Use Case and Performance Analyses for Missing Data Imputation Methods in Big Data Analytics

Lan Yang, Jason Amaro Chiang
Computer Science Department
California State Polytechnic University
Pomona, CA 91768, United States
lyang@cpp.edu

ABSTRACT

In big data analytics the phenomenon of missing data is universal due to reasons such as faulty equipment and nonresponses in surveys. Imputation is the process of replacing missing data with substituted values. Proper imputation could greatly improve the accuracy and effectiveness of big data analytics.

In this paper, we analyze a rich set of deletion and imputation methods, focusing on strengths, weaknesses, best use cases, implementation strategies, and error-examination based performance analysis. Our goal is to find the best fitted imputation method(s) for each given use case.

CCS Concepts

• **Information systems → Data management systems → Information Integration → Data cleaning**

Keywords

Big data analytics; missing data; imputation; error estimation.

1. INTRODUCTION

In big data analytics the phenomenon of missing value is universal due to many reasons including but not limited to faulty sensors, nonresponses in some survey questionnaires, and human/code errors. Missing data could play a significant effect on the conclusions that are drawn from the data analysis. Methods for handling missing data are either deletion or imputation. Deletion removes selected entries containing missing data, while imputation is the process of replacing missing data with substituted values. We analyzed a diverse set of deletion and imputation methods. Our conclusions are: (1) deletion methods could only be used when missing data is insignificant; and (2) imputation methods are use-case sensitive and proper choice of imputation methods could greatly improve the accuracy and effectiveness of big data analysis. The outcome of our research could help identify best fitted imputation method for a given use case.

2. BACKGROUND

Missing data is a widespread problem in both the scientific community and industry. Often researchers must do statistical analysis on data to gain new insights into a problem or to find some pattern or trend. But missing data poses an issue when conducting such analyses. Before delving into the handling missing data, it is important to discuss the different reasons for why data goes missing.

In Statistics, missing data can be classified into three different categories: MCAR, MAR, and MNAR [1][2].

(1) Missing Completely at Random (MCAR)
There is no relationship between the missingness of the data and any values observed or missing. There is nothing systematic going on that makes some data more likely to be missing than others. All data points share the same probability of going missing. Common examples are forgetting to fill an entry in a survey, dropping a test tube and omitting the data collected on it.

(2) Missing at Random (MAR)
In this category, there is a systematic relationship between the propensity of missing values and the observed data but not the missing data itself. For example, members of a library who live farther to that library are possibly to have more overdue books.

(3) Missing Not at Random (MNAR)
There is a relationship between the propensity of a value to be missing and its hypothetical value. For example, people with high salaries are less likely to report income in a survey, people who have many overdue books are less likely to report how many overdue books they have because it is embarrassing.

There have been a lot of research works done on missing data imputation and most works are focused on the implementation of imputation methods [3][4]. However, we view the problem from a different angle – from the viewpoint of use cases. For example, an efficient imputation method may work well with filling up missing/invalid data in a movie rating survey, but may not work effectively for a store's sales data. It is very hard for ordinary users to understand, not to mention classify cases as MCAR, MAR, and MNAR. In this research, we perform a thorough analysis of various deletion and imputation methods as well as error bounds for commonly used imputation methods. The outcome of our analysis would serve as an advisory system for choosing best-fit imputation methods for different use cases.

3. ANALYSIS OF DELETION AND IMPUTATION METHODS

In general, there are two ways of handling missing data. We can delete it, or we can perform data imputation. In this section we provide a summary of a diverse set of commonly used deletion and imputation methods we implemented and/or validated. We

ICCDE '20, January 4–6, 2020, Sanya, China
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7673-0/20/01…$15.00
https://doi.org/10.1145/3379247.3379270

explain the strengths, weaknesses, best use cases, as well as a brief description of the implementation details.

3.1 Deletion Methods

We analyzed listwise and pairwise deletions. Deletion methods are simple and computationally inexpensive. However, they should only be applied to cases when missing data are insignificant.

3.1.1 Listwise Deletions

Listwise deletions remove a row/record of data or a column/variable. There are two common methods: (1) complete case analysis (CCA) which deletes a row/record if one of the fields/columns has missing value; and (2) variable deletion (DV) which removes a column/variable if more than X% of its data is missing. For example, assume you are conducting analyses of college student records including cumulative GPA, units taken for each semester, and grades in college algebra and so on. Participant X has missing data for units taken for the first semester, therefore, participant X will be completely removed from the analyses because the participant does not have complete data for all variables. Or, if a lot of participates, for example 40% of participates, miss grades in college algebra, thus, the college algebra column will be deleted as the substantial amount of missing data could invalidate the analysis results. In our implementation, for the CCA method we deleted each row if at least one of its values was missing. In the DV implementation columns are only dropped if more than 30% of its data is missing. Otherwise, we impute each missing value with zero – this step was added for convenience in processing data as dataframes we created from the datasets need to have the same shape when calculating the error.

The major strength of listwise deletions is that it will not introduce bias. We suggest that this method could be applied to missing completely at random (MCAR) cases where the deleted records/variables are insignificant. When there is a substantial amount of missing data, using this method could result in losing large amounts of sample data due to omission of records or variables. [5]

3.1.3 Pairwise Deletion

Using pairwise deletion will not omit a case completely from the analyses. Pairwise deletion only omits cases when analysis is done using the variable with missing data. As a result, analysis may be completed on subsets of the data depending on where values are missing. For the above example, Participant X will be omitted from any analyses using units taken, but will not be omitted from analyses for which he/she has the complete data such as analysis of cumulative GPA. This method adds flexibility to the examination of all cases (records/rows) in which the variables of interests are present. However, it cannot be applied to cases when missing not at random (MNAR) or missing at random (MAR). Pairwise deletion face challenges of drawing inferences from subset of samples, not the complete set of samples, thus in some cases the results may not be completely convincing. Same as listwise deletion, pairwise deletion should only be used when missing data is insignificant.

3.2 Imputation Methods

Imputation is the process of replacing missing data with substituted values. The main problems causing by deleting missing data are the possibility of introducing a substantial amount of bias, and the difficulties of guaranteeing the completeness of samples. Imputation is seen as a way to avoid such pitfalls. Also, imputation methods are proved to be effective in handling a time series data [3][4], i.e a sequence of data taken at successive equally spaced points in time. For example, heights of ocean tides, historical rain falls, and the daily close values of stock markets. Normally these are sequences of discrete-time data. Time series data may exhibit trends (increasing or decreasing) as well as seasonality, which is the presence of variations that occur at specific regular intervals less than a year, such as weekly or monthly. Here we analyze a set of most commonly used imputation methods.

3.2.1 Last Observation Carried Forward (LOCF) and Next Observation Carried Backward (NOCB)

In LOCF method missing values are replaced by the last observed non-missing value [6]. In NOCB method, missing values are replaced by the next non-missing values [6]. LOCF and NOCB are both computationally inexpensive and easy to implement. They act well if the values local to the entry containing the missing data are similar. However, the methods can introduce bias in analysis and do not perform well when the data has a visible trend or pattern. For example, if there are huge gaps in the series in which values are missing in succession of one another, the dataset's variability will decrease greatly. The best scenario for the use cases is when a time series data is MCAR (missing completely at random) without exhibiting a trend. However, an exception to this is that, if the data only has small gaps of contiguous missing values, then using these methods will not accrue much error.

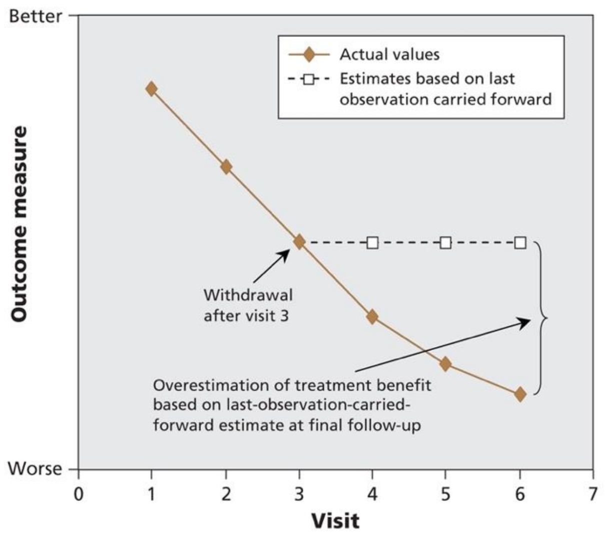

Figure 1: A visualization of the "gap" problem in LOCF. The same problem applies to the NOCB method. [6]

In our implementation, for the LOCF method each missing value is replaced with the previous non-missing value. Sometimes the very first entry was missing and there obviously isn't any value before the first record. To fix this issue we would simply impute it with the value next to it. We used similar strategy with NOCB. Each missing value is replaced with the next non-missing value and for cases were the last value was missing the previous non-missing value was imputed.

3.2.2 Imputation with Median, Mode, or Mean

These are another set of computationally inexpensive and easy to implement methods.

In imputation by median, for each column that has missing values the median of the column is taken and used to impute all the missing values of that column. This result of this method is not affected by outliers. However, the dataset may lose variability. In a time-series dataset with trend the accuracy of the imputed value drops as the record/row when the missing data is further away from the median. We recommend to use this method when the data behaves MCAR and the data records/rows are reasonably similar to one another. It may also work properly with small datasets even if their records/rows are dissimilar. However, do avoid using it for time-series dataset with trend where the accuracy of the imputed value drops significantly as the record/row with missing data is further away from the median.

For imputation by mode, the mode of each column is calculated and used to impute the missing values in that column. It has the same advantages as that of imputation by median. It is an effective method for discrete and categorical MCAR datasets. However, it is a bad method to use if the dataset is continuous.

Imputation by mean takes the mean of each column as the substitute value to impute the missing values in that column. This method incorporates data from the entire dataset to impute values regardless of how far they are from the imputed values (the mean), unlike the median imputation which loses its "predictive" power the further the record is from the imputed value. However, this method is susceptible to outliers. We recommend to use this method for non-time-series, MCAR dataset composed only of continuous variables, and avoid cases where the dataset has many large fluctuations or volatile as is the case with time-series data without trend and seasonality.

3.2.3 Linear Interpolation

Linear interpolation method [11] creates a line using two points. One point is the previous non-missing value and the other is the next non-missing value. Using these two points a line is created wherein the missing value will be somewhere in between these two points (i.e. somewhere in the line). The value is then imputed by taking its "x" position, inputting it into the line equation, and getting the "y", i.e. imputed value. For example, if you have two known points (x_1, y_1) and (x_2,y_2) such that the missing value of some point x is between x_1 and x_2, we can use the following equation to interpolate that value.

$$y = y_1 + \frac{y^2 - y^1}{x_2 - x_1}(x - x_1)$$

We implemented this method with the following findings. This method imputes with good accuracy for the missing values of time-series datasets with trend. Unlike median imputation its accuracy does not wane and unlike mean imputation it is less susceptible to a decrease in variability. It also handles the "large gaps in the data" problem that LOCF and NOCB methods suffer. However, this method does not incorporate the relationship between the different variables to impute values. This limits its applications to time-series datasets with trend only. Neither does it work well with time-series dataset with seasonality or without trend. We suggest to use this method for time-series datasets with a visible trend but no seasonality.

To cope with seasonality, an improved method is Linear Interpolation with Seasonal Adjustment [11] which removes the seasonal components of a time-series dataset to reveal the trend of the time-series dataset. Thus, this method could be applied to time-series datasets with a visible trend and with or without seasonality.

3.2.4 Linear Regression

Linear regression [12] models the relationship between dependent variables and the independent variables. A dependent variable changes value as the result of the changes to the independent variable. For example, how tall you are at different ages. Here age is an independent variable while height is a dependent variable, i.e. dependent on age.

This method attempts to predict the value of the independent variable (the variable with missing data) using the dependent variable by creating a training and testing dataset. The strength of this method is that it imputes the missing values by using the information from all available data as well as the relationship between variables. Therefore it can be applied to various types of datasets including MCAR and MAR datasets, broader than linear interpolation. However, it is more computationally expensive than any of the above discussed methods and more difficult to implement. Also, it may introduce bias into dataset. Traditional implementation of this method generally does not work well for small datasets or datasets with large amount of missing data.

We implemented this method using a variant of linear regression called Bayesian ridge regression [11] to better handle the bias that this method introduces. The implementation only handles cases where data variables have a linear relationship as applying generalization and regularization varies by datasets. Because having a lot of missing values in a dataset affects the accuracy of this model, missing values were first filled using mean imputation. In this sense our implementation of linear regression is slightly unusual. If we followed the traditional method and we had a very large amount of missing data, we would essentially be doing linear regression with little data that it would be very difficult to learn the relationship between features. Thus, we decided to impute the missing values with the arithmetic mean before proceeding to do the regression. Theoretically the reduction in variability caused by mean imputation and that of linear regression should generate poor results but in practice (at least according to our own experiments – see Section 4 Error Analysis) the model seems to perform well. We conclude that the linear regression method works with either time-series or continuous large datasets without a substantial amount of missing data.

3.2.5 KNN

The KNN method (k-Nearest Neighbors, a supervised machine learning method) identifies the "K" nearest neighbors to the record/row based on some distance function.[9] For continuous variables this function can be Euclidean, Manhattan, or Cosine distance functions. For categorical/discrete data variable this function can be the Hamming distance function. [9] We implemented this method and validated that this method can be used on either continuous or categorical/discrete datasets for both MAR and MCAR missing cases. However, it is computationally expensive and suffers from the "curse of dimensionality" (i.e. the accuracy drops with high dimensional data). Also, it introduces bias into the dataset. Thus, we advise not to use this method for high dimensional datasets.

3.2.6 MICE

The MICE (Multivariate Imputation by Chained Equations) method imputes all missing values using one of the other imputation methods above, usually mean imputation, and for each variable containing missing values we reset those previously imputed values to zero and perform a Bayesian ridge regression using the other variables. We do the same for each variable. The output will be some "m" number of datasets. [7][8][9]

We implemented this method using a Sci-kit learns' IterativeImputer module. According to Sci-kit learns website[13], this module is still in the experimental stage. Theoretically, this method should produce the best results since it avoids the bias introduced by the other imputation method, but in practice we found that it underperformed (note: we only tested on time series datasets). This method not only computationally intensive – the most computationally expensive one compared to all above methods, but also the implementation requires most sophisticated skill. However, this is the most comprehensive method covering a broader range of use cases. We are working towards implementing an improved version of this method.

4. ERROR ANALYSIS

To evaluate the methods we implemented, we analyzed the error bound caused by imputations using the following test.

4.1 Original Data Files

Data files: GOOG.csv and MSFT.csv, both are stock time-series data, but they differ in the number of records they contain. The MSFT.csv file contains 21 records and the GOOG.csv contains 1259 records. They contain columns "Date", "Open", "High", "Low", "Close", "Adj Close", and "Volume". For testing purpose, original test files do not contain any missing data as we need a way to measure the error produced by imputing the data. Missing data will be randomly introduced to the files during the testing.

4.2 Program Design

The program was implemented in Python using libraries including numpy, pandas, sklearn, datetime, and fancyimpute.

The program first reads in one of the original test files and then checks to see if the dataset is a time-series dataset or a generic non-time series dataset. It then casts all numeric columns to use the float data type for convenience purpose. After that, it will create a duplicate of our dataset with NaN (not-a-number, a Python representation of missing data) values randomly inserted (although uniformly distributed across each column). We replace 30 percent of our dataset with NaN values. To reproduce the same data just set the seed parameter of the insert_NaN method to 1323. This duplicate will also be stored in a file. After the program has successfully completed execution it will display the name of imputation methods sorted from best to worst and the error produced by using the corresponding imputation method.

Each of our methods will take this modified dataset with NaN values and impute them accordingly. Our program evaluates the error using root means square error (RMSE) [10]. In the case of MICE we compute the RMSE of each imputed dataset and take the average of the RMSEs.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

4.3 Results and Analysis

Table 1 and Table 2 show the RMSE our implementation of imputation metjhods as well as testing for one small and one large files.

Table 1 Root Mean Square Error for MSFT.csv file.

Imputation Method	Total Error
LOCF	1708980.6431008733
Median	1789105.940801078
Mean	1867736.4291548103
Linear Regression	1867736.4291594278
Linear Interpolation	2026215.113838474
MICE	2295778.1464756355
KNN	2744828.2395017953
NOCB	3161465.8894100194
MODE	4621238.6085569095
DV	5690425.8728139335
CCA	9568172.94113178

Table 2: Root Mean Square Error for GOOG.csv file

Imputation Method	Total Error
Linear Interpolation	183432.21266488076
NOCB	192096.96715111565
Linear Regression	221982.96261052156
LOCF	224330.22491739472
Mean	224646.08842357513
KNN	227610.2545243355
Median	233082.83853735152
Mode	251699.36066708792
MICE	254374.4580689835
CCA	756089.4625512282
DV	794843.3912876638

One expected result is that the deletion methods turn out to be the biggest underperformers compared to the other methods. Each time we delete a row or a column as is the case with complex case analysis or when deleting variables, we gain a large amount of error. Another expected result (at least for our examples and with time-series data in general) is that mode imputation will produce error greater than most of the other methods of imputation because we are dealing with continuous data. For the MSFT data we see that median imputation does quite well but when we switch to the much larger GOOG dataset its' performance drops down drastically. We speculate that this is due to the following two reasons: (1) when working with a subset of a time-series dataset the records will usually have similar values and so the median will also be similar to those values; (2) when working with a subset of the time-series dataset the median does not incorporate the fluctuation and patterns of the rest of the dataset and thus minimizes the error incurred. We believe that this also applies to mean imputation. Though the reason that mean imputation does better than median imputation for the GOOG dataset is that it takes the average of all the records for that specific feature and in a way accommodates itself to cater to the missing values of the entire dataset rather than a subset of the data which the median imputed value will usually fall under. Surprisingly, KNN imputation seems to underperform while imputation methods like LOCF and NOCB tended to perform better than KNN for both the smaller and larger datasets. We think that the reasons for KNN's underperformance is due to the dimensionality of the data. After using different seed values, we noticed that for GOOG dataset the results were more accurately demonstrated as shown in Table 3.

Table 3: Improved and more accurate results for GOOG.csv file

Imputation Method	Total Error
Linear Interpolation	151631.1872534534
NOCB	172843.53162783672
LOCF	196786.4529484747
Linear Regression	199425.45562761303
Mean	201404.2331007026
Median	206843.5187788882
KNN	214469.74364670532
MODE	216089.6714247699
MICE	244715.83870469176
DV	435621.19727965293
CCA	741889.0080532677

As seen from the above results as ordered from best to worst the winners tended to be linear interpolation, followed by either NOCB/LOCF first and linear regression second or vice versa, followed by the imputation methods measuring central tendency (mean, median, mode) and KNN, followed by MICE, and finally the deletion methods. We believe that the reasons NOCB and LOCF performed so well is due to the fact that the imputed values using these methods were very similar to their neighbors as the actual values would be. Linear interpolation and linear regression tended to be top performers, if not the best methods, for imputing missing data. Linear interpolation works well because it can bone in on the point containing the missing value by examining the two closest points before and after it while at the same time ignoring any irrelevant data which could skew the accuracy of the imputed value. Linear regression takes an almost opposite approach and incorporates all the data in the dataset but does so in a way different from mean imputation. It finds the relationship of the different features/variables and then imputes the missing value by looking at the values of the other variables and imputing a value based on the known relationship.

Perhaps the most surprising result was how bad MICE performed. For smaller datasets like the MSFT dataset it performed well but not stellar. For larger datasets like the GOOG dataset it was only able to best the error for the deletion methods.

5. CONCLUSION AND FUTURE WORK

In this paper we researched and implemented a rich set of deletion and imputation methods for handling missing data. We suggested best use cases for each method based on the method's behaviors and characteristics. We analyzed each method using root means squared error. There are many other methods of imputation that we can implement including but not limited to: rolling averages, maximum likelihood estimation, linear interpolation with seasonal adjustment, and different implementations of MICE using varied estimators. We are also in the process of implementing an intelligent advisor system that, by inputting feature descriptions of a use case the system would provide one or more recommended imputation methods as well as some methods that definitely should be avoided for this use case.

6. ACKNOWLEDGMENTS

Our thanks to a Discovery Through Computer Science Award for partial support of this research.

7. REFERENCES

[1]. van Buuren, S., Flexible Imputation of Missing Data, 2nd edition, CRC Press.

[2]. Swalin, A., How to Handle Missing Data, retrieved on August 15, 2019 from https://towardsdatascience.com/how-to-handle-missing-data-8646b18db0d4

[3]. Asadi, R., Regan, A. A convolutional recurrent autoencoder for spatio-temporal missing data imputation, Proceedings of 2019 International Conference on Artificial Intelligence (ICAI'19), pp. 206-212, ISBN: 1-60132-501-0, CSREA Press ©

[4]. L. Li, J. Zhang, Y. Wang, and B. Ran, "Missing value imputation for traffic-related time series data based on a multi-view learning method," IEEE Transactions on Intelligent Transportation Systems, 2018.

[5]. Statistics solution, Handling Missing Data: Listwise Versus Pairwise Deletion, retrieved August 15, 2019 from https://www.statisticssolutions.com/handling-missing-data-listwise-versus-pairwise-deletion/

[6]. Molnar, F. J., Hulton, B., Fergusson, D., Does analysis using "last observation carried forward" introduce bias in dementia research?, retrieved on August 15, 2019 from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2553855/

[7]. Little, R., A Tet of Missing Completely at Random for Multivariable Data with Missing Value, Journal of the American Statistical Association Vol. 83, No. 404 (Dec., 1988), pp. 1198-1202

[8]. Li, C. Little's Test of Missing Completely at Random. The Stat Journal (2013), 13, No. 4, pp.795-809

[9]. Pedro J. GARC ́IA-LAENCINA, Jos ́e-Luis SANCHO-G´OMEZ, An´ıbal R. FIGUEIRAS-VIDAL, Machine Learning Techniques for Solving Classification Problems with Missing Input Data, retrieved August 15, 2019 from https://pdfs.semanticscholar.org/7b1e/b4a482bf903079e5775b19e88225f956b9f2.pdf

[10]. Statistics How To, RMSE: Root Mean Square Error, retrieved on August 15, 2019 from https://www.statisticshowto.datasciencecentral.com/rmse/

[11]. Lepot, M., Aubin, J., Clemens, F., Interpolation in Time Series: An Introductive Overview of Existing Methods, Their Performance Criteria and Uncertainty Assessment, retrieved on August 15, 2019 from https://www.mdpi.com/2073-4441/9/10/796/pdf

[12]. Bingham, N. H., Fry, John M., Regression – Linear Models in Statistics, Springer, SBN 978-1-84882-969-5

[13]. https://scikit-learn.org/stable/modules/impute.html