Vrije Universiteit Amsterdam

Bachelor Thesis

Data imputation with deep learning and a comparison with statistical techniques

Author: Minh Hai Nguyen (2613201)

1st supervisor: *Michael Cochez*
daily supervisor: *Michael Cochez*
2nd reader: *Peter Bloem*

A thesis submitted in fulfillment of the requirements for the VU Bachelor of Science degree in Computer Science

July 2020

Contents

Abstract

Data is widely accessible nowadays and if correctly utilized, they can be powerful tools for providing meaningful insights into any field of research. However, despite their availability, datasets are often incomplete. As a response, scientists are required to deal with these situations with various methods such as list-wise deletion and data imputation. However, the cost of list-wise deletion is too expensive and data imputation methods such as mean imputation are ineffective and generally discouraged. The purpose of this study is to perform data imputation using deep learning on datasets with both linear and non-linear correlations and compare their performance to statistical techniques. Since achieving this requires the features to be correlated, this study first evaluates deep learning's capability to identify feature correlations and reconstruct them. The results showed that data imputation can be achieved while also preserving the feature correlations, as the distance correlation between the imputed values and the original values did not show any significant difference.

1 Introduction

1.1 Data Imputation

Missing data is a recurrent problem found in all sciences. Data collected is not always guaranteed to be complete whether it's due to random or systematic missingness. For instance, in social science, this can be due to respondents' refusal to answer for personal reasons [1]. The process behind the missingness is usually not known and ignoring the reasons behind this can cause implications when dealing with real data [2]. The missing values in a dataset can limit the scope of analysis for any data scientist and this issue has been persistent even in recent years. To avoid inconsistencies in the data, there are various traditional techniques used in practice to deal with the missing values. Examples of these are list-wise deletion and single imputation methods [3]. These methods were, however, criticized and "generally not recommended" as "they are amongst the worst methods available for practical applications" [4]. The former technique refers to the act of excluding cases from the analysis if the entry contains a missing value [5]. Although this yields a complete data, the costs that come along usually means that other techniques are more favorable. Not only does this reduce the sample size available, but it can also eliminate crucial data if the pattern of missingness is related to the variables. For example, minor ethnic groups may prefer to not disclose their nationalities in surveys. This would lead to a systematic pattern in missingness which could result in the absence of crucial insights. For imputation methods, an example would be mean imputation. This refers to replacing missing values with the arithmetic mean of the available data and treating them as if they were real [3]. Although this does ensure that the data sample size is maintained, the replacement values may not represent the data accurately. For instance, the correlation may not be well preserved and can yield a correlation coefficient difference of up to 0.34 [3]. This paper aims to challenge the traditional techniques by comparing the performance of deep learning and the traditional methods for data imputation.

1.2 Feature Relatedness

Correlation between the columns are essential for data imputation, and traditional statistical techniques such as Pearson's R-value does a great job in calculating whether two variables are dependent. Unfortunately, this is only valid for linear correlations, because Pearson's coefficient is only a measure of a monotonic association. Monotonic association can be defined as a "relationship between 2 variables is a one in which either (1) as the value of 1 variable increases, so does the value of the other variable; or (2) as the value of 1 variable increases, the other variable value decreases" [6]. In this paper, the aim is to provide an insight into deep learning's capability to detect correlations and aim to do this even beyond monotonic associations. If a correlation is present between the features in the dataset, the goal is to create a neural network that is capable of learning these patterns and reconstruct a feature that was missing. Should the model be able to do this, it can be suggested that it was able to capture the relationship. In this paper, models 1 and 2 are neural networks that reconstruct a missing feature using other existing features (see Appendix A for source code).

1.3 Autoencoder

Models 3, 4 and 5 are neural networks that aim to perform data imputation. The type of neural network chosen for this was an autoencoder. An autoencoder can be defined as "an artificial neural network that attempts to reproduce its input, i.e., the target output is the input." [7] The idea of an autoencoder is to reconstruct the input into another dimension with a different latent representation before reconstructing it back. These representations allow the neural network to identify key structures in the data. Figure 1 shows a simple visualization of how autoencoders function.

Typical autoencoders reduce the dimension of the input in latent representations to capture the key patterns. Upon experimentation, however, having the dimensions increase in latent representations yielded better learning results. This was observed in previous researches as well where "mapping the input data to a higher dimensional subspace creates representations capable of adding lateral connections, aiding in data recovery". [8]

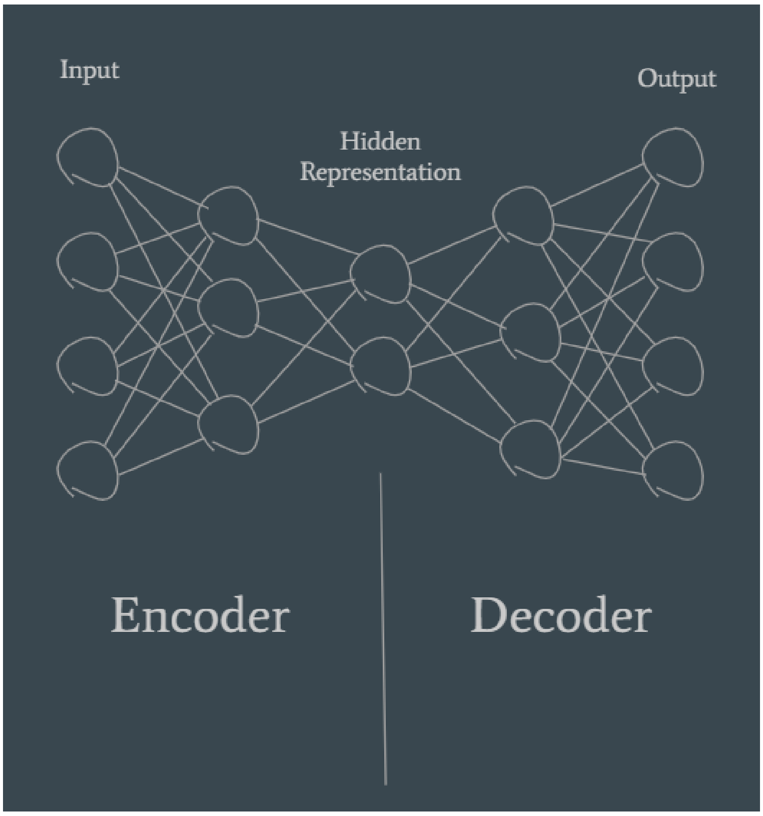

Figure 1: Basic Autoencoder - Encoder transforms input into hidden representation, decoder transforms the hidden representation back as output

These attributes make autoencoders a viable option for data imputation. By identifying the structure of the data, the model can potentially reconstruct the missing values as well. In this paper, models 3, 4, and 5 are autoencoders performing data imputation (see Appendix A for source code).

2 Models

All models can be found in appendix A.

2.1 Feature Reconstruction

Models 1 and 2 were implemented to evaluate a neural network's ability to reconstruct a feature. The models were trained on the iris dataset [9] and the Eckerle dataset [10] respectively. The former dataset was chosen due to its simplicity regarding the number of features and their linear relationships, while the latter dataset was chosen due to the bell-curve relationship between the features.

Model 1

The iris dataset consists of 150 instances of 4 features (600 values in total). The context was to imagine that the entire fourth column (petal width) is missing and the model has to construct this using the other three existing features (sepal width, sepal length, petal length). The petal width has a strong linear correlation with the petal length and the model's goal is to learn this linearity to construct the above mentioned variable.

Model 2

The Eckerle dataset was collected from a NIST study involving circular interference transmittance [10]. The x variable is the transmittance, and the y variable is the wavelength. The presence of a pattern similar to a bell curve provides a more challenging task and the aim of this model is to show that feature relatedness can be detected even with a relationship beyond a linear one. Model 2 reconstructs a single column similarly to model 1, except that it only has one other existing feature to use and the relationship is also more difficult to learn. The variables in the dataset have a significant difference in magnitude, hence the min-max normalization was used on the dataset as one of the advantages of this technique is that the it can "preserve exactly all relationships in the data". [13]

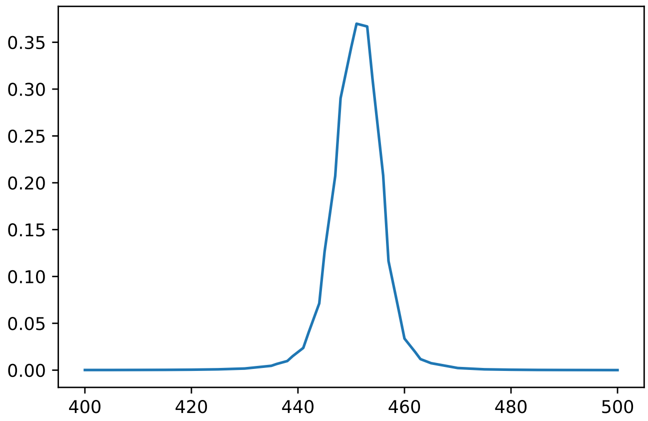

Figure 2: Transmittance plotted against the wavelength in Eckerle dataset

2.2 Data Imputation

Models 3, 4, and 5 were implemented in the form of an autoencoder to perform data imputation. The datasets that the models operated with consisted of linear correlations, non-linear correlations and an unknown correlation type respectively.

Model 3

The Wisconsin breast dataset was filtered in such a way that only features with linear correlations were kept, resulting in 6 features of 569 instances (3414 values in total). This model does not operate with a dataset consisting of a missing feature, but rather with missing values placed in random rows and columns. The third model aims to show that data imputation can be performed to a high degree on a real dataset based on linear relationships. Again, the features vary in sizes, hence the min-max normalization was applied.

Model 4

Data imputation was performed on a generated dataset consisting of correlations in the shape of a bell curve. Each feature has a different shape of a bell curve to see how well the model can learn for each shape. The data set has 6 features of 2000 instances (12000 values in total). Gaussian noise was added to create an imperfect shape that adds some variance similar to a real dataset. The success of this model then implies that data imputation using autoencoders is possible even with feature relatedness consisting of non-linear relationships.

Model 5

To test data imputation for a real dataset with an unknown correlation, model 5 was used on the Dutch census dataset consisting of 97 features of 6303 instances [12]. However, features that have more than 50% missing values were filtered out to ensure that the model is only trained on sufficiently available features, resulting in 3 features of 2546 instances (7638 values in total) The dataset contains real missing values with unknown feature relatedness. If the model successfully predicts the missing values that are manually inputted during the test phase, it can be concluded that deep learning can indeed perform data imputation successfully.

3 Design

Data cleaning slightly varied in each model depending on the needs, for instance normalization was only applied to datasets where magnitudes differed significantly. On one hand, some design choices were kept consistent, such as the splitting method and the loss function. On the other hand, specific parameters were tuned to maximize the learning potential of a neural network, for instance, the learning rate and the number of hidden layers.

3.1 Splitting Method

The splitting method chosen was the traintestsplit from scikit-learn with the default 8:2 ratio to split the data into training and test sets. The splitting method always used stratification to ensure that the neural network is exposed to the whole range of values. However, this is not always feasible for regression problems since a specific numeric value may only appear once and cannot be guaranteed to be in the training set. To tackle this, instead of regarding each value as a unique item, they were represented by spans. For example, in a binary classification problem with values "0" and "1", stratification can guarantee that the train set has an equal portion of both the values. In regression problems, the value can be for instance "5.52" for which the stratified split cannot guarantee it will be present in the train set since the value can be unique across the whole dataset. Instead, stratification used spans where it ensured that both the train and test sets had values in the range "5.50 - 6.00" which guaranteed that the model is trained on all ranges. Stratification was, therefore, possible as it allowed the training set to consist of values from all the ranges.

3.2 Activation Function

Despite Rectified Linear Unit (ReLU) being the most commonly used activation function, in certain contexts its variants such as Leaky ReLU can prove to be the better performer [14]. Based on hyper-parameter tuning, Leaky ReLU slightly outperformed its predecessor, resulting in the variant being used for all models.

3.3 Loss Function

The chosen loss function was the Mean Squared Error (MSE). Not only does this loss function apply well to regression problems, but its complexion also allows us to amplify large losses which consequently ensures that the errors do not cancel each other out.

3.4 Model Evaluation

This research paper used three different metrics for evaluating the models:

1. How accurate are the model predictions?
2. How well is the correlation preserved?
3. How well is the data distribution captured?

3.4.1 Losses: how accurately can the model predict

Train and test losses acquired from the mean squared error function were used for evaluating the accuracy of the predicted missing values. The test loss is the main metric for assessing the accuracy of the predictions. Additionally, the train loss was also carefully revised as its value relative to the test loss can determine if the model has been over-fitted.

3.4.2 Correlation Coefficient: how well is the correlation preserved

Correlation coefficients are used for evaluating how well the neural network can preserve feature relatedness when performing data imputation. Pearson's r-value is limited only to linear correlations, hence it is only used for evaluating models 1 and 3. The two widely used correlation coefficients for calculating feature relatedness in non-linear datasets are distance correlation [16] and maximal information correlation [17]. These two are similar and vary slightly based on the given relationship, but generally can be considered on par when it comes to their effectiveness. [15] For this paper, an already-implemented distance correlation was used [18] which can be found in appendix A.

3.4.3 Distance Correlation Explained

Distance correlation is similar to Pearson's R-value in the way that it also uses distance co-variance followed by normalization with distance standard deviation. In Pearson's correlation coefficient, the measure consists of calculating how the variables co-vary from the mean. In distance correlation, the measure calculates how the variables co-vary from all other data points by using distance matrices. This means that the correlation coefficient does not only take into account the distance between the points and their respective means, but also the distance between each individual points. The correlation coefficient, distance co-variance, and double-centered matrices are denoted as $Cor_D(x, y)$, $Cov_D^2(x, y)$, and $\hat{X}$ respectively in the formulas below:

$$\hat{X} = X_{ij} - \bar{X}_i - \bar{X}_j + \frac{1}{N^2}\sum_i^N\sum_j^N X_{ij}$$

$$Cov_D^2(x, y) = \frac{1}{N^2}\sum_i^N\sum_j^N \hat{X}_{ij}\hat{Y}_{ij}$$

$$Cor_D(x, y) = \frac{Cov_D(x, y)}{\sigma_D(x)\sigma_D(y)}$$

3.4.4 Paired T-Test: how well is the data distribution captured

The statistical procedure paired t-test was used for model evaluations. The paired t-test shows the mean difference between the two sets of observations. Based on this, comparisons can be made between the data distribution of the predicted values and the original values. This metric was only used for model 2 and model 4 since they operate with datasets that have a uniform data distribution in a shape of a bell curve.The two hypothesis are as follows:

- **Null Hypothesis:** The model yields a significantly different set of values compared to the original values
- **Alternative Hypothesis:** There is no significant difference between the original values and the predicted values yielded by the model

4 Results

Model 1: Feature reconstruction on IRIS dataset

The difference in Pearson's r-value was only 0.045 (table 1), suggesting that the neural network was able to capture the correlation of the original dataset. The results of model 1 show promising signs of neural network's capabilities regarding reconstruction of features using feature relatedness. Figure 3 shows that the model reconstructed the petal widths based on the linear correlation successfully, however, there was a low variance in the reconstructed values compared to the original values. This suggests that the model yielded predictions too strict to the linear correlation it detected. Assessing the model's accuracy in reconstructing the petal width, the loss for the unseen test set was 0.046 (table 6) for the iris dataset with an average value of 1.30.

The small sample size of 600 values limits the learning potential of model 1. Given a larger sample size in the magnitude of thousands, the model would potentially capture the variance as well and preserve it accordingly in the reconstructed values. Nevertheless, the findings provide insights regarding its ability to detect feature relatedness and perform feature reconstruction using the omnipresent linear correlation.

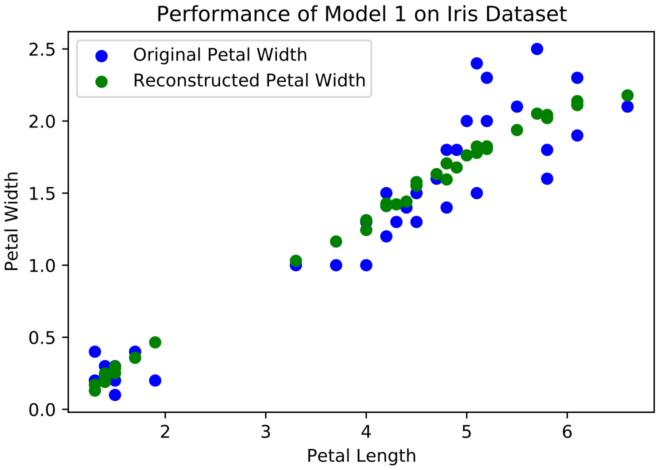

Figure 3: Reconstructed petal width using an unseen test set

	Original data	Predicted data	Difference
Model 1	0.953	0.998	**+0.045**

Table 1: Pearson correlation for IRIS dataset using Model 1

Model 2: Feature reconstruction on Eckerle dataset

In model 2, the aim was to see how well the neural network can detect feature relatedness in a dataset with a non-linear correlation. Figure 4 shows that the model managed to reconstruct values with a similar correlation shape to the original values.

The distance correlation of the original values and the reconstructed values are 0.463 and 0.473 respectively, yielding a difference of just 0.010. The reconstructed values show an identical level of variance compared to the original values, however even the original data had minimal variance hence it is not conclusive whether the model captured the variance or it simply reconstructed the values based on the strict correlation shape detected similarly to model 1. With a p-value of 0.992 at a 95% significance level, the null hypothesis was rejected, meaning that we cannot conclude that the model yielded a significantly different set of values compared to the original values.

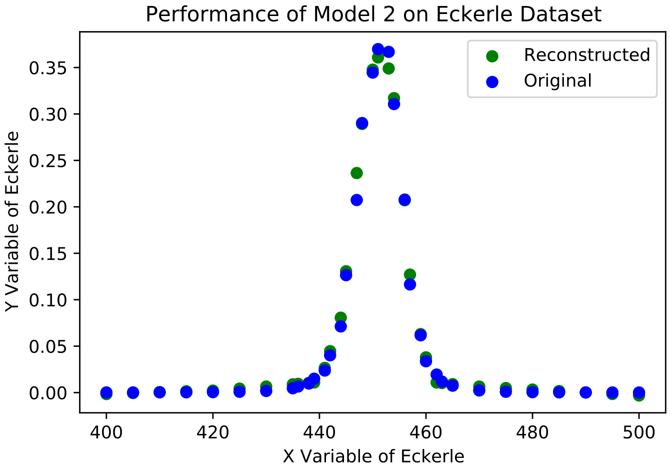

Figure 4: Reconstructed variable Y using an unseen test set

	Original data	Predicted data	Difference
Model 2	0.463	0.473	+**0.010**

Table 2: Distance correlation for Eckerle dataset using Model 1

Model 3: Data Imputation on Wisconsin breast dataset

The average value in the breast dataset was 0.27 and the loss achieved for the unseen test set was 0.0051 (table 6). One of the weaknesses of the traditional mean imputation technique is that the correlation of the imputed values do not give an accurate representation of the correlation in the original data [3]. The results of model 3, however, show that neural networks are capable of preserving correlation when performing data imputation. The highest correlation difference was 0.054 between the original and predicted values of feature 1 (table 3). For feature 1, the predicted values of model 3 actually have more variance than the original values (figure 5). Unlike traditional mean imputation, the variance is higher which may be more representative of real datasets. However, in this case, the linear correlation is especially strong in the original values, hence the variance in the predicted values may represent inaccuracy in predicted values.

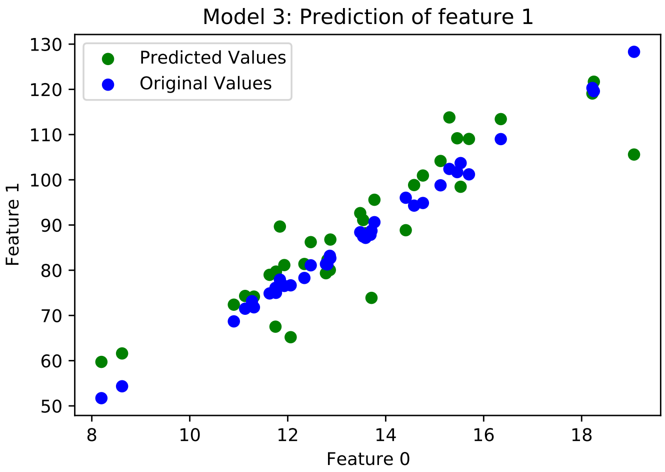

Figure 5: Reconstructed perimeter mean using an unseen test set

	Original data	Predicted data	Difference
Column 1	0.999	0.945	**-0.054**
Column 2	0.992	0.972	**-0.020**
Column 3	0.960	0.990	+**0.030**
Column 4	0.960	0.971	+**0.011**
Column 5	0.946	0.966	+**0.020**

Table 3: Pearson R between the Column 0 (original data) and other columns (both original and predicted)

Model 4: Data Imputation on custom dataset

For the generated dataset with a correlation of a bell curve, the traditional mean imputation technique would yield predictions that do not conform to the real relationship as it takes the mean between the available data points. Model 4, however, showed that deep learning is capable of preserving the correlation of the dataset when performing data imputation. Figure 6 shows that the predicted values have a significant resemblance to the shape of the original values. However, such results are not consistent with all the features. For instance, the model's predictions had a variance significantly higher to the original values for feature 1 (figure 13 in appendix B). The average difference in the distance correlation of all features was only 0.042. Furthermore, for a dataset with an average value of 124, the loss for the unseen test set was 86.034. Since the relationship was similar to a normal distribution, a paired t-test was performed to see how well the model was able to capture the data distribution. With a p-value of 0.155 at a 95% significance level, the null hypothesis was rejected, meaning that we cannot conclude that the model yielded a significantly different set of values compared to the original values.

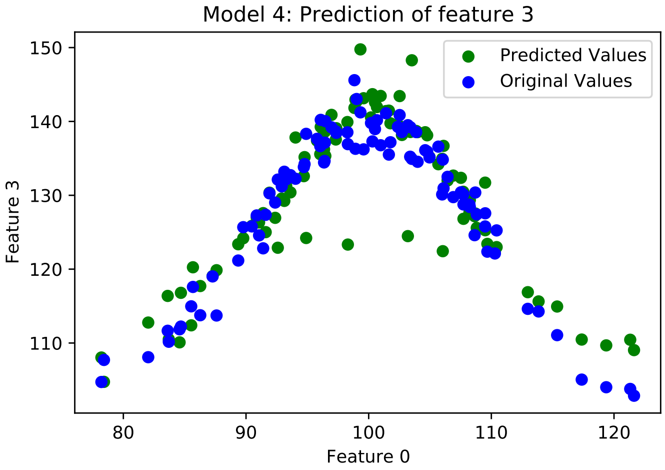

Figure 6: Reconstructed perimeter mean using an unseen test set

	Original data	Predicted data	Difference
Column 1	0.556	0.493	**-0.063**
Column 2	0.628	0.550	**-0.078**
Column 3	0.556	0.527	**-0.029**
Column 4	0.533	0.519	**-0.014**
Column 5	0.533	0.559	+**0.026**

Table 4: Distance correlation between the Column 0 (original data) and other columns (both original and predicted)

Model 5: Data Imputation on Dutch historical census dataset

The results of model 5 showed that feature relatedness has to have a strong presence in order for the model to learn and predict effectively. Initially, figure 7 shows that the dataset has a linear correlation to a limited extent. However, upon removing the extreme outliers (figure 8), it can be seen that the variance in the dataset is immensely larger than the previous datasets which caused problems for this model to learn and predict. For a dataset with an average value of 0.000059, the loss for the unseen test set was 0.0000030.

Even though autoencoders have the potential to perform data imputation to a large extent, it is important that the datasets have strong correlations. Without strong presence of feature relatedness, it can be seen from experiment 5 that the model cannot predict the high range of disparity in the dataset.

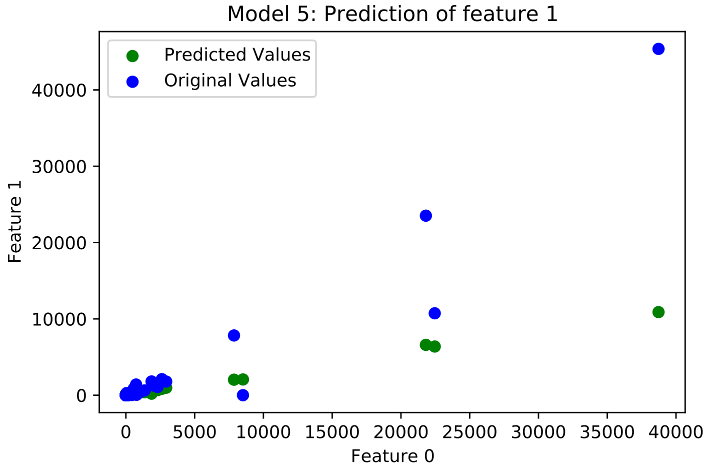

Figure 7: Predicted Rotterdam census using an unseen test set

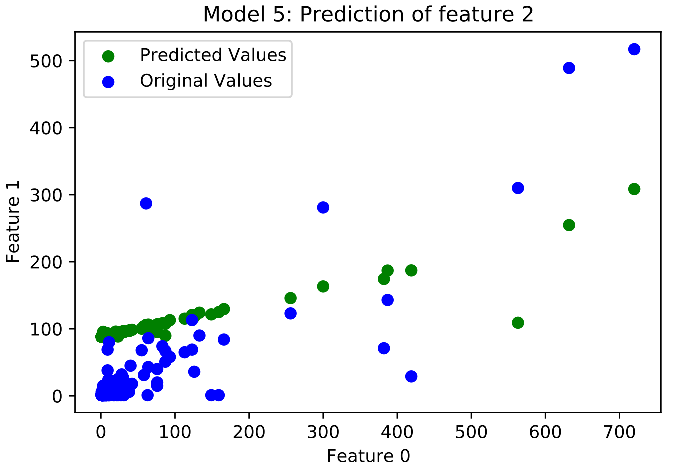

Figure 8: Predicted Rotterdam census without outliers (30 highest data points)

	Original data	Predicted data	Difference
Column 1	0.624	0.860	+**0.236**
Column 2	0.943	0.950	+**0.007**

Table 5: Distance correlation between the Column 0 (original data) and other columns (both original and predicted)

4.1 Losses

	Avg. Value	Train Loss	Test Loss
Model 1	1.30	0.033	0.046
Model 2	0.20	0.00028	0.00026
Model 3	0.27	0.0020	0.0051
Model 4	124	63.17	86.034
Model 5	*0.000059	0.000042	0.0000030

Table 6: Train and test losses for each model. The losses of autoencoders only take into account imputed data. *Median used instead of mean for calculating average due to extreme outliers.

4.2 Statistical Tests

	Average P-Value	Average T-Value	Null Hypothesis
Model 2	0.992	-0.010	Rejected
Model 4	0.155	0.679	Rejected

Table 7: Statistical significance (rejected at 95% significance level)

5 Discussion

There are some variables in this research that can be tweaked to provide further insights. Models 2 and 4 showed that both feature reconstruction and data imputation can be performed by deep learning to a large extent even on non-linear correlations. However, this is only limited to a correlation in the shape of a bell curve. Stronger conclusions could be drawn if similar results were obtained in future experiments that involved datasets with other types of correlations regarding both linear and non-linear relationships, such as exponential correlations.

One of the variables that were kept constant regarding the training of the neural networks was the amount of missing data that the model used for training. This research removed a fixed 20% of the available data in all the experiments. Future research, however, could evaluate how different portions of missing data affect the models' learning potential, or investigate if there is any preferred portion for different types of datasets and correlations.

When it comes to autoencoders, one of the questions that may arise is 'how does the dimensionality of the input affect the performance of autoencoders when performing data imputation?' This research used five different datasets which consisted of two to six features. The range of the dimensions is narrow, hence it does not provide any significant insight into the effect of data dimensionality on the data imputation performance. In future work, this could be investigated by experimenting on datasets with a wider range of features.

Each model was implemented to provide a different insight, such as whether feature reconstruction/data imputation can be performed, and on which type of correlation can this be accomplished. As a result, each model used only one dataset. Subsequently, the findings provide insufficient depth regarding the neural network's structure. With the results from this research, it cannot be concluded how complex the networks have to be for maximizing performance for the different tasks and correlation types. For further research, various ranges of complexities regarding the neural network's structure should be assessed for different datasets.

6 Conclusion

This paper has shown that neural networks are capable of detecting both linear and non-linear correlations. As the research further shows, the potential of doing this also allows data imputation to be performed to a large extent. The findings showed that using autoencoders to predict missing data has its strengths compared to traditional techniques, however, the results of model 5 showed that a strong correlation has to be present in the first place to perform data imputation accurately. The accuracy in predictions are comparable to traditional techniques for linear correlations, but the main take away is the performance in data imputation that neural networks can deliver for non-linear relationships while also preserving the correlation coefficient measures. This was evaluated with distance correlation which showed that the imputed values were insignificantly different to the original values. However, these findings are limited to a narrow range of feature dimensions and correlation types. The feature dimensions spanned three to six features, while the linear correlation were limited to positive correlations and non-linear correlations were limited to bell-curved correlations. Hence, future research could enhance the conclusions from this research regarding autoencoders and their performance on data imputation by diversifying the types of datasets. This could be achieved by testing a variety of magnitudes regarding feature dimensionality while also exploring the different types of correlations for both linear and non-linear correlations. Furthermore, future research could also evaluate the complexity of neural networks for different types of datasets. This would yield an insight into how compound the models need to be for different correlations to be captured, ranging from simple to complex relationships.

References

[1] Roderick J. A. Little, Donald B. Rubin. *The Analysis of Social Science Data with Missing Values.* Sociological Methods and Research, 1989.

[2] Donald B. Rubin. *Inference and missing data.* Educational Testing Service, Princeton, New Jersey, 1976.

[3] Amanda N. Baraldi, Craig K. Enders. *An introduction to modern missing data analyses.* Arizona State University, United States, 2009.

[4] James L. Peugh, Craig K. Enders. *Missing Data in Educational Research: A Review of Reporting Practices and Suggestions for Improvement.* Review of educational research, 2004.

[5] Marina Soley-Bori. *Dealing with missing data: Key assumptions and methods for applied analysis.* Boston University, 2013.

[6] P. Schober, C. Boer, L.A. Schwarte. *Correlation Coefficients: Appropriate Use and Interpretation.* Anesthesia and Analgesia, 2018.

[7] David Meyer. *Introduction to Autoencoders.* 2015.

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

[9] R.A. Fisher. *Iris Data Set.* 1936.

[10] Eckerle. *Circular Interference Transmittance Study.* K. and NIST (1979). Available at https://www.itl.nist.gov/div898/strd/nls/data/eckerle4.shtml

[11] William H. Wolberg, W. Nick Street, Olvi L. Mangasarian. *Breast Cancer Wisconsin (Diagnostic) Data Set.* Computer Sciences Dept. University of Wisconsin, 1995.

[12] Centraal Bureau voor de Statistiek (CBS); *Thematic collection: Dutch censuses 1795-1971. DANS.* NIWI-KNAW; (1999): https://doi.org/10.17026/dans-x6t-brh5

[13] T.Jayalakshmi, Dr.A.Santhakumaran. *Statistical Normalization and Back Propagation for Classification.* International Journal of Computer Theory and Engineering, 2011.

[14] Bing Xu, Naiyan Wang, Tianqi Chen, Mu Li. *Empirical Evaluation of Rectified Activations in Convolution Network.* University of Alberta, Nov 2015.

[15] Michael Clark. *A Comparison of Correlation Measures.* Center For Social Research, University of Notre Dame, 2013.

[16] Gabor J. Szekely, Maria L. Rizzo, Nail K. Bakirov. *Measuring and testing dependence by correlation of distances.* Bowling Green State University, USC Russian Academy of Sciences, 2007.

[17] David N. Reshef, Yakir A. Reshef, Hilary K. Finucane, Sharon R. Grossman, Gilean McVean, Peter J. Turnbaugh, Eric S. Lander, Michael Mitzenmacher, and Pardis C. Sabeti. *Detecting Novel Associations in Large Datasets.* Science, 2011.

[18] Satrajit Ghosh. *Distance Correlation in Python.* GitHub Gist repository, https://gist.github.com/satra/aa3d19a12b74e9ab7941, 2014.

A GitHub Repository

Source Code: `https://github.com/fteslal1/thesis`

B Additional Results of Model 3

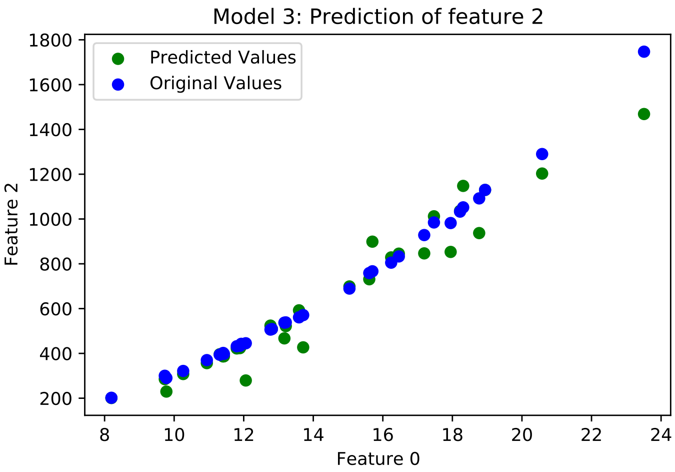

Figure 9: Reconstructed perimeter mean using an unseen test set

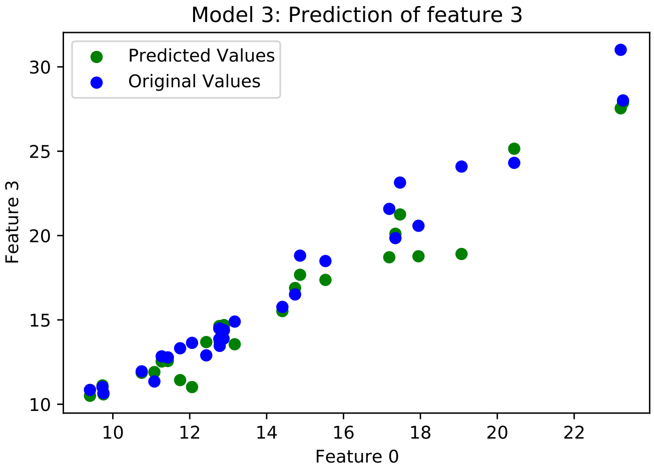

Figure 10: Reconstructed perimeter mean using an unseen test set

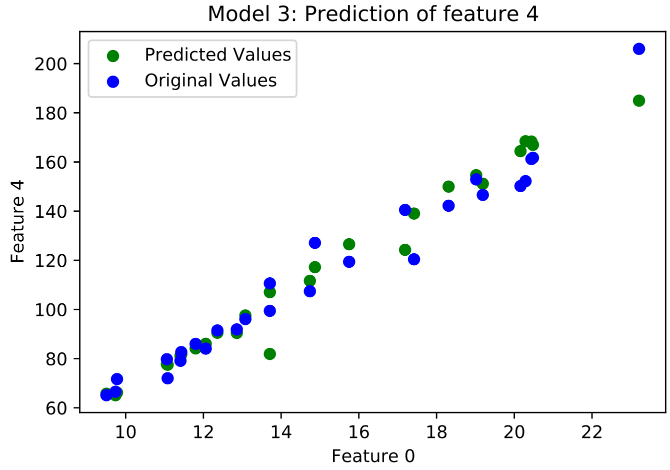

Figure 11: Reconstructed perimeter mean using an unseen test set

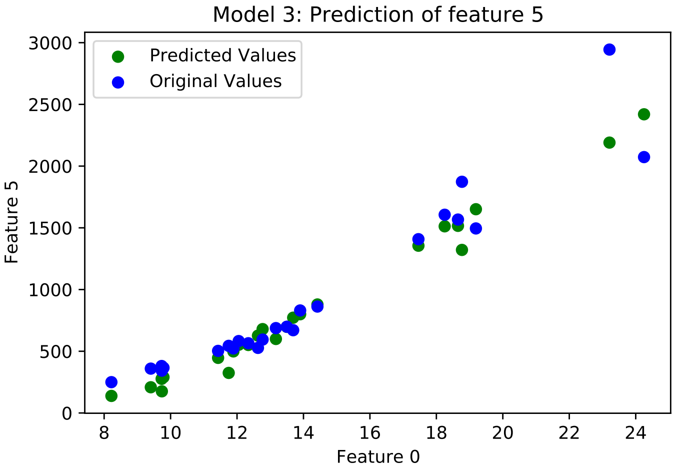

Figure 12: Reconstructed perimeter mean using an unseen test set

C Additional Results of Model 4

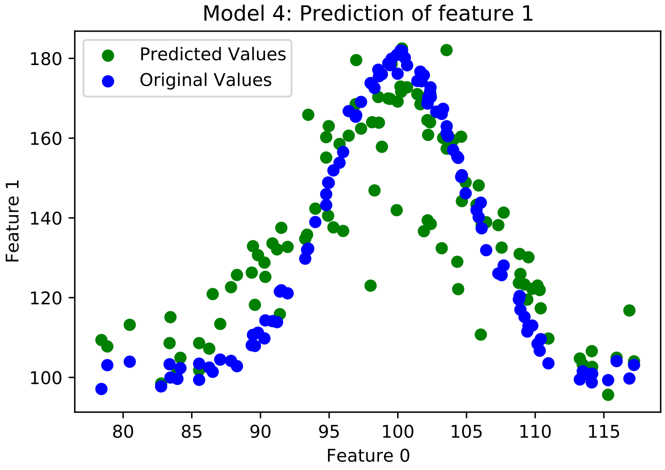

Figure 13: Data imputation of feature 1 from the generated dataset

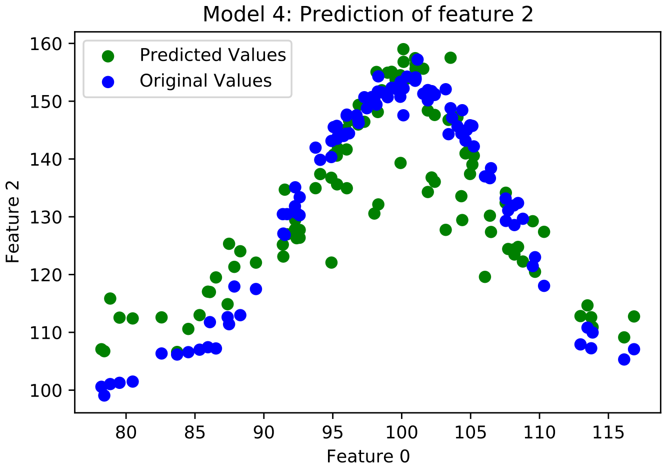

Figure 14: Data imputation of feature 2 from the generated dataset

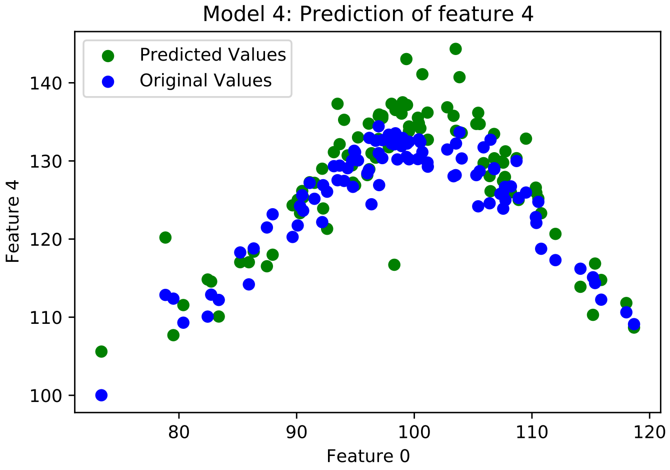

Figure 15: Data imputation of feature 4 from the generated dataset

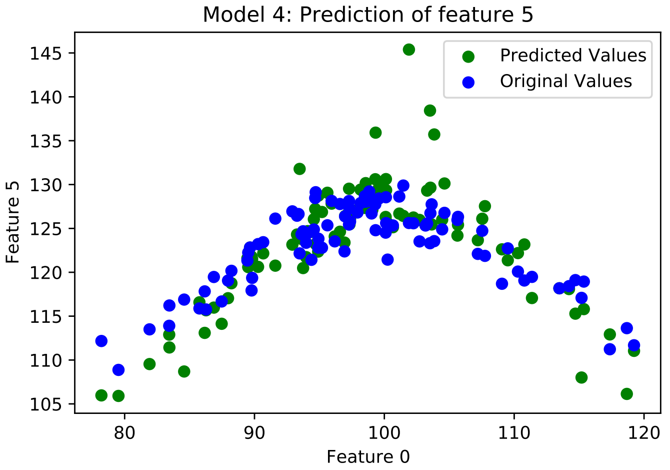

Figure 16: Data imputation of feature 5 from the generated dataset