Data Imputation Methods for Orthogonal Data Sets

Eilif B. Mikkelsen - Rochester Institute of Technology

1 Abstract

Extensive research and clear guidelines have emerged for the imputation of variable sets that exhibit interdependence that can be effectively modeled. Multiple imputation and its derivative methods serve as the state of the art for data sets with conforming characteristics. Little guidance can be found for orthogonal feature sets. Through a factorial experiment, the shape of model degredation is measured in the presense of various levels of missing data, data relationship characteristics, and imputation methods. This paper proposes a series of reccomendations for data imputation on datasets that show little to no multicolinearity.

2 Introduction

Handling missing data effectively Datasets often have missing data in some or all of the features of interest. Given a data set with independent regressors that can be described using ordinary least squares linear regression, what is the best way to fill in any missing data.

There is extensive literature on how to leverage feature relationships to fill in missing data however there is little information on the best practices for data imputation of independent data sets. The objective of this study is to quantify the impact of various data imputation methods on the model performance and provide recommendations on best practices for independent datasets.

3 State of Research

4 Methodology

The methods explored in this paper were implemented using the Python/Numpy/Pandas stack with the Statsmodels package providing a robust implementation of Ordinary Least Squares linear regression modeling. The source is available under MIT license on GitHub for exploration by the reader.

4.1 Input Data Characteristics

The data set must be comprised of n i.i.d variables. The response y is computed to follow a linear model with all main effects and two factor interactions in accordance with equation 1. In practice, most of the research focused on the two variable case shown in equation 2.

$$y = \beta_0 + \beta_{X_1} X_1 + \ldots + \beta_{X_n} X_n + \beta_{X_1 X_2} X_1 X_2 + \ldots + \beta_{X_{n-1} X_n} X_{n-1} X_n + \epsilon \tag{1}$$

$$y = \beta_0 + \beta_{X_1} X_1 + \beta_{X_2} X_2 + \beta_{X_1 X_2} X_1 X_2 + \epsilon \tag{2}$$

To simplify data generation, all β values are proportional to β_{X_1}. For example, in a data set with three variables, let $\beta_{X_1} = 10$, then the array $[0.1, 1.5]$ defines β_{X_2} and β_{X_3} to 1 and 15 respectively. The β values for interaction terms are set explicitly.

As the dataset is artificially generated, noise is added to the final computation of the response. A `noise_factor` variable is introduced to control the variance of the noise proportional to the largest β value of the main effects. The mean of the noise must be zero in accordance with the underlying assumptions of linear regression. These relationships are described in equations 3, 4 and 5.

$$\epsilon \sim N(0, \sigma) \tag{3}$$

$$\texttt{noise\_factor} \geq 0 \tag{4}$$

$$\sigma = MAX(\beta_{MainEffects}) * \texttt{noise\_factor} \tag{5}$$

4.2 Simulating Missing Data

For a given observation, the probability that a given variable is missing is idependent from the probability that any other variable is missing. This randomness assumption is refered to as Missing Completely at Random (MCAR). For the implementation used in this study, a boolean mask was randomly created for each of the variables. The selection of data to eliminate can be written as replacement. The DataFrame method, `pandas.DataFrame.sample` is used for performance and simplicity reasons. Removed values were set to NULL. The number of datapoints to remove is controlled as the fraction of data missing. Given MCAR, it is possible for more than one variable to be missing. These observations are removed from the dataset before imputation begins. The response y is never nulled.

4.3 Imputation Methods

Four methods of data imputations were explored in this experiment.

- Drop Imputation; removal of observations where any value is null.
- Mean Imputation; replace missing values of a variable with the mean of the available samples for the respective variable.
- Random Imputation; random replacement from the set of available samples for the the respective variable.
- Inversion Imputation; invert values using a model fit to the complete observations.

$$X_m = \begin{bmatrix} 1 & X_1 & X_2 & X_1 X_2 \end{bmatrix} \tag{6}$$

The complete observations are given as.

$$X_c = \begin{bmatrix} X_m \neq NULL \end{bmatrix} \tag{7}$$

Compute the beta estimates for the data using the remaining complete observations.

$$\hat{\beta}_c = (X_c^T X_c)^{-1} X_c^T y \tag{8}$$

4.4 Experimental Design

Variable	Levels	Description
`sample_size`	50, 100, 500	Samples in the generated data set
`noise_factor`	0.1, 0.3	Relative noise in the response
β_{X_2} Factor	0.1, 1	$\beta_{X_2} = \beta_{X_2} Factor * \beta_{X_1}$
$\beta_{X_1 X_2}$ Factor	0.000001, 10, 50	$\beta_{X_1 X_2} = \beta_{X_1 X_2} Factor * \beta_{X_1}$
Imputation Method	Drop, Mean, Inverse, Random	Imputation method used
Drop Column	X_1, X_2, X_1 and X_2	Variables whos values are destrpyed in a run

5 Results and Analysis

Inversion imputation forces the missing data onto the line defined by the remaining data. The imputed data set is now biased toward the model with which it was imputed. When a model is fit again to the whole data set, the result is a tighter CI around the β_1 estimate. This bias is magnified as the percentage of missing data increases. The model will also converge on zero error as the vast majority of data points will be exactly on the inverted model line. Whether or not this is the correct line, the model will converge on perfect fit as seen in the R^2 plots.