NAmored: Interactive Imputation of Missing Data

June 2022, IML Missing Data Group: Yan Hao, Talu Karagoz, David Metzger, Michael Zellinger

ABSTRACT

We present NAmored, a web app for interactively imputing missing data in stratified multivariate time series with continuous and categorical variables. The inspiration for NAmored comes from data collected in the intensive care unit (ICU) of hospitals, where multivariate time series data are recorded for each patient. NAmored follows a Visualize-Group-Impute paradigm allowing the user to conveniently choose different imputation methods for different variables in the data.

Specifically, the user first visualizes the missing data, then selects groups of variables with similar missing data patterns, and finally imputes the data by choosing an imputation method separately for each group. The user visualizes the resulting imputation performance and may repeat any of the preceding steps to optimize performance.

1 INTRODUCTION

Data collected in the real world is often incomplete. For example, patient records from a modern intensive care unit contain many gaps: patients' heart rates, blood pressures, *etc.* at specific times may be missing.

Such gaps in the data admit different interpretations. As a first example, suppose a clinician cannot measure a patient's heart rate because of a temporary equipment malfunction. The missing data later appears as completely unknown to a data analyst. In this case, the data analyst may extrapolate the missing heart rates from the measurements taken immediately before and after the equipment malfunction. However, the missing data is fundamentally unknown and adds uncertainty.

Another case arises when missing data conveys knowledge. For example, an ICU physician routinely administers different drugs to patients via injection. While the hospital continuously monitors the patient on a minute-to-minute basis, the patient may only receive injections a few times per hour. A common way to code this data consists of recording the drug dosage whenever an injection takes place. At the times when no injection takes place, no value is recorded. The resulting data set contains many missing values. Rather than implying uncertainty, these missing values express our knowledge that no injection took place.

These two examples demonstrate that different variables in the data may require different imputation schemes. In the first example, it would be appropriate to impute a missing heart rate by extrapolating from other measurements. In the second example, however, such an extrapolation would not make sense. Instead, it would be more sensible to replace all missing drug dosages with the value 0. Carrying out such heterogeneous imputation can be time-consuming even for experienced data scientists. NAmored simplifies this task by giving the user intuitive tools for grouping variables together. Following grouping, the user may conveniently apply different imputation methods to each group.

While NAmored's target user is a medical data scientist analyzing stratified time series data from a hospital's intensive care unit (ICU), our conceptual innovations are generalizable to other domains. For example, many forms of economic data assume the same mathematical structure as ICU data (stratified multivariate time series).

Figure 1: Feature Gradients

2 PARADIGM

NAmored follows a Visualize-Group-Impute paradigm for interactive missing data imputation.

After loading a data set, the user first visualizes the missing value patterns in the data. Second, the user groups together variables that share similar missing value patterns or have similar meaning in the application domain. After grouping the variables, the user can select a different imputation method for each group of variables. NAmored then allows the user to impute the missing data on a per-group basis.

We visualize the results of missing data imputation by graphically displaying imputation errors on a per-group basis. This display allows the user to assess the quality of the imputation. At this stage, the user can either download an imputed copy of the original data set, or else select different groupings and imputation methods to achieve higher performance.

In the next section, we describe the methods we employ for visualization, grouping, and imputation.

3 METHODS

In this section, we describe the graphical and statistical techniques we employ to help the user visualize missing data, group similar variables, and impute missing data on a per-group basis.

3.1 Visualization

Feature Gradients: To provide the user with a quick overview over missing value patterns, we display so-called *feature gradients* for each variable. Such a gradient visualizes the amount of missing data for a variable in a stratified multivariate time series.

In the example of data from the intensive care unit (ICU), we observe for each patient many measurements at different time points. Some of these measurements may be missing. Hence, for each patient we compute a percentage of available values *PctAvail*. We represent this percentage as a thin rectangular strip colored in grayscale according to its magnitude between 0 and 1. Arranging such strips for all patients and ordering them by their grayscale color (*PctAvail*) results in a gradient from black to white.

This *feature gradient* displays the overall amount of missing data: if the gradient is mostly white, then we have 100% of all data for most patients. Conversely, if the gradient is mostly black or dark gray, we lack most data for most patients.

Feature Histograms: We allow the user to click on the feature gradient for a particular variable to visualize its missing data pattern in greater depth. Specifically, we display the full histogram of *PctAvail* values over all the patients in the dataset.

Feature Squares: This visualization panel is an experimental feature of our app. It allows the user to study any variable in detail by visualizing all available data for this variable. For each patient, all

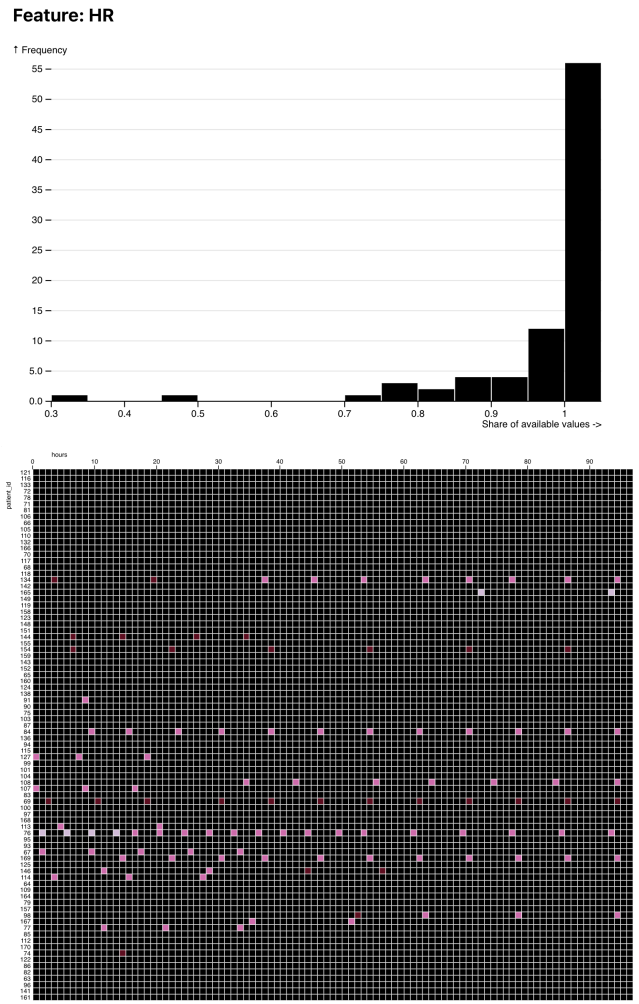

Figure 2: Histogram and Feature Squares (detailed data view, missing values are black)

measurements are shown as small colored squares ordered by time. Missing values are colored black and available values are colored according to their value.

3.2 Grouping

NAmored allows the user to group variables either manually or automatically.

Manual Grouping: NAmored features a convenient interface for grouping variables together. After creating a group, the user can add variables to this group from a list of variables. This list displays the feature gradient for each variable next to its name.

To make adding variables easier, we offer string-based selectors that allow adding variables to groups in bulk. Specifically, the user may add all variables containing, starting with, or ending with a certain string.

To illustrate the benefit of these string selectors, imagine a data set from the intensive care unit. The variables representing drug treatments may all end in `_iv` or `_bol` to indicate an intravenous (IV) infusion or bolus injection. To group all drug treatments together, a user of NAmored can create a group "Drug Treatments" and use string selectors `startsWith(_iv)` and `endsWith(_bol)` to select all relevant variables.

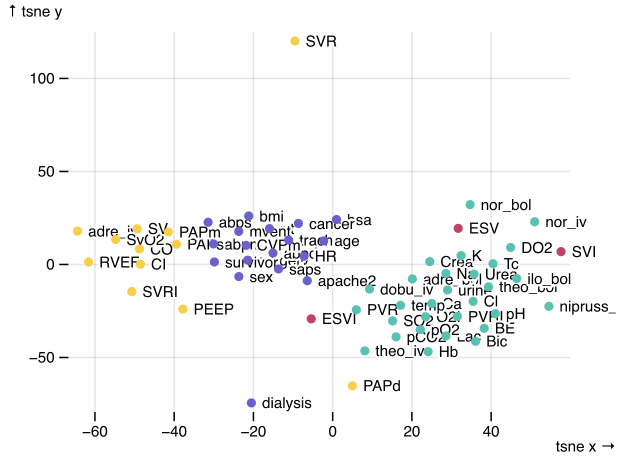

Figure 3: TSNE-Scatterplot of the different features (medical variables), obtained from the transformed data set with ones for available and zeros for missing data points

Automatic Grouping: In some cases, a data analyst does not have sufficient domain knowledge to manually partition variables into sensible groups. Even if such domain knowledge is available, automatic methods can reveal additional insights. For this reason, NAmored supports automatic clustering of the variables based on their missing value patterns. Specifically, we run the K-Means algorithm on a binary encoding of the data whereby each variable is replaced by a vector of 0's and 1's of the same dimensionality, where each entry is 0 or 1 depending on whether the corresponding value is missing. To run K-Means on these missing value patterns, we first estimate the number of clusters using a stability-based approach.

Visualization of Grouping: We visualize the user-defined grouping of the variables by embedding the missing value patterns for each variable in 2D. We use the t-SNE algorithm for this purpose. In the resulting scatterplot, each dot represents a variable and is colored according to group membership. Clear separation of the different groups indicates a high-quality grouping. For now, we only display this t-SNE scatterplot for the automatic grouping derived from clustering. However, in future work we aim to continuously update the scatterplot to reflect a user's manual changes to the grouping.

3.3 Imputation

NAmored allows the user to apply a different imputation method to each user-defined group of variables. In addition, the app estimates and visualizes the imputation error for each variable.

Imputation Methods: NAmored supports a range of popular imputation methods. These include basic techniques such as forward-fill, fill-by-constant, or fill-by-mean, as well as more sophisticated approaches such as k-nearest-neighbor imputation and multiple imputation. Multiple imputation is called "iterative" in NAmored following the implemention in Python's `scikit-learn` package.

Imputation Errors: We estimate the imputation error for each variable by artificially removing 10% of available data and imputing it with the chosen method. To quantify the error, we compute a root-mean-square error $\sqrt{\frac{1}{|S|}\sum_{i\in S}(\hat{y}_i - y_i)^2}$, where S is a random subsample containing 10% of available data. To make errors for different variables more comparable, we normalize each root-mean-square error (RMSE) by the mean absolute value of the corresponding variable. The resulting normalized RMSE expresses imputation error relative to the average magnitude of a variable.

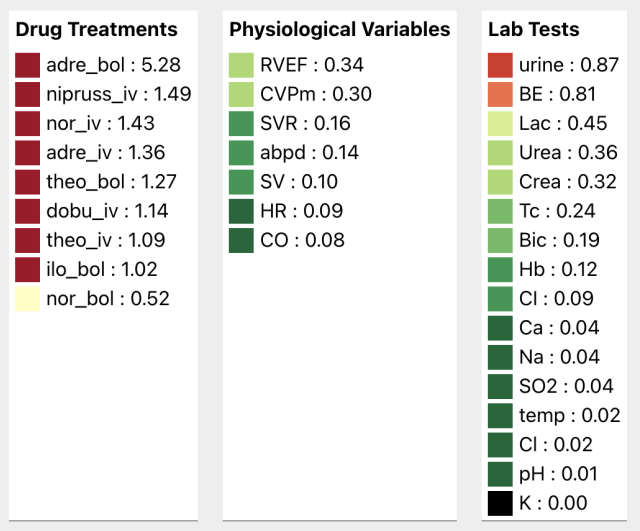

Figure 4: Imputation errors for different groups of features visualized using a red-green color scale

4 Conclusion

NAmored simplifies missing data imputation by adopting an innovative Visualize-Group-Impute paradigm.

In the visualization stage, the user examines the patterns of missing values in the data. In the grouping stage, the user partitions variables into meaningful groups based on insights from automatic clustering and domain knowledge. In the imputation stage, the user selects an imputation method for each group and assesses the estimated imputation errors.

The main benefit of NAmored lies in making heterogeneous imputation simple and convenient. Indeed, such analyses are often time-consuming even for seasoned data analysts. Besides allowing such analysts to save time, NAmored opens up the possibility of heterogeneous imputation for data workers with comparably weak programming skills. For such workers, carrying out an equivalent analysis in R or Python would be a significant challenge.

Acknowledgments

Michael J. Zellinger thanks Peter Bühlmann for his support. We all thank Menna El-Assady and her TAs for the support and feedback on the project.

5 Members' Contribution Statements

5.1 David

- did the web design, write most of the HTML and CSS
- designed the detail visualisation with the tiny squares on the visualize page
- took the role of the team leader in some of the weeks
- proposed the data structure for the communication between back- and front-end
- programmed big parts of the front-end code for the group page
- made the d3/observable visualization templates available for use in react
- displayed the imputation errors on the impute page
- made sure we comply to the submission criteria
- did a lot of debugging

5.2 Michael

- propose project
- help organize the group
- suggest feature gradient visualization
- suggest three-tab layout [Visualize — Group — Impute]
- suggest automatic clustering pipeline with tSNE visualization
- push string-based selectors
- implement panel for uploading data
- implement string-based selectors on group page
- implement menu for selecting imputation options
- propose clean software architecture according to which we rewrote our code base at around the halfway point
- encourage clean coding practices
- suggest the name "NAmored" for the app
- write the group's report

5.3 Yan

- worked on clustering pipeline of TENS dimension reduction
- did the scatterplot visualization of automatic clustering results
- designed and implemented gradient visualization
- implemented download functionality for the imputed data
- was team leader for some of the weeks
- made react class instead of function components for the whole app
- worked on the communication between the backend and the frontend to get clusters on the frontend side
- came up with idea to use read and green color to represent the imputation error

5.4 Talu

- designed clustering pipeline
- implemented different clustering algorithms
- analyzed best practices for clustering the missing data
- created different heuristic approaches to data transformations for clustering
- implemented percentage availability per patient of all data
- implemented imputation methods as objects
- came up with and implemented imputation evaluations
- parallelized imputations for efficient computation
- came up with binary string imputation class to deal with non-number data with a different evaluation metric
- implemented asynchronous data storage from frontend to back-end
- implemented communication between frontend and backend on the backend side