2021

MIDAS

MULTIPLE IMPUTATION WITH DENOISING AUTOENCODERS

DATA7703 GROUP X

JOSEPH MENESCH

MATTHEW REA

ROBERT WILLIAMSON

JULIAN STANNARD

We give consent for this to be used as a teaching resource.

Abstract

Analysing datasets through statistics and machine learning has proven to provide great insights to improve our world. With many people around the world realising and harnessing the power of data, being able to draw accurate and meaningful insights is of vital importance. Any form of data analysis is greatly dependent on the quality of data being used and unfortunately many datasets suffer from incorrect or missing entries. Without properly dealing with this problem, incorrect or misleading conclusions can be made with potentially dire consequences. Imputation methods aim to fill in the blanks by replacing missing or incorrect data with a predicted value that aims to reflect the true dataset. This analysis compared different imputation techniques across two classification and two regression datasets of varying size and dimensionality. Simpler techniques, such as mean imputation, were compared against more advanced techniques, such as Multiple Imputation using Denoising Autoencoders (MIDAS). The accuracy of MIDAS was found to be overall the greatest of all the imputation methods, even outperforming the advanced method of Multivariate Imputation by Chained Equations (MICE). MIDAS was found to be significantly faster than MICE, but much slower than the simpler imputation methods.

Contents

Introduction

Over the last few decades, computational power has exploded, enabling tasks that were once never possible. Moore's Law states that the number of transistors in an integrated circuit roughly doubles every two years, with the law holding true throughout the 20th century [1]. As a result, applications such as machine learning are now possible on an extremely accessible scale, where anyone with a computer can create and implement these algorithms. Whilst machine learning is a useful and fascinating field, the effectiveness of every machine learning model depends greatly on the quality of the data that it is used on. Researchers found that datasets used in educational and psychological research commonly contained roughly 15-20% of missing data, which can lead to incorrect or exaggerated conclusions made by the studies [2]. It is clear that being able to handle the missing data, such as via imputation, or predicting missing values, is important so that machine learning models can be enabled to perform at their best.

Missing data is an important issue that often needs resolution for datasets to be of any use. There has been extensive research into a range of methods of dealing with missing data and their limitations. One basic method of addressing missing data is to ignore the entries that contain any missing attributes. This method unfortunately reduces overall dataset size, which some datasets can't afford, and can reduce model accuracy, which more importantly can introduce model bias. Data missingness is commonly attributed to one of three categories: Missing Completely at Random (MCAR), Missing at Random (MAR) or Missing Not at Random (MNAR). As such, the choice to simply exclude data with missing values is not recommended and data replacement is preferred.

As concluded by Gondara and Wang [3], missing data replacement techniques falls in two camps; one method is to use the non-missing data to attempt to model the missing data process and estimate the parameters of a model, where the other method seeks to impute the missing data with plausible values. There are various statistical methods by which to impute missing values using the values of the non-missing data, however this can lead to increased bias in the model, especially if using a median or mean value of the attribute.

Gondara and Wang [3] propose that Multiple Imputation using Denoising Autoencoders (MIDAS) can be used with great effect if you simply consider missing data as data corruption. Gondara and Wang outline their architecture of an overcomplete, multi-layer neural network of denoising autoencoders to reconstruct complex, high dimensional and extensive datasets with improved performance over traditional Multivariate Imputation by Chained Equations (MICE) techniques. Our project aims to assess multiple imputation with the use of denoising autoencoders as a more robust and scalable solution to datasets of increased size and complexity.

Background

As humanity moves further online, the ability to gather and compile larger and larger datasets has grown exponentially. Even as our datasets grow in scale and complexity, they don't necessarily grow in completeness. For this reason, there is a need for more robust and efficient methods of data imputation that can scale as required. Lall and Robinson [4] seek to prove MIDAS is such a technique, investigating its prowess in imputation in large and complex datasets. The approach is to treat missing data as an additional part of corrupted data and then draw imputations from a model trained to minimize reconstruction error on the data that is present.

Multiple Imputation

A number of developments into imputation models using neural networks has improved complexity and accuracy of value estimation, however any single imputation method risks introducing data bias by means of model bias. If the model used to impute the single value is not accurately tuned – under or overfitted on training data – then the single imputation values risk introducing this bias. For this reason, multiple imputation methods are generally preferred as this risk is greatly reduced. Multiple imputation seeks to impute a range of replacement values using several models, then pool these imputed values to find consensus.

At its heart, MIDAS leverages the strengths of the fundamentals of multiple imputation in its operation. As outlined by Lall and Robinson, the process of multiple imputation consists of three steps. The first is to replace the missing attributes in the dataset with *M* independently imputed replacement values that fit the model relationships observed in the attributes. These *M* – now complete – datasets can be analysed independently and dataset metrics estimated. The third process now combines these *M* sets of estimated metrics using a set of simple rules to "leverage variation across these datasets to reflect our uncertainty about the correct imputation model" [4].

Denoising Autoencoders

An autoencoder is classified as a type of artificial unsupervised neural network and seeks to automatically learn features or dimensional reduction methods to be able to accurately reproduce the input data. As such, the autoencoder's basic architecture consists of an encoder and a decoder. The encoder maps the input into a hidden layer while the decoder reconstructs the values from this latent space. A variation on this network is a denoising autoencoder (DAE), which takes data corrupted by some noise and then undoes the corruption to reconstruct a value without the noise [5]. This forces the autoencoder to learn robust features in its hidden layers used for reconstruction.

Denoising autoencoders have been well researched as a method of dimensionality reduction. Pascal Vincent and his team at the University of Montreal in 2008 [6] further researched the use of DAEs as a method of making an unsupervised neural network learn representations robust to partial distortion of the input data. They hypothesised that a DAE is expected to capture a stable structure of dependencies and characteristics of the observed inputs such that outputs should be recoverable from partial observations only. Vincent outlines a method of building a deep neural network that first corrupts the initial input vector (x) into a partially destroyed vector ($\tilde{x}$). It then maps this corrupted input vector ($\tilde{x}$) to a hidden representation (y) through a deterministic mapping, parameterized by a weight matrix and a bias vector. The resulting latent representation is then mapped back to reconstructed representation (z). Stochastic gradient descent is then used to

optimise the weight and bias vectors to minimise a reconstruction cross-entropy loss equation (L_H). This is depicted below in Figure 1, originally outlined by Pascal Vincent, 2008.

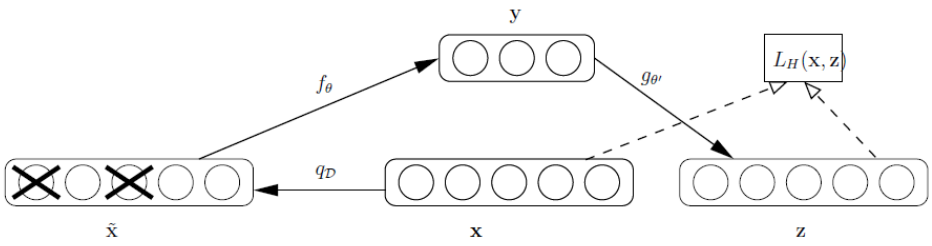

Figure 1: Denoising autoencoder [6]

MIDAS Algorithm

MIDAS modifies the traditional denoising autoencoders model in two key areas. Firstly, other than choosing random subsets of inputs to work with, it also substitutes all missing attribute values as '0'. MIDAS then predicts the replacements to the corrupted values that where both originally missing and originally observed by minimising the loss function, but the loss function only measures the reconstruction error on the originally observed values [5]. MIDAS uses a neural network to characterize the data by optimising several nonlinear functions. MIDAS is able to capture both simple and highly complex characterisations of the input data by systematically introducing additional missingness during the training steps and drawing imputations from the final trained networks.

Lall and Robinson in their work with combining denoising autoencoders into a multiple imputation methodology, outlines their implementation of a "dropout" technique that seeks to avoid the risk of overfitting in the training phase. This regularization technique randomly drops a number of neurons and their connections from the neural network in each iteration during training [5]. In effect this is equivalent to training different neural networks in an ensemble method. Lall and Robinson propose this use of dropout "extends the corruption process deeper into the neural network architecture" [4] and leads to MIDAS having the flexibility to work with complex datasets while matching performance of more traditional multiple imputation methods on simple data structures. The MIDAS algorithm process is outlined in the schematic in Figure 2 originally outlined in Lall and Robinson [7].

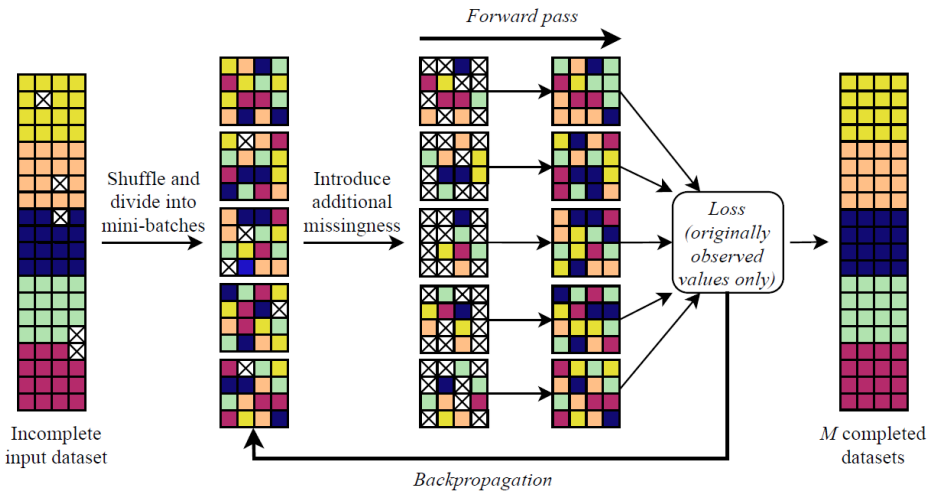

Figure 2: Schematic of MIDAS Training Steps [7]

MIDASpy [8] is a Python package available on github.com that provides an implementation of MIDAS that is in-line with the procedure outlined by Lall and Robinson [4]. This is the variation of the MIDAS technique we seek to review and assess on a range of datasets and missingness. We focus on the performance of MIDAS in comparison to MICE and other benchmark imputation methods.

Methodology

We present a framework for testing various imputation methods to improve the performance of different machine learning algorithms against diverse datasets. Each method is benchmarked against performance on the complete dataset. We cover both classification and regression tasks in our assessment.

Figure 3 provides a summary of the assessment framework including datasets, imputation methods and machine learning algorithms for each learning task.

Task	Dataset[1]	Imputation scenarios				Evaluation • cross-validation • accuracy for classification • R2 for regression		
		Complete dataset	Simple • Mean • Mode • Median	MICE[2] (max iterations)	MIDAS (hidden layers)			
Classification	**Cover type (10% of rows)** • 54 features (44 binary) • 58,101 rows • 7 classes	**Standard scaling** • ex. binary features	**Corrupt dataset** • 20% per feature	100	256 256 256	**KNN classifier** • k = 5	**Random forest classifier** • n_est = 100	**SVM classifier** • kernel = 'rbf' • degree = 3
	Wine • 13 features • 178 rows • 3 classes			10	256 256			
Regression	**California housing** • 8 features • 20,640 rows			40	256 256	**Linear regression**	**Random forest regressor** • n_est = 100	**SVM regressor** • kernel = 'rbf' • degree = 3
	Diabetes • 10 features (1 binary) • 442 rows			100	256 256			

Figure 3: Imputation assessment framework

Datasets

The datasets used for this assessment aim to test each imputation method's performance on simple and more complex data. As well as understanding how each method scales in both dimensionality (features) and the number of samples (rows).

Four datasets were chosen for the assessment. Two for the task for classification and two for regression. All datasets used were taken from the Scikit-Learn datasets module to enable easy reproduction of the code.

Cover type (reduced) – was chosen as a complex classification dataset and tests both dimension and row scalability, with 54 features and 58,101 rows and 7 target classes. The data was reduced by randomly sampling 10% of the original rows due to limitations in computing resources to train both multiple imputation methods.

Wine – provided a small simple classification dataset with 13 features, 178 rows and 3 target classes.

California housing – was used to test scalability in regression with 20,640 rows and 8 features.

Diabetes – provided a small dataset for regression with 10 features and 442 rows.

Imputation methods

The range of imputation methods included in our assessment were:

- Simple methods: mean, median and mode imputation
- Multiple imputation methods: MICE and MIDAS

Simply removing the corrupted rows was not included as part of our assessment.

Multiple imputation tuning

For MICE a trial-and-error approach was applied to improved convergence. This is controlled via the max_iter hyper-parameter, which allows the algorithm to continue training up to an upper limit until the its loss function converges to within a tolerance of 0.001. Figure 3 shows the final values applied for max_iter for each dataset.

MIDAS was tuned in its structure. For most datasets the default structure of two hidden layers with 256 nodes was sufficient. For very simple datasets, this structure avoids overfitting with the application of dropout, which randomly drops nodes and connections in the structure during each mini-batch iteration in the training procedure. It is also computationally efficient with the default structure, therefore there was no incentive to simplify the structure on smaller datasets.

For the Cover type dataset, the training loss struggled to converge, so an additional hidden layer was added to improve performance.

Machine learning algorithms

For the classification the selected algorithms were K-nearest neighbours (KNN), random forest classifier and support vector machines (SVM) classifier. For regression the selected algorithms were linear regression, random forest regressor and SVM regressor. All algorithms were applied with the default hyper-parameters from Scikit-Learn as shown in Figure 3.

Evaluation metrics

For classification and regression accuracy score and R-squared were selected. These were chosen for three key reasons:

1. Both are scale invariant, so performance is easily compared across different datasets
2. Both increase with improved performance
3. Both can be expressed on the same continuous scale from 0 to 1

Procedure

The following procedure was undertaken for each dataset:

1. **Standard scaling** was applied to all non-binary features by subtracting the mean and dividing by the variance over all samples.
2. **Corruption** was applied by randomly sampling 20% of samples per feature and replacing the values with ‘None’. 20% was chosen to align with Enders as the typical proportion of missing data in research datasets.
3. **Imputation** was applied using the methods described above.
4. **Learning and evaluation** was repeated for each model (appropriate for classification or regression) and for each of the complete and imputed versions of the data. The true performance was then estimated using cross-validation with the evaluation metrics described above.

Analysis & Results

Results are presented for both the task performance and computational performance of each imputation method.

Task performance

Figure 4 show the results of the assessment for each dataset. For the classification task the complete dataset outperformed all imputation methods as expected. With some datasets showing a large difference in accuracy between the original dataset and the imputed datasets.

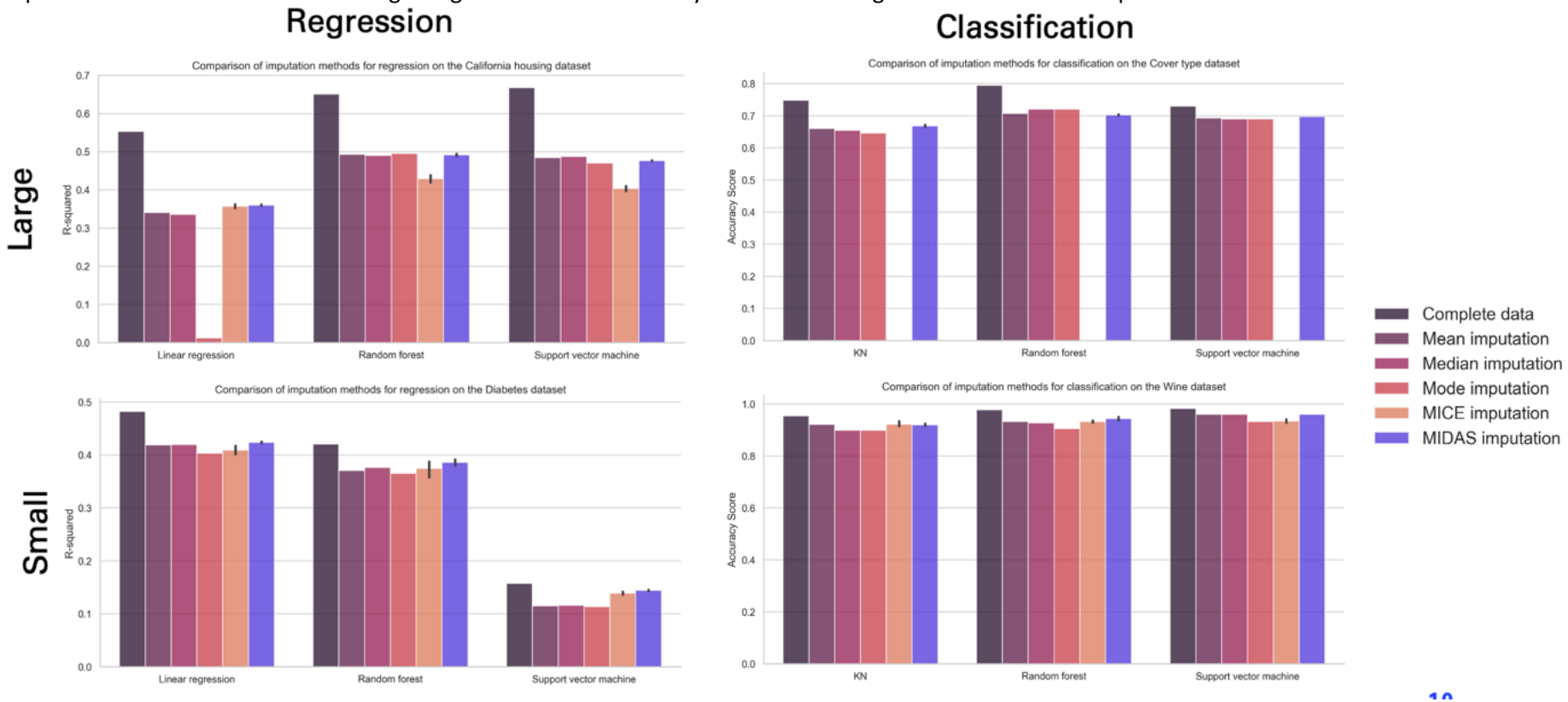

Figure 4: Machine learning model accuracy across datasets

On the whole, the imputation methods all performed roughly the same across all datasets. The simple imputation methods, being mean, median, and mode imputation, all performed roughly the same for each machine learning model used on each imputed dataset. MICE and MIDAS appeared to have a slightly better impact on machine learning model accuracy overall, although the difference between them and the simpler techniques was not very substantial. MICE appeared to perform worse than MIDAS across all datasets and even failed to converge for the large Cover type dataset. MIDAS appeared to provide the best performance out of all imputation methods.

Computational performance

As can be seen in Figure 5, the three simple imputation methods all showed very good performance that was roughly comparable to each other across all datasets. By comparison, the more advanced MICE and MIDAS methods took a much longer time to impute the data. For example, the median imputation method took only 0.016 seconds to impute the California housing dataset, whereas MIDAS took over 117 seconds and MICE 362 seconds. MIDAS was seen to outperform the MICE algorithm on all datasets, showing drastically reduced runtimes. It should be noted that no runtime was recorded for MICE on the Cover type dataset as it failed to converge. A table of runtime results can be found under Appendix A.

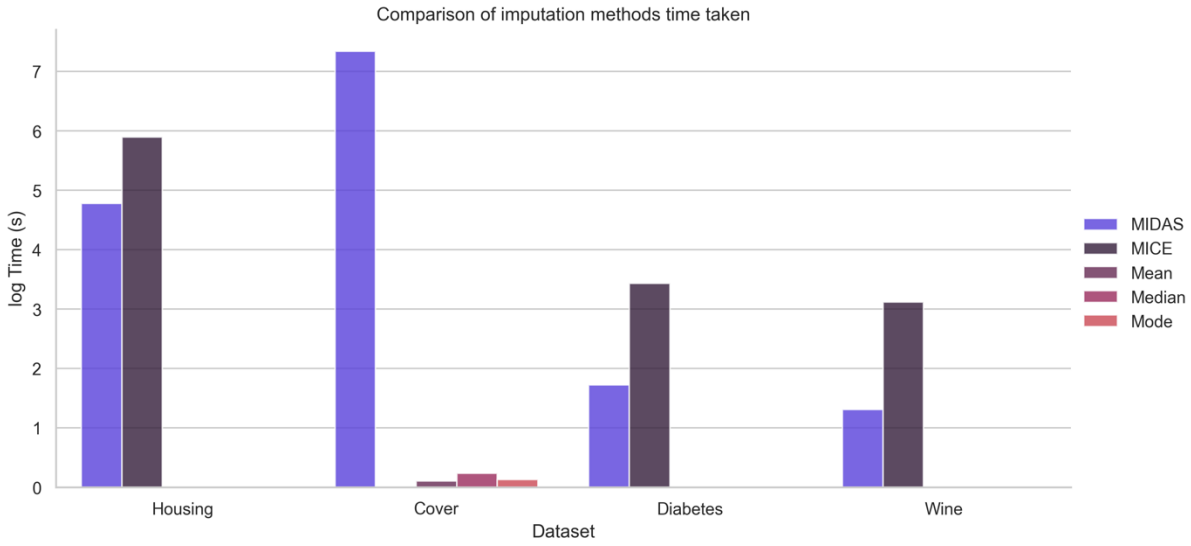

Figure 5: Runtime of imputation techniques

Overall, MIDAS was found to be the best performing imputation algorithm from the three simple techniques and one more advanced technique tested. It was found to be significantly faster than MICE, but severely slower than the simpler methods. The accuracy found from all imputation methods were all roughly the same and for the most part, noticeably worse than the complete dataset, with MICE even failing on the high-dimensional Cover type dataset.

Conclusion

As we have seen, modern day computational power has enabled the storage and use of big data. Analysing data and using machine learning models to make data-driven decisions is now widely accessible and available to anyone with a computer. Whilst there is great potential that can be unlocked through data analysis and machine learning, the success of these methods is heavily dependent on the quality of the data used. Many datasets contain some level of missing or incorrect data and not all datasets can simply remove any rows that have a missing value, nor should one always do this anyway. One method of dealing with these situations is by using imputation techniques to replace the missing or incorrect data with suitable values.

Many imputation methods exist, from simple algorithms like mean imputation to more advanced ones like MICE. MIDAS was developed by Lall and Robinson and aimed to use denoising autoencoders and multiple imputation to more accurately impute datasets, by harnessing the power of deep learning architectures to detect deep trends within the data. The analysis implemented and compared MIDAS against other imputation algorithms on two regression and two classification datasets of varying size and dimensionality.

The results showed that MIDAS was at least as good as the simpler imputation algorithms on all datasets and even outperformed them on others. MICE by comparison was overall outperformed by MIDAS and did show greater performance across some datasets than the simpler algorithms, but failed to converge on the more complex “Cover type” dataset. The difference in performance between MIDAS and the other techniques was not very large, but the computation time of MIDAS was far greater than the simpler algorithms. MICE on the other hand had a significantly greater computation time than MIDAS and showed weaker performance.

MIDAS appeared to be the superior algorithm when compared to MICE, but its performance when compared to the simpler imputation methods was not very substantial and arguably not worth the difference in computation time, at least on the datasets and conditions that it was tested with.

Limitations and Future Recommendations

Due to time constraints with the project, the machine learning models used on the complete and imputed datasets were fitted using default parameters. In some cases, this led to poor performance from the estimators, as was seen especially using the support vector machine on the Diabetes dataset. Taking some time to better train the models on the complete dataset could reveal larger differences between the imputation algorithms.

Further to this, more imputation methods could be used, such as the method of simply removing any rows that contain a missing value in any of the features. Additional imputation techniques could also be tested and compared to see if there are any “middle-ground” algorithms that provide better performance accuracy than the simpler algorithms, but don’t take as long to compute as the more advanced methods of MICE and MIDAS.

Lastly, using a wider range of datasets could help determine particular circumstances where each imputation method is best suited. Perhaps more complicated datasets are required for MIDAS to show its strengths over simpler methods such as mean imputation.

References

[1]. Moores Law No More, PWG, Available here: https://www.pwc.com.au/digitalpulse/moores-law-no-more.html

[2]. Enders CK, Psychol Methods. 2003 Sep; 8(3):322-37

[3]. Lovedeep Gondara and Ke Wang, MIDA: Multiple Imputation using Denoising Autoencoders, *Department of Computing Science Simon Fraser University,* 2018

[4]. Ranjit Lall and Thomas Robinson, Applying the MIDAS Touch: How to Handle Missing Values in Large and Complex Data, 2020

[5]. Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning, *MIT Press,* p499-523, 2016, Available here: https://www.deeplearningbook.org/contents/autoencoders.html

[6]. Pascal Vincent, Hugo Larochelle, Yoshua Bengio, Pierre-Antoine Manzagol, Extracting and Composing Robust Features with Denoising Autoencoders

[7]. Ranjit Lall and Thomas Robinson, Applying the MIDAS Touch: Accurate and Scalable Missing-Data Imputation with Deep Learning, 2020

[8]. MIDASpy Python Library, Available here: https://github.com/MIDASverse/MIDASpy

Appendix

Appendix A

Runtime measured (seconds) for each dataset and each imputation method.

	California Housing	Cover type	Diabetes	Wine
Mean	0.000	0.109	0.000	0.000
Median	0.016	0.266	0.000	0.016
Mode	0.016	0.141	0.000	0.000
MICE	362.422	-	29.905	21.618
MIDAS	117.828	1539.297	4.609	2.703