value y in row i and column j of matrix Y
R	The missing indicator matrix, where each value $r_{ij} \in 0, 1$
r_{ij}	A scalar value r in row i and column j of matrix R

Expression	Definition
$Y = (Y_{mis}, Y_{obs})$	The complete data matrix equals the conjoined observed and missing data matrices
$r_{ij} = 0$	When a value $r_{ij} = 0$, the corresponding value y_{ij} is missing
$r_{ij} = 1$	When a value $r_{ij} = 1$, the corresponding value y_{ij} is observed

Appendix C

Concepts Related to Missingness

Concept	Definition
complete data matrix	The complete matrix Y, which is comprised of both observed and unobserved records
missing indicator matrix	The missing indicator R, which indicates which values in Y are missing or not

Appendix D

Univariate Imputation Methods

Appendix E

Multivariate Imputation Methods

Appendix F

Analysis Models and Diagnostics

References

Allison, P. D. (2012). Handling Missing Data by Maximum Likelihood. SAS Global Forum 2012. Retrieved April 23, 2019, from https://statisticalhorizons.com/wp-content/uploads/MissingDataByML.pdf.

Economic Commission for Europe of the United Nations (UNECE) (2000), "Glossary of Terms on Statistical Data Editing", Conference of European Statisticians Methodological material, Geneva.

Gelman, A., & Hill, J. (2017). Data analysis using regression and multilevel/hierarchical models. Cambridge: Cambridge University Press.

Introduction to SAS (2017). UCLA: Statistical Consulting Group. From https://stats.idre.ucla.edu/sas/modules/sas-learning-moduleintroduction-to-the-features-of-sas/ (accessed April, 21, 2019).

Morris, T. P., White, I. R., & Royston, P. (2014). Tuning multiple imputation by predictive mean matching and local residual draws. BMC Medical Research Methodology, 14(1). https://doi.org/10.1186/1471-2288-14-75

Nakagawa, S. (2015). CHAPTER 4 Missing data : mechanisms , methods , and messages.

Rubin, D. B. (1976)., Inference and missing data, Biometrika, Volume 63, Issue 3, Pages 581–592, https://doi.org/10.1093/biomet/63.3.581

Van Buuren, S. (2018). Flexible imputation of missing data. Boca Raton: Chapman & Hall/CRC.

Yu, A., Wagner, J. T., & Murugesan, M. (2017). Comparative Evaluation of Recent ML Based Missing Data Imputation Methods with NORC's Survey of Consumer Finances. University of Chicago Graham School. Retrieved November 1, 2018.