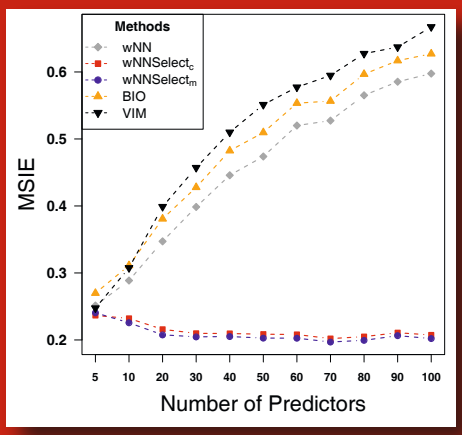
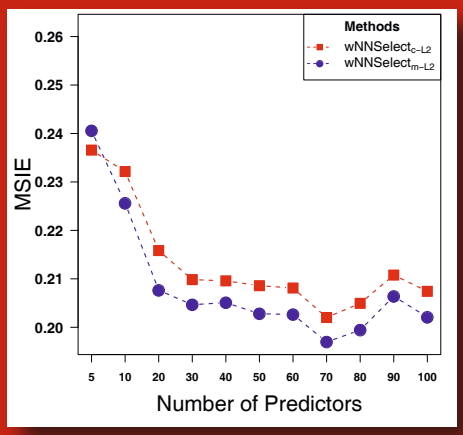
Shahla Faisal

Nearest Neighbor Methods for the Imputation of Missing Values in Low and High-Dimensional Data





Cuvillier Verlag Göttingen
Internationaler wissenschaftlicher Fachverlag

Shahla Faisal

Nearest Neighbor Methods for the Imputation of Missing Values in Low and High-Dimensional Data

Dissertation an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

Eingereicht am 12.01.2018

Shahla Faisal

Nearest Neighbor Methods for the Imputation of Missing Values in Low and High-Dimensional Data

Dissertation an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig-Maximilians-Universität München

Eingereicht am 12.01.2018

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.
1. Aufl. - Göttingen: Cuvillier, 2018
Zugl.: (LMU) München, Univ., Diss., 2018

Erster Berichterstatter: Prof. Dr. Gerhard Tutz
Zweiter Berichterstatter: Prf. Dr. Martin Spieß
Dritter Berichterstatter: Prof. Dr. Christian Heumann

Tag der Disputation: 20.02.2018

Nonnenstieg 8, 37075 Göttingen
Telefon: 0551-54724-0
Telefax: 0551-54724-21
www.cuvillier.de

1. Auflage, 2018
Gedruckt auf umweltfreundlichem, säurefreiem Papier aus nachhaltiger Forstwirtschaft.

ISBN 978-3-7369-9741-7
eISBN 978-3-7369-8741-8

To my parents

Acknowledgements

I owe my deepest gratitude to my supervisor Prof. Dr. Gerhard Tutz, for his motivation, immense knowledge, and for reading all the drafts so quickly and thoroughly that helped me to finish this dissertation.

I extend my gratitude to my co-supervisor Prof. Dr. Christian Heumann for his ever available guidance and support and sharing his great knowledge for missing data. Without the continuous optimism, constructive suggestions and feedback of these two persons, this thesis would hardly have been completed.

I would also like to thank Prof. Dr. Martin Spieß who kindly agreed to serve as the third reviewer and to Prof. Dr. Helmut Küchenhoff and Prof. Dr. Thomas Augustin, for being part of the doctoral committee. My special thanks go to Christian and Gerhard for providing me financial support that gave me the opportunity to present my work in many valuable international conferences around the globe.

Besides, this thesis would not have been possible without the help encouragement and support of many people. My special thanks go to ...

... all my colleagues at the Department of Statistics LMU, specially Gunther, Margret and Moritz.

... my teachers in Pakistan, Prof. Dr. M. Inayat Khan and M. Iqbal Javed (late), who put their faith in me and urged me to do better.

... my family specially my parents, whose decisions in life changed my future entirely, and who taught me to trust in Allah and believe in hard work.

... my husband, for always pushing me forward whenever I needed. Thanks for your continuous support and for encouraging me to believe in myself.

... my cutest little sunshine, my daughter Meerub, who completed my life with her innocent smiles.

Zusammenfassung

Durch die technische Entwicklung der letzten Jahrzehnte hat sich die Verfügbarkeit großer Datenmengen erheblich gesteigert. Eines der großen Probleme in der Datenanalyse ist das Auftreten fehlender Werte. Das Problem ist insbesondere dann schwierig zu bearbeiten, wenn die Variablen unterschiedliches Skalenniveau (metrisch, nominal oder ordinal) aufweisen. Was immer der Grund für das Auftreten fehlender Werte ist, das Problem tritt in allen Bereichen angewandter Forschung auf.

Der inadequate Umgang mit fehlenden Werten kann zu verzerrten Analyseergebnissen und fehlerhafter Inferenz führen. Die üblichen Analyseverfahren verlangen vollständige Daten, bei denen alle Werte beobachtet wurden. Der einfache Weg, nur Daten in die Analyse aufzunehmen, die vollständig beobachtet wurden, ignoriert verfügbare Information und führt typischerweise zu unzureichenden Analyseverfahren. In der vorliegenden Dissertation werden Imputationsverfahren auf der Basis von Nächste -Nachbarn Methoden entwickelt, die sowohl für niedrig- als auch hochdimensionale Datensätze geeignet sind.

Nächste-Nachbarn (NN-) Verfahren wurden erfolgreich eingesetzt in Klassifikationsproblemen, im Clustering und zur Imputation fehlender Werte. Im ersten Teil der Arbeit werden verbesserte Verfahren entwickelt, die zu deutlich besseren Imutationen führen. Insbesondere die entwickelte Lokalisierungstechnik, die auf einer Gewichtung der Nächsten Nachbarn beruht, hat sich als sehr effizient erwiesen. Im hochdimensionalen Fall wird eine Selektion der Variablen benutzt, die die Korrelation der Variablen explizit berücksichtigt. Dies unterscheidet das Verfahren deutlich von bisher verfügbaren Methoden. Sowohl Simulationsstudien als auch die Analyse realer Datensätze zeigen, dass mit der Methode gute Imputationswerte erzielt werden. Über den klassischen Fall metrischer Daten hinaus werden Verfahren für binäre und mehrkategoriale, sowie den Fall unterschiedlichen Datentyps entwickelt. Es wird gezeigt, dass die Verfahren auch in Fällen, in denen die Anzahl der Beobachtungen kleiner ist als die Anzahl der Variablen, effizient einsetzbar sind und bessere Ergebnisse erzielen als bisher verfügbare Alternativen.

Im zweiten Teil der Arbeit werden analytische Verfahren zur Inferenz in Datensätzen mit imputierten Werten entwickelt. Es wird ein Verfahren vorgeschlagen, das Bootstrap-Techniken mit der Nächsten-Nachbarn-Imputation kombiniert. Darüberhinaus wird die Klassifikationsgüte von diskriminanzanalytischen Verfahren, die auf unterschiedliche Imputationen aufsetzen, evaluiert.

Im letzten Teil der Arbeit wird ein multiples Imputationsverfahren auf der Basis der Nächsten Nachbarn entwickelt. Multiple Imputation ist insbesondere deswegen attraktiv, weil sie es ermöglicht, die Unsicherheit, die durch die Imputation erzeugt wird, explizit in der Inferenz zu berücksichtigen. Es werden explizit zwei unterschiedliche Verfahren der multiplen Imputation vorgeschlagen und gezeigt, dass die erzielten Ergebnisse besser oder zumindest vergleichbar mit den von den besten existierenden Verfahren zu erwartenden Ergebnissen sind.

Summary

Nowadays, due to the advancement and significantly rapid growth in the technology, the collection of high-dimensional data is no longer a tedious task. Regardless of considerable advances in technology over the last few decades, the analysis of high-dimensional data faces new challenges concerning interpretation and integration. One of the major problems in high-dimensional data is the occurrence of missing values. The problem is in particular hard to handle when the distributional forms of the variables are different or the variables are measured on different measurement scales (e.g. binary, multi-categorical, continuous, etc.). Whatever the reason, missing data may occur in all areas of applied research.

The inadequate handling of missing values may lead to biased results and incorrect inference. The standard statistical techniques for analyzing the data require complete cases without any missing observations. The deletion of the cases with missing information to obtain complete data will not only cause the loss of important information but can also affect inferences. In this dissertation, different imputation techniques using nearest neighbors are developed to address the missing data issues in high-dimensional as well as low dimensional data structures.

The nearest neighbors (NN) methods is a well known technique that has been successfully used in classification, clustering and imputation of data. In the first part of the thesis, improved procedures are developed that lead to much better imputations. In particular, a localized approach to missing data imputation that uses a weighted average of nearest neighbors based on L_q distances is developed. For the high-dimensional case, a new distance function that explicitly uses the correlation among variables is proposed. In contrast to classical approaches, a main advantage is that the proposed method automatically selects the relevant variables that contribute to the distance. The results from simulation studies as well as real studies show that the weighted distance procedure can successfully handle missing values for high dimensional data structures. In addition, extensions to the binary, multi-categorical and mixed type data are also developed. It is shown in simulations, that the proposed imputation method works efficiently even when the number of samples is smaller than the number of variables. The method typically outperforms the considered competitors.

The second part of the thesis presents analytic techniques for inference from an imputed dataset in which missing values have been replaced by the proposed nearest neighbors imputation method. A novel approach, that combines bootstrap techniques with the nearest neighbors imputation, is proposed. In addition, the classification accuracy of discriminant analytical methods based on different imputation methods is evaluated.

In the last part of the thesis, a multiple imputation procedure based on the neighbor neighbors is developed. In particular, multiple imputation is attractive because it allows the uncertainty generated by the imputation to be explicitly considered in the inference. Explicitly, two different methods of multiple imputation are proposed, and it is shown that the results achieved are better or at least comparable to the results expected from the best existing methods.

Contents

1. Introduction

Missing data are ubiquitous problem in every field of research. The handling of the missing values in an inadequate manner may lead to biased results and inference based on them are, in turn, misleading. The standard statistical techniques for analyzing data require complete cases without any missing observations. The deletion of the cases with missing information to obtain complete data will not only cause the loss of important information but can also affect inferences. In this dissertation, different imputation techniques using nearest neighbors are developed to address the missing data issues in high-dimensional as well as low dimensional data structures.

Due to the advancement and rapid growth in technology, the collection of high-dimensional data is no longer a tedious task. Regardless of considerable advances in technology over the last few decades, one has to face challenges when dealing with the high-dimensional data, specially regarding the analyses, interpretation and integration. One of the major problems, that an analyst has to face in high-dimensional data, is the occurrence of missing values. In particular, the situation becomes worse, when the distributional forms of the variables are different or these variables have been measured at different measurement scales (e.g. binary, multi-categorical, continuous, etc.). Whatever the reason, missing data may occur in all areas of applied research.

Nearest neighbors is a well known technique that has been successfully used in classification, clustering and imputation of data. A key issue in classical nearest neighbor imputation is the selection of the suitable number of nearest neighbors k. Another important topic, in particular in higher dimensions, is the selection of the relevant dimensions, since the computation of distances may suffer from the curse of dimensionality yielding poor performance in high-dimensional settings. These problems are handled by a novel approach to compute the weights on nearest neighbors.

In the first part of the thesis, some improved nearest neighbors imputation methods are developed that have better better performance than the classical NN imputation. In particular, a localized approach to missing data imputation that uses a weighted average of nearest neighbors based on L_q distances is developed. For the high-dimensional case, a new distance function that explicitly uses the correlation among variables is proposed. In contrast to classical approaches, a main advantage is that the proposed method automatically selects the relevant variables that contribute to the distance. The results from simulation studies as well as real studies show that the weighted distance procedure can successfully handle missing values for high dimensional data structures. In addition, extensions to the

binary, multi-categorical and mixed type data are also developed. It is shown in simulations, that the proposed imputation method works efficiently even when the number of samples is smaller than the number of variables. The method typically outperforms the considered competitors.

Imputation is not the ultimate goal of any analysis, an estimate with smaller imputation error may not perform well in the downstream analysis. In particular, treating the imputed data just as the complete is not recommended. This is the starting point for the second part of the thesis. Several data analytic techniques for inference from an imputed dataset in which missing values have been replaced by the proposed nearest neighbors imputation method have been proposed. A novel approach that combines the nearest neighbors imputation with bootstrap resampling estimation is suggested to obtain valid bootstrap inferences in a regression model. More specifically, imputing the bootstrap samples in the exact same way as the original data was imputed produces correct bootstrap estimates. The classification accuracy of the imputed data using different imputation methods is also compared.

Single imputation methods provide a single value as an estimate of the missing value, and thus do not account for the uncertainty of imputation. In contrast, multiple imputation takes this uncertainty into acount by providing more than one plausible values corresponding to each missing value in the data. Multiple imputation is a preferred choice of data analysts, due to its flexibility and since it can applied in a wide variety of missing data scenarios. In the presence of high-dimensional data ($p \gg n$), the missing values might be a more serious issue as the existing softwares/packages may fail. A non-parametric approach for multiple imputation in combination with the nearest neighbors is suggested. In particular, an algorithm that combines the bootstrap resampling with the nearest neighbors is given. The other algorithm uses a sequential procedure for nearest neighbors imputation of the missing values. The method successfully imputes missing values also in high-dimensional settings, in which existing software tend to fail. Using a variety of simulated data with MCAR and MAR missing patterns, the proposed algorithm is compared to existing methods. The performance is evaluated by using mean squared imputation errors and inference results obtained for the imputed data. Various measures are used to compare methods, including mean squared errors of estimated regression coefficients, their standard errors, confidence intervals and their coverage probabilities. The simulation results, for both cases $n < p$ and $n > p$, show that the sequential imputation using weighted nearest neighbors can be successfully applied to a wide range of data settings and outperforms or is close to the best when compared to existing methods.

Guidelines through the Thesis

This thesis consists of 10 chapters and 4 appendices. Due to the close interdependence, some paragraphs contain a certain degree of overlap with regard to content. These overlaps

are consciously retained to enhance comprehensibility and allow for a separate reading of the single chapters.

Chapter 2 discusses the general concepts regarding missing values. It gives an introduction to the issues that arise in the presence of missing data as well as the current methodology and literature to handle these problems.

This thesis can be divided into three main parts which are dedicated to single imputation (Chapter 3 to 6), impact of imputation on inference and classification (Chapter 7 and 8) and multiple imputation (Chapter 9). In order to keep the single chapters self-contained, every chapter contains separate introductions to the relevant topics and a separate conclusion. Therefore, every chapter can also be read separately.

Chapter 3 and Chapter 4 deal with missing values problem where the covariates are metric in nature.

Chapter 5 extends the imputation method for binary and multi-categorical data.

Chapter 6 is dedicated to mixed-type data.

Chapter 7 is based on using bootstrap sampling to reach valid inferences from imputed data.

Chapter 8 investigates the impact of imputation on the classification of data.

Chapter 9 presents multiple imputation algorithms.

Throughout the thesis, the scalars are represented by lowercase letters (e.g. x), column vectors are represented by boldfaced lowercase letters (e.g. $\mathbf{x}$), the row vector as the transpose of the column vector (e.g. $\mathbf{x}^T$), and the matrices are represent by boldfaced uppercase letters (e.g. $\mathbf{X}$).

In the following we present the summaries of the individual chapters.

In **Chapter 3** we present improved versions of the nearest neighbor imputation method. First, a weighted nearest neighbor imputation method based on L_q distances is proposed. It is demonstrated that the method tends to have a smaller imputation error than other nearest neighbor estimates. We then consider weighted- neighbor imputation methods that use distances for selected covariates. The careful selection of distances that carry information about the missing values yields an imputation tool that can outperform competing nearest neighbor methods . This approach performs well, especially when the number of predictors is large. The methods are evaluated in simulation studies and with several real data sets from different fields.

Chapter 4 focuses on high-dimensional data setting in the real world situations and applies the methods developed in Chapter 3 to the real data. High-dimensional data like gene expression and RNA-sequences often contain missing values. The subsequent analysis and results based on these incomplete data can suffer strongly from the presence of these missing values. Several approaches to imputation of missing values in gene expression data have

been developed but the task is difficult due to the high-dimensionality (number of genes) of the data. In this chapter, an imputation procedure is proposed that uses weighted nearest neighbors. Instead of using nearest neighbors defined by a distance that includes all genes the distance is computed for genes that are apt to contribute to the accuracy of imputed values. The method aims at avoiding the curse of dimensionality, which typically occurs if local methods as nearest neighbors are applied in high-dimensional settings. The proposed weighted nearest neighbors algorithm is compared to existing missing value imputation techniques like mean imputation, KNNimpute and the recently proposed imputation by random forests. We use RNA-sequence and microarray data from studies on human cancer to compare the performance of the methods. The results from simulations as well as real studies show that the weighted distance procedure can successfully handle missing values for high-dimensional data structures where the number of predictors is larger than the number of samples. The method in chapter 4 typically outperforms the considered competitors.

In **Chapter 5** the weighted nearest neighbors imputation method is extended for the binary and categorical variables. While various imputation methods are available for metrically scaled variables, methods for categorical data are scarce. An imputation method that has been shown to work well for high-dimensional metrically scaled variables is the imputation by nearest neighbor methods. One has to use specific distances or similarity measures, which are typically based on contingency tables for categorical data. The Euclidean or variants of the Minkowski distance give an equal importance to all the variables in the data matrix when computing the distance. But for a larger number of variables, the equal weighting ignores the complex structure of correlation/association among these variables. In Chapter 5, we propose a weighted nearest neighbors approach based on dummy variables to impute missing values in categorical variables. As demonstrated in the chapter, better distance measures are obtained by utilizing the association between variables. More specific, we propose a weighted distance that explicitly takes the association among covariates into account. Strongly associated covariates are given higher weights forcing them to contribute more strongly to the computation of the distances than weakly associated covariates. The performance of different imputation methods is compared in terms of the proportion of falsely imputed values. Simulation results show that the weighting of attributes yields smaller imputation errors than existing approaches. A variety of real data sets is used to support the results obtained by simulations.

Chapter 6 is dedicated to mixed type of data. One has to deal with combination of continuous and nominal variables in many real world applications, therefore the methods to impute mixed data become more important. Since the multiple imputation techniques fail to impute high-dimensional missing data, the nonparametric single imputation methods are gaining more popularity. We propose an improved version of the popular nonparametric nearest neighbors method which uses information only on potentially relevant neighbors to impute missing values. More specifically, We introduce a distance function that is more appropriate for mixed data. It is an extension of Tutz and Ramzan (2015) and uses information on association among variables. A particular advantage of the proposed method is that while imputing the missing values, it simultaneously takes into account the similarities between samples and the relationships between covariates. The performance of the pro-

posed method is investigated under a variety of data settings. The results show a smaller imputation error and better performance when compared to other approaches. It is shown that the proposed imputation method works efficiently even when the number of samples is smaller than the number of variables.

In **Chapter 7** we present analytic techniques for inference in datasets in which missing values have been replaced by nearest neighbors imputation methods. It is not advisable to treat the imputed data just as the complete data. To apply the existing methods for analyzing the data, for example, to estimate the variance and/or statistical inference will probably produce invalid results because these methods do not account for the uncertainty of imputations. To overcome this, we presents a bootstrap algorithm that combines the nearest neighbors imputation with bootstrap resampling estimation to obtain valid bootstrap inference in a linear regression model. The proposed procedure that provides estimated values for the missings which have not only smaller MSEs, but also provide better inference, when one variable is a response variable and the rest of the variables are predictors in a linear regression model. The confidence intervals for the regression coefficients are constructed using bootstrap sampling. Simulation results show that the suggested imputation method provides promising results and the bootstrap has the desired nominal coverage.

Chapter 8 investigates the impact of imputation on the classification of the data. Although some methods in machine learning can be tolerant to the presence of missing values, many statistical methods require a complete data matrix and so does classification methods. We compare the accuracy of different classifiers using the imputed data obtained by different imputation methods. The results show that some classification methods are robust to the presence of missing values and hence their accuracy is not affected much. While some methods really influenced by imputed values. Our study shows that the impact of imputation varies between data sets and different classifiers. Overall the proposed weighted nearest neighbors imputation provided best results in four out of six datasets considered and second best for the remaining two datasets. The random forests imputation provided the second best classification results in our experiments, whereas `MEAN` imputation was seen to yield poor performance. The results based on misclassification error and Brier score show that imputation improves the classification accuracy and in particular, beneficial for the higher amounts of missing data. When the amount of missing data is small, there is not much difference among the imputation methods.

In **Chapter 9**, multiple imputation methods are proposed based on the sequential nearest neighbors. In the presence of high-dimensional data ($p \gg n$), the missing values lead to a more serious issue as the existing softwares/packages fail respond. In this chapter we present a multiple imputation method based on nearest neighbors which sequentially imputes the missing values. The specific distances are computed using the information of correlation among the target and candidate predictors. Thus only the relevant predictors contribute to the computation of distances. The method successfully imputes missing values in high-dimensional settings. Using a variety of simulated data with MCAR and MAR missing patterns, the proposed algorithm is compared to three existing methods, Multivariate Imputation by Chained Equations (MICE), Iterative robust model-based imputation

(IRMI) and Amelia. Our extensive simulation results show that the sequential imputation using weighted nearest neighbors can be applied to a wide range of data settings considered here and outperforms or is close to the best when compared to existing methods.

The thesis concludes with a short summary and an outlook to potential aspects for future research in **Chapter 10**.

Contributing Manuscripts

Parts of this thesis were published as articles in peer reviewed journals or as pre-prints at the Cornell University Library's open access archive `arXiv.org`. Other parts were published in proceedings of scientific conferences or as technical reports at the Department of Statistics of the Ludwig-Maximilians-Universität München. All manuscripts have been written in cooperation with (supervising) coauthors. In the following, chapter by chapter all contributing manuscripts are listed together with a declaration of the personal contributions of the respective authors:

Chapter 3 is based on the published paper

- Tutz and Ramzan (2015). Improved methods for the imputation of missing data by nearest neighbor methods. *Computational Statistics & Data Analysis*, 84–99. https://doi.org/10.1016/j.csda.2015.04.009

Gerhard Tutz initiated the project and developed the imputation approach. Shahla Faisal wrote the code for the method in R (R Core Team, 2017). Shahla Faisal was responsible for the implementation of the method, of the simulation studies and the application to real data. Gerhard Tutz wrote the first version of the paper. Furthermore, Shahla Faisal developed the corresponding R package `wNNSel` (available on CRAN). The technical report (Tutz and Ramzan, 2014) contains first versions of work on the project. Preliminary work on the topics of Chapter 3 has been presented at IWSM 2014 (Ramzan et al., 2014) as a talk

- Ramzan, Tutz, and Heumann (2014). Improved methods for nearest neighbor imputation. 29th International Workshop on Statistical Modelling. Georg-August-Universität Göttingen, Germany.

Chapter 4 is based on the published paper

- Faisal and Tutz (2017c). Missing value imputation for gene expression data by tailored nearest neighbors. Statistical Applications in Genetics and Molecular Biology 16 (2), 95–106. DOI 10.1515/sagmb-2015-0098.

This project was initiated jointly by Shahla Faisal and Gerhard Tutz. Shahla Faisal developed the R code (R Core Team, 2017) as well as implemented and conducted the numerical experiments and the data analyses. She mainly wrote the paper and Gerhard Tutz added valuable remarks and complementary notes which improved the manuscript. Apart from some minor modifications in the notations, Chapter 4 and Faisal and Tutz (2017c) match.

Chapter 5 is based on a paper available on arXiv

- Faisal and Tutz (2017e). Nearest Neighbor Imputation for Categorical Data by Weighting of Attributes. `arXiv:1710.01011 [stat.ME]`

The project was initiated by Shahla Faisal and further developed jointly by the two authors. Shahla Faisal wrote the R code (R Core Team, 2017) as well as implemented the method and conducted the simulations and applications on real data. She mainly wrote the manuscript in close collaboration with Gerhard Tutz. Preliminary work on Chapter 5 has been presented at IWSM-2016 (Ramzan and Tutz, 2016) as a talk.

- Ramzan and Tutz (2016). Nearest neighbor imputation for categorical data by weighting of attributes. 31st International Workshop on Statistical Modelling. Rennes, France.

Chapter 6 The project was jointly developed by the two authors. Shahla Faisal was responsible for writing the code in R (R Core Team, 2017). Shahla Faisal implemented and conducted the numerical experiments and the data analyses. She also wrote the first version of the paper. Preliminary results linked to the topic of Chapter 5 have been presented at conference HDDA-IV (Faisal and Tutz, 2017a) as a talk

- Faisal and Tutz (2017a) Imputation for Missing Values in High-Dimensional Data Structures. International Workshop on Perspectives on High Dimensional Data (HDDA) VII. Guanajuato, Mexico.

and at the conference IWSM-2017 as a poster,

- Faisal and Tutz (2017b). Imputation in High-dimensional Mixed-Type data by Nearest Neighbors. 29th International Workshop on Statistical Modelling. University of Groningen, Netherlands.

The conference paper (Faisal and Tutz, 2017b) contains the preliminary work on the project. Shahla Faisal presented the poster and won the *best poster presentation* award in the conference (Faisal and Tutz, 2017b).

Chapter 7 The idea was initiated by both coauthors. Christian Heumann helped to develop the algorithm. The manuscript was mainly written by Shahla Faisal. Christian Heumann and Gerhard Tutz added their valuable comments to improve the manuscript. Currently, the manuscript is submitted for publication.

- Faisal, S. and Heumann, C. (2017). Bootstrap Inference for Weighted Nearest Neighbors Imputation.

Preliminary results linked to the topic of Chapter 7 have been presented in the conferences in the talks

- Ramzan, Heumann, and Tutz (2016a). Bootstrap Confidence Interval after Nearest Neighbors Imputation. 14th International Conference on Statistical Sciences: Statistics for Better Decision-Making and Development, Jinnah Sindh Medical University, Karachi, Pakistan. March 14-16, 2016.
- Ramzan, Heumann, and Tutz (2016b). Inference when using Nearest Neighbors methods and the Bootstrap. 22nd International Conference on Computational Statistics, Auditorium/Congress Palace Principe Felipe, Oviedo, Spain.

Chapter 8 The idea was initiated by all the coauthors. Shahla Faisal mainly wrote the manuscript. She was also responsible for conducting the simulations in `R` (R Core Team, 2017). Gerhard Tutz added valuable remarks which helped to improve the manuscript.

Chapter 9 The project was initiated and developed in close collaboration. Shahla Faisal developed the R code for the methods. She, as the first author, mainly wrote most of the manuscript and performed the presented analyses. Gerhard Tutz helped to improve the manuscript by extensive discussions. Shahla Faisal implemented the method and conducted the simulations and applications on real data. Preliminary results linked to the topic of Chapter 9 have been presented in the conference as a talk

- Faisal and Tutz (2017d). Multiple Imputation using Sequential Nearest Neighbors. The Third Annual Kliakhandler Conference on *Bayesian Inference in Statistics and Statistical Genetics*. Michigan Technological University, USA. August 16-20, 2017.

Software

Most computations in this thesis were done with the statistical program `R` (R Core Team, 2017). For most of the methods proposed in this thesis add-on packages for `R` were developed and some package are being developed. In particular, the following `R`-package was developed which is available on CRAN (R Core Team, 2017):

`wNNSel` provides the methods proposed in Chapter 3 and Chapter 4. (Faisal, 2017)

2. Methodological Concepts for Missing Data

In this chapter, we briefly describe missing data, their impact on downstream analysis and issues in high-dimensional data. Section 2.1 presents the basic concepts related to missing values, and the mechanisms of missing values. A brief overview of the available techniques to handle missing data is provided in Section 2.2. Section 2.3 is aimed at a brief history of the nearest neighbors methods, the classic nearest neighbors algorithm is described in Section 2.2. The next sections are dedicated to the proposed approaches regarding single imputation (Section 2.4-2.7). The remaining sections describes the concepts and algorithms related to inference based on imputed data and the multiple imputation of missing values.

2.1. Missing Values

Missing data are often a major problem in all areas of quantitative research. The term *missing* or *incomplete* refers to unavailability of an information on some characteristics of the data. Missing values values may occur due to many reasons, for example, patients may fail to respond to certain question. The subjects may withdraw or expire before completion of the studies. The respondents may not provide the information at all, for example income, age etc. It is also possible that some respondents do not provide the complete information on the queries, which is the most common reason for missing values in surveys. Sometimes the information may not be recorded or included into the database due to failure of recording mechanisms. Whatever the reason, missing values or incomplete data occur in all areas of applied research.

The major issue with missing data is that almost all standard (classic and modern) statistical techniques for analyzing the data require complete cases without any missing observation. The commonly used statistical packages are also set to the default options for dealing missing data, i.e., to discard the incomplete cases before preforming the actual analysis, despite that the case may have potential information to contribute to the overall analysis.

In real data applications, if the data contains fifteen percent or more missing values, it can greatly affect the results, some proper technique/procedure is required to handle five to fifteen percent of missing values. However, one to five percent missing values are considered to be manageable and less than one percent are usually trivial (Edgar Acuna and Rodriguez, 2004, Jiang and Yang, 2015).

Now-a-days, the collection of high-dimensional data is no more a tedious task due to the advancement and significantly rapid growth in technology. Therefore it is not uncommon to have a large number of variables measured on a few number of samples in a dataset, for example, in biomedical, epidemiological and social research. It becomes more challenging for the analysts to deal with missing values if they occur in high-dimensional data. An algorithm based on the popular random forests technique was proposed by Stekhoven and Bühlmann (2012). It is able to impute missing data in high-dimensional settings. Random forests avoid overfitting by bootstrap aggregation of multiple regression trees. Furthermore, the accuracy of predictions is enhanced by combining the predictions across all the trees (Breiman, 2001). An adaptation of this method and its comparison to parametric imputation methods was given by Shah et al. (2014). The results of their studies showed that their proposed method is more efficient and may produce confidence interval that are narrower than standard MI approaches. Four different variants of the nearest neighbors imputation were developed by Liao et al. (2014). But all of these methods do not properly account for the uncertainty of imputation, Deng et al. (2016) regarded them as *improper* in the sense of Rubin (1987).

To deal with high-dimensional data, some model based methods are also available in the literature apart from kNN and random forests. But, in general, these MI methods suffer from the curse of dimensionality and hence may not be suitable for imputing the high-dimensional data. When the number of predictors is less than the sample size, many software packages provide an easy application of MI methods, e.g., `R` packages `mice` and `Amelia`. When the number of predictors is less than but close to the sample size, they can be applied to get imputation results but may not perform well (Zhao and Long, 2016). But when the number of predictors exceeds the sample size, the available software packages crash or fail to respond. MI methods, particularly, can have problems when $n < p$, since imputation model may not be defined in this case. Some researchers have suggested to use regularized regression methods, which allow for variable selection before building the final imputation model (Long and Johnson, 2015).

2.1.1. Missing Data Mechanism

An important aspect in missing data imputation is the pattern of missing values because it determines the selection of an imputation procedure (Little and Rubin, 2002; Allison, 2001). Most imputation methods assume the data to be at least MAR, if not MCAR, and so does the NN method. Little and Rubin (2002) defined three categories of missing data, missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR). If any of MCAR or MAR assumptions hold, the missing data mechanism is said to be *ignorable*. The missingness is known to be *non-ignorable* mechanism if the data has NMAR missing values. In the following we briefly describe these mechanism with explanatory example.

Missing Completely At Random (MCAR)

Missing completely at random: missing pattern results from a completely random process and the probability of missingness is same for all units.

The missing data mechanism is MCAR, when the probability of a sample having a missing value for a covariate depends neither on the covariate having missing data nor on the other covariates in the data matrix (Rubin, 1987). In other words, the probability of missing data is completely independent of the data. The probability of missingness $P(O_{ij} = 0)$ is not related to any of the covariates, i.e.

$$P(O_{ij} = 0)|\boldsymbol{X}_{obs}, \boldsymbol{X}_{mis}) = P(O_{ij} = 0)$$

where $\boldsymbol{X}_{obs}$ and $\boldsymbol{X}_{mis}$ denote the observed and missing parts of the data matrix $\boldsymbol{X}$.

The distribution of the X_{ij} is the same as that of the $X_{ij,obs}$ given X_i. Thus the observed data can be considered a random sample from the hypothetical complete data. Examples include, death due to a car accident, the lab technician accidentally destroyed the blood sample, It has been argued that MCAR is a very strict assumption as it requires the missingness to be completely unrelated to all the study variables (Raghunathan, 2004) . In practice, this assumption is likely to be violated, despite of the fact that it may hold in certain situations.

Missing At Random (MAR)

The mechanism is MAR if the probability that a variable is missing depends only on the available information and not on the observations that are missing in the data (Rubin, 1987). For example, females are less likely to answer age question (with gender completely observed), or the weight of individuals measured with high blood pressure only. The $P(O_{ij} = 0))$ depends on the observed values , i.e.

$$P(O_{ij} = 0)|\boldsymbol{X}_{obs}, \boldsymbol{X}_{mis}) = P(O_{ij} = 0|\boldsymbol{X}_{obs})$$

The MAR assumption is not as strong as MCAR. It is also to note that MCAR assumption is testable (Little, 1988) but MAR is usually not testable.

Not Missing At Random (NMAR)

When the probability of a missing data value depends on the missing data itself, the mechanism is said to be *Not Missing At Random* (NMAR) or *non-ignorable* (Little and Rubin, 2002; Schafer, 1997).

The $P(O_{ij} = 0)$ depends on the observed as well as on the unobserved values. For example, heavy people hesitate to give their weight; respondents with high income less likely to report

income; and in a study on pain relief, patients with severe pain are less likely to answer the phone and give their current pain status.

2.2. An Overview of Traditional Missing Data Techniques

Missing data have always been a challenge to researchers. Since the 1980s, many techniques to impute missing data have been proposed, for example, Little and Rubin (2002) and Schafer (2010). In the following we briefly describe the some popular approaches that are available in literature to deal with the missing data.

2.2.1. Deletion Methods

The simplest and most basic approach to address the problem of missing values include deletion methods. The listwise deletion (complete case analysis, case-wise deletion) method is to simply remove the missing observations by discarding the corresponding cases. However, this procedure entails loss of valuable information and can lead to a selection bias, in particular if the number of missing values is large (Tuikkala et al., 2008). Moreover, the anaysis of such data will produce biased estimates when the missing data violates the MCAR mechanism assumption. The situation calls for methods to impute the missing values before using standard statistical analysis tools.

2.2.2. Substitution Methods

The simplest approach consists of replacing the missing values with zeros or mean/average values (Alizadeh et al., 2000). This approach is more popular to impute missing values in gene expression or similar data sets. A distinct disadvantage of this method is that the information on the correlation among predictors/genes is ignored. The estimated parameters and their standard errors are biased due to the reason that variance of the variable being imputed reduces since it shifts the possible extreme values towards the center of the distribution. It has been demonstrated that imputation methods perform better than both the simple deletion of whole observations and the replacement of the missing data by zeros (ZeroImpute) or average values (MeanImpute), see, for example, Troyanskaya et al. (2001).

2.2.3. Imputation

Imputation refers to a procedure of filling the missing values with plausible values to get complete data. A number of methods to estimate or impute missing values have been proposed in the literature, and a continuous effort is in progress to develop improved imputation methods. An imputation may be deterministic or stochastic (random). For given

data, the deterministic imputation always produces the same value as an estimate whereas stochastic method may produce different results.

In fact all the missing data handling techniques replace the missing values by some kind of estimated values. How close these estimated values are to the true ones, tremendously depends upon the algorithm used to get the final outcome. The imputation of missing data usually yields better results as compared to the approached mentioned previously. The reason is that one gets a complete data set after imputation which represents the true characteristics of the data to some extent.

Broadly speaking, the imputation methods for filling in an incomplete data matrix can be divided into two main categories, single imputation and multiple imputation (Little and Rubin, 2002).

Single Imputation

The missing data is retrieved by replacing each missing value by one *plausible* value. The imputed values are assumed to be the real values that would have been observed at the time of data collection and the method does not address the uncertainty of the imputation process. Some examples include hot-deck, mean substitution, regression imputation, nearest neighbors imputation etc.

Multiple Imputation

One of the most popular methods for handling missing values is multiple imputation (Rubin, 1987). It is a technique to get more than one *plausible* values for each missing value in data. This technique provides valid results even when the missing mechanism is missing at random (MAR) (He and Belin, 2014). Multiple imputation (MI) takes in the uncertainty factor inherent in the imputations by providing more than one imputed values corresponding to each missing value in the data. As a result, we have more than one imputed datasets associated with the missing data. The standard procedures for complete data analyses are then applied in the usual way to each of the imputed data set produced by the multiple imputation. To get an overall inference, the results obtained by analyzing each of the imputed data set are combined across all the imputed data sets using the well known Rubin's rule (Little and Rubin, 2014; Rubin, 2004). The missing data uncertainty, which is ignored in single imputation, can be included using multiple imputation. In the recent years, MI has emerged as a leading approach for imputing the missing data with lot of advancements in the methodology and software (Harrel and Zhou, 2007; Horton and Kleinman, 2007).

2.3. Nearest Neighbors Methods

Missing data has always been a challenging area for the researchers. Usually the softwares ignore all the missing cases while doing analyses. But this is not an advisable way out for dealing with the missing values. Many techniques have been suggested in literature to fill these missing data, nearest neighbor imputation is one of them (Troyanskaya et al. (2001)). This technique is based on the average of k *nearest neighbors* based on some distance measure from the available data. A well-known and computationally simple method for the imputation of missing data is mean substitution. However, its disadvantage is that the correlation structure among the predictors is ignored. An alternative is the *nearest neighbors* (NNs) approach, which uses observations in the neighborhood to impute missing values. NNs as a nonparametric concept in discrimination dates back to Fix and Hodges (1951). The approach has been successfully used to impute data in gene expression (Troyanskaya et al., 2001; Atkeson et al., 1997; Hastie et al., 1999), machine learning (Batista and Monard, 2002), medicine (Waljee et al., 2013), forestry (Eskelson et al., 2009; Hudak et al., 2008), and compositional data (Hron et al., 2010).

NN imputation has earned its popularity due to its simplicity and ease of implementation, and still yields fairly better results than other methods. The reason is that it estimates a missing value by utilizing the information of other samples/instances/neighbors that are *closest* or *nearest* to the sample/instance being imputed.

Before considering NN approaches in detail, a few general remarks on the method are warranted. The main approach for imputation is multiple imputation because it yields valid inference under several basic conditions. In contrast to NN approaches, multiple imputation provides consistent estimates under MAR conditions. However, NN approaches are useful in high-dimensional problems in which multiple imputation cannot be applied and complete cases do not generate enough data.

Strengths and Weaknesses of NN

This approach has many appealing benefits along with some weaknesses.

+ It is a nonparametric approach, and does not require any particular assumptions or model specification to relate the response to covariates, and is therefore, less likely to produce model misspecification.

+ NN method make use of auxiliary information obtained from other values in the same dataset and thus preserves the original structure of the data.

+ Both qualitative and quantitative data can be utilized to find an estimated value.

+ Since this method does not require building a predictive model to get imputation estimates, it evades the computational cost and time consumption on modeling.

– Since it uses a distance function to compute the similarity, the missing attributes do not contribute to the distance calculations.

– Since to find the k closest neighbors, the method requires to compute the distance for all the instances, it can be time consuming for the data with higher number of samples.

Literature review NN

Although a deterministic method, Chen and Shao (2000) proved that the nearest neighbors approach can estimate the distributions correctly. Most imputation methods assume the data to be at least MAR, if not MCAR, and so does the NN method.

Several NN-based approaches to imputation have been considered in the literature. A comparison of k nearest neighbor (KNN) imputation with mean imputation and singular value decomposition (SVD) techniques for filling missing values in gene expression data, is given in Troyanskaya et al. (2001). The results of their simulation studies showed that KNN method performs well as compared to mean imputation and singular value decomposition (SVD) approaches. Hastie et al. (1999) presented the idea of using Euclidean distance for selecting k nearest neighbors. Hron et al. (2010) used imputation techniques for the compositional data. Batista and Monard (2002) used k nearest neighbor method to impute missing data in Machine Learning and compared with the existing methods C4.5 and CN2. Malarvizhi and Thanamani (2012) compared single imputation methods and found that median or standard deviation substitution perform better than mean substitution. A comprehensive overview of the imputation methods used in forestry can be found in Eskelson et al. (2009). Hudak et al. (2008) compared the methods using Forestry data on the basis of root mean square difference and found the *random forest* as best. In a comparative study of single imputation methods, Malarvizhi and Thanamani (2012) found that median or standard deviation substitution perform better than mean substitution. A further comparison of KNN with mean, ordinary least squares (OLS) and partial least squares (PLS) imputation methods in microarray data was that of Nguyen et al. (2004). In that study, the good performance of KNN imputation was demonstrated.

Many alternative procedures have been proposed that rely on basic concepts to impute values by building averages over qualifying neighbors, as described in Ouyang et al. (2004), Kim et al. (2004), Sehgal et al. (2005), and Scheel et al. (2005). Liew et al. (2011) and Moorthy et al. (2014) reviewed the available methods and algorithms, with a focus on gene expression data. Johansson and Hakkinen (2006) proposed *WeNNI*, which utilizes continuous weights in the NN imputation procedure. Bø et al. (2004) and Wasito and Mirkin (2005) proposed a NN procedure with a modification based on the least squares method. Other variants include the local least squares (LLSimpute) method of Kim et al. (2005), the sequential local least squares (SLLSimpute) method of Zhang et al. (2008), and the iterative LLS (ILLSimpute) of Cai et al. (2006).

2.3.1. Traditional k Nearest Neighbors Imputation

When using NNs to impute data, two preliminary steps must be carried out. First, one has to define the NNs, that is, how they are computed and which distance measures are to be used. Second, one has to determine how these NNs are used to obtain an imputed value.

Distances and Computation of Nearest Neighbors

Let n observations on p covariates be collected. The corresponding $n \times p$ data matrix is given by $\mathbf{X} = (x_{is})$, where x_{is} denotes the ith observation of the sth variable. Let $\mathbf{O} = (o_{is})$ denote the corresponding $n \times p$ matrix of dummies with entries

$$o_{is} = \begin{cases} 1 & \text{if } x_{is} \text{ was observed} \\ 0 & \text{for missing value.} \end{cases}$$

For metrically scaled data, distances between two observations $\mathbf{x}_i$ and $\mathbf{x}_j$, represented by rows in the data matrix, can be computed by using the L_q metric for the observed data, given by

$$d_q(\mathbf{x}_i, \mathbf{x}_j) = [\frac{1}{a_{ij}} \sum_{s=1}^{p} |x_{is} - x_{js}|^q I(o_{is} = 1) I(o_{js} = 1)]^{1/q}, \tag{2.1}$$

where $a_{ij} = \sum_{s=1}^{p} I(o_{is} = 1) I(o_{js} = 1)$ denotes the number of valid components in the computation of distances. The indicator function $I(a)$, which is used in the definition, has the value 1, if a is true and 0 otherwise. The computation of distances does not use all the components of the vectors but only those for which observations in both vectors are available. The components included in the computation of neighbors are given by $C_{ij} = \{s : I(o_{is} = 1) I(o_{js} = 1) = 1\}$. The distances define the NNs when imputing a specific value. It should be noted that the number of components varies over the sample because C_{ij} depends on the specific observation for which imputations are to be made.

Similar concepts to define distances and therefore NNs were described by Hastie et al. (1999), Troyanskaya et al. (2001), Myrtveit et al. (2001), and Kim et al. (2005). In those studies the Euclidean distance was $q = 2$. Alternatives to compute distances in gene expression studies are based on the Pearson correlation (Dudoit et al., 2002, Bø et al., 2004) and on covariance estimates (Sehgal et al., 2005).

Imputation for Fixed Number of Neighbors

Consider the imputation for $\mathbf{x}_i$ in component s, that is, $o_{is} = 0$. The imputation estimate for $\mathbf{x}_i$ is based on the k NNs in the reduced data set. The k NNs are determined from the corresponding $(\tilde{n} \times p)$-dimensionally reduced data set $\tilde{\mathbf{X}} = (x_{ij}, o_{is} = 1)$ to obtain

$$\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(k)} \quad \text{with} \quad d(\mathbf{x}_i, \mathbf{x}_{(1)}) \leq \cdots \leq d(\mathbf{x}_i, \mathbf{x}_{(k)}),$$

where $\mathbf{x}_{(j)}^T = (x_{(j)1}, \ldots, x_{(j)p})$ denotes the jth NNs. The *imputation value for a fixed* k is then

$$\hat{x}_{is} = \frac{1}{k} \sum_{j=1}^{k} x_{(j)s}. \tag{2.2}$$

Thus, the missing value in the sth component of observation vector $\mathbf{x}_i$ is replaced by the average of the corresponding values of the k NNs. The accuracy of the method is mainly determined by the number of neighbors that are used. Simple rules use a fixed number of neighbors that is chosen by the experimenter. However, it is advantageous to consider the number of neighbors as a tuning parameter that is chosen in a data-driven manner, for example, by cross-validation.

2.4. Modification of the Traditional kNN Imputation

A disadvantage of the imputation based on the k NNs is that the value of the first NN has the same importance as the kth NN. A more appropriate method uses weights that account for the distance of the observations. We consider a weighted average of neighbors based on distances that are determined by kernel functions (Chapter 3). The *weighted imputation estimate* considered here has the form

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\mathbf{x}_i, \mathbf{x}_{(j)}) x_{(j)s} \tag{2.3}$$

with weights

$$w(\mathbf{x}_i, \mathbf{x}_{(j)}) = \frac{K(d(\mathbf{x}_i, \mathbf{x}_{(j)})/\lambda)}{\sum_{l=1}^{k} K(d(\mathbf{x}_i, \mathbf{x}_{(l)})/\lambda)}, \tag{2.4}$$

where $K(.)$ is a kernel function and λ is a tuning parameter. For small λ the weights decrease very strongly with distance, whereas for $\lambda \to \infty$ all neighbors are of equal weight. To make λ the crucial tuning parameter, a large number of potential neighbors should be used. In the extreme case, if $k = \tilde{n}$, the window-width λ will be the only tuning parameter.

Alternative weights were described by Troyanskaya et al. (2001). They used the weighted average over the k NNs based on the Euclidean distance, with the weights determined by the inverse of that distance.

2.5. Nearest Neighbors for High-Dimensional Data

In high-dimensional settings, the computation of distances over the whole feature space tends to suffer from the curse of dimensionality. The strategy that is proposed here is to select the features that actually carry information on the missing values. It has already been shown by Troyanskaya et al. (2001) that the correlation between gene expression profiles plays a major role in the imputation of missing values. Variables can contribute to the accuracy of the estimated value only if they are correlated to the variable that is to be imputed. If there is no correlation between these two components, the variable contains no information and the nearest neighbor imputation will not perform better than simple mean imputation.

The proposed `wNNSelect` algorithm explicitly links the selection of predictors or features used in the imputation method to the correlation between the variable that is used to impute and the target variable, that is, the variable that is missing and is to be imputed. The link is obtained by replacing the simple distance function by a weighted distance, where the weights are determined through the strength of correlation. More precisely, when imputing a value x_{is}, the `wNNSelect` algorithm uses the distance function

$$d_{q,C}(\mathbf{x}_i, \mathbf{x}_j) = \left(\frac{1}{a_{ij}} \sum_{l=1}^{p} |x_{il} - x_{jl}|^q I(o_{is} = 1) I(o_{js} = 1) C(r_{sl}) \right)^{1/q}, \tag{2.5}$$

where r_{sl} is the empirical correlation between covariates s, l and $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the correlations into weights. The transformation is constructed such that only the covariates that are strongly correlated with component s strongly contribute to the computation of distances, while components that are not correlated do not contribute to the distance. Therefore, if the lth variable is uncorrelated with the target variable s the variable is not used when computing distances. If the correlation is strong, no matter positive or negative, the variable contributes to the computation of the distance. A convex function $C(.)$ that can be used is defined by

$$C(r) = \begin{cases} \frac{|r|}{1-c} - \frac{c}{1-c} & \text{if } |r| > c \\ 0 & \text{if } |r| \leq c. \end{cases} \tag{2.6}$$

It is linear in the absolute value of the correlation. If $|r_{sj}| \leq c$, the component s does not contribute to the distance. If $|r_{sj}| > c$, then the weights increase linearly with increasing correlation and have a weight of 1 if $|r| = 1$. The threshold parameter c can be chosen as fixed, for example by $c = 0$, or can be considered as an additional tuning parameter from the range $[0, 1)$. A smoother function, which can also be used, is the power function

$$C(r) = |r|^m \tag{2.7}$$

with natural number m as an additional tuning parameter. However, a problem with this weighting scheme is that r_{sl} cannot be computed, since missing values are present in the data. Therefore, we use a simple first-step imputation with five un-weighted NNs and compute the correlations from the observed and imputed data. The actual imputation is then carried out by using the correlations computed in the first step. It should be noted that, depending on the scale level, alternative dependence measures might be more appropriate. For variables that have been measured on an ordinal scale, Spearman's correlation coefficient can be used. For categorical variables, a standardized chi-squared measure, such as Cramer's V-statistic, is an option.

The weighted L_q distances with the selection of dimensions includes the calculation of an additional tuning parameter. Thus, in addition to the tuning parameters for the kernels, λ, an appropriate value c for the convex function (3.8) or an integer m for the convex function (3.9) must be found. The parameters are chosen in the same way as λ, namely, by cross-validation (see Section 3.2.3 in chapter 3). For the simultaneous selection of the tuning parameters (λ, c) or (λ, m) cross-validation is computed on a two-dimensional grid of values (see algorithm 4.1 in chapter 4).

2.5.1. Practical Issues in High-Dimensional Data ($n << p$)

DNA microarray data are important in a wide range of application areas such as biology, genetics and medicine. They typically come in the form of large matrices with missing values affecting most of the genes. Missing values can affect up to 90% of the genes (Ouyang et al., 2004). The main causes for the occurrence of missing values are artifacts on the microarray, dust or scratches on the slide, hybridization failures, image noise and corruption, and insufficient resolution. Additionally, the robotic methods used to create data can be responsible (Chapter 4).

Due to the presence of missing values, the subsequent analysis and results based on these incomplete data can suffer strongly from the presence of these missing values. Several approaches to imputation of missing values in gene expression data have been developed but the task is difficult due to the high-dimensionality (number of genes) of the data. Instead of using nearest neighbors defined by a distance that includes all genes the distance is computed for genes that are apt to contribute to the accuracy of imputed values. The method aims at avoiding the curse of dimensionality, which typically occurs if local methods as nearest neighbors are applied in high-dimensional settings. The proposed weighted nearest neighbors algorithm is compared to existing missing value imputation techniques like mean imputation, KNNimpute and the recently proposed imputation by random forests (Chapter 4). We use RNA-sequence and microarray data from studies on human cancer to compare the performance of the methods. The results from simulations as well as real studies show that the weighted distance procedure can successfully handle missing values for high-dimensional data structures where the number of predictors is larger than the number of samples. The method typically outperforms the considered competitors.

2.6. Extensions to Binary and Multi-Categorical Data

Categorical data are important in many fields of research, examples are surveys with multiple-choice questions in the social sciences (Chen and Shao, 2000), single nucleotide polymorphisms (SNPs) in genetic association studies (Schwender, 2012) and tumor or cancer studies (Eisemann et al., 2011). It is most likely that some respondents/patients do not provide the complete information on the queries, which is the most common reason for missing values. Sometimes, also the information may not be recorded or included into the database. Whatever the reason, missing data occur in all areas of applied research. Since for many statistical analyses a complete data set is required, the imputation of missing values is a useful tool. For categorical data, although prone to contain missing values, imputation tools are scarce.

The simulation studies of Ezzati-Rice et al. (1995) and Schafer (1997) showed that it provides an attractive solution for missing categorical data problems. However, its use is restricted to cases with a small number of attributes (Erosheva et al., 2002) since model selection and fitting becomes very challenging for larger dimensions.

A non-parametric method called hot-deck imputation has been proposed as an alternative (Rubin, 1987). This technique searches for the complete cases having the same values on the observed variables as the case with missing values. The imputed values are drawn from the empirical distribution defined by the former. The method is well suited even for data sets with a large number of attributes (Cranmer and Gill, 2013). A variant, called approximate Baysian bootstrap, works well in situations where the standard hot-deck fails to provide proper imputation Rubin and Schenker (1986). But the hot-deck imputation may yield biased results irrespective of the missing data mechanism (Schafer and Graham, 2002), and it may become less likely to find matches if the number of variables is large (Andridge and Little, 2010).

Another popular non-parametric approach to impute missing values is the nearest neighbors method (Troyanskaya et al., 2001). The relationship among attributes is taken into account when computing the degree of nearness or distance. The method may easily be implemented for high-dimensional data. However, the k-nearest neighbors (kNN) method, originally developed for continuous data, cannot be employed without modifications to non-metric data such as nominal or ordinal categorical data (Schwender, 2012). As the accuracy of the kNN method is mainly determined by the distance measure used to calculate the degree of nearness of the observations, one needs different distance formula when data are categorical. Some existing methods to impute attributes are based on the mode or weighted mode of nearest neighbors (Liao et al., 2014).

Schwender (2012) suggested a weighted kNN method to impute *categorical* variables only, that uses the Cohen or Manhattan distance for finding the nearest neighbors. The imputed value is calculated by using weights that correspond to the inverse of the distance. One limitation of this approach is that it can handle only variables that have the same number of categories. Also the value of k, which strongly affects the imputation estimates, is needed. There are some methods for imputing mixed data that can also be used for categorical data,

for example, see Liao et al. (2014), Stekhoven and Bühlmann (2012). The latter transform the categorical data to dichotomous data and use the classical k-nearest neighbors method to the standardized data with mean 0 and variance 1. The imputed data are re-transformed to obtain the estimates. However, it has been confirmed by several studies that rounding may lead to serious bias, particularly in regression analysis (Allison (2005), Horton et al. (2003)).

For categorical data one has to use specific distances or similarity measures, which are typically based on contingency tables. Commonly used distance measures include the simple matching coefficient, Cohen's kappa κ_c (Cohen, 1960)), and the Manhattan or L_1 distance. The Euclidean or variants of the Minkowski distance give an equal importance to all the variables in the data matrix when computing the distance. But for a larger number of variables, the equal weighting ignores the complex structure of correlation/association among these variables. As will be demonstrated in Chapter 5, better distance measures are obtained by utilizing the association between variables. More specific, we propose a weighted distance that explicitly takes the association among covariates into account. Strongly associated covariates are given higher weights forcing them to contribute more strongly to the computation of the distances than weakly associated covariates.

2.6.1. Selection of Attributes by Weighted Distances

The Euclidean or variants of Minkowski distance give an equal importance to all the variables in the data matrix. When the number of variables is large and they are correlated/associated, it is useful to give unequal weights to covariates when calculating the distance. We present a weighted distance which explicitly takes the association among covariates into account. More specifically, highly associated covariates will contribute more to the computation of the distance than less associated covariates.

For a concise definition we distinguish between cases that were observed in the corresponding component and missing values, only the former contribute to the computation of the distance. Let us consider the data matrix $\mathbf{Z}$ with dimension $n \times p$. Let $\mathbf{O} = (o_{is})$ denote the $n \times p$ matrix of dummies, with $o_{is} = 0$ if the value is missing, and $o_{is} = 1$ if the the value is available in the data matrix $\mathbf{Z}$. Let $\mathbf{z}_i^T = (Z_{i1}, \cdots, Z_{ip})$ denote the ith row or observation vector in the data matrix $\mathbf{Z}$. The categorical observations Z_{is} in the data matrix $\mathbf{Z}$ can take values from $\{1, \ldots, k_s\}, s = 1, \ldots, p$, where k_s is the number of categories of the s^{th} attribute.

For the computation the categorical variable Z_{is} is transformed into binary variables. Let $\mathbf{z_{is}}^T = (z_{is1}, \ldots, z_{isk_s})$ be the dummy vector built from Z_{is} with components being defined by

$$z_{isc} = \begin{cases} 1 & \text{if } Z_{is} = c, \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mathbf{Z}^D$ denote the matrix of dummies which is obtained from the original data matrix. Thus, the ith row of the matrix $\mathbf{Z^D}$ has the form $(\mathbf{z}_{i1}^T, \cdots, \mathbf{z}_{ip}^T)^T$ with dummy vectors $\mathbf{z}_{is}$,

$s = 1, \cdots, p$. The dummy vectors $\mathbf{z_{is}}^T$ for a nominal variable with four categories can be written as

category	z_{is1}	z_{is2}	z_{is3}	z_{is4}
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Let now Z_{is} be a specific missing entry in the data matrix $\mathbf{Z}$, that is, $o_{is} = 0$. For the computation of distances we use the corresponding matrix of dummy variables $\mathbf{Z}^D$.

We propose to use as distance between the i-th and the j-th observation

$$d_{Cat}(\mathbf{z}_i, \mathbf{z}_j) = \left(\frac{1}{a_{ij}} \sum_{l=1}^{p} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|^q I(o_{il} = 1) I(o_{jl} = 1) C(\delta_{sl}) \right)^{1/q}, \tag{2.8}$$

where $I(.)$ denotes the indicator function and $a_{ij} = \sum_{l=1}^{p} I(o_{il} = 1) I(o_{jl} = 1)$ is the number of valid components in the computation of distances. The crucial part in the definition of the distance is the weight $C(\delta_{sl})$. $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the measure of association between attributes s and l, denoted by δ_{sl}, into weights.

It is worth noting that the distance is now specific to the s^{th} attribute, which is to be imputed. For $C(.)$, we use the power function $C(\delta_{sl}) = |\delta_{sl}|^{\omega}$ (equation (2.7)). So the attributes that have a higher association with the s^{th} attribute are contributing more to the distance and vice versa. The higher the value of association, the more it contributes to the computation of the distance. Note also that only the *available* pairs with $I(o_{il} = 1) I(o_{jl} = 1)$ are used for the computation of the distance.

2.6.2. Using Nearest Neighbors to Impute Missing Values

Classical nearest neighbor approaches fix the number of neighbors that are used. We prefer to use weighted nearest neighbors by using weights that are defined by kernel functions. Uniform kernels yield the classical approach, however, smooth kernels typically provide better results. Let Z_{is} be a missing value in the $n \times p$ matrix of observations. The k nearest neighbor observation vectors are defined by $\mathbf{z}^D_{(1)}, \ldots, \mathbf{z}^D_{(k)}$ with $d(\mathbf{z}_i, \mathbf{z}_{(1)}) \leq \cdots \leq d(\mathbf{z}_i, \mathbf{z}_{(k)})$, where $\mathbf{z}^D_{(i)}$ are rows from the matrix $\mathbf{Z}^D$, and $d(\mathbf{z}_i, \mathbf{z}_{(k)})$ is the computed distance using equation (2.8). It is important to mention that the row $\mathbf{z}^D_{(i)}$ is composed of values of dummy variables of the form $(\mathbf{z}^T_{(i)1}, \cdots, \mathbf{z}^T_{(i)p})^T$, where $\mathbf{z}^T_{(i)s} = (z_{is1}, \cdots, z_{isk_s})$ are the dummy values. For the imputation of the value Z_{is} we use the weighted estimator

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\mathbf{z}_i, \mathbf{z}_{(j)})\mathbf{z}_{(j)sc}, \tag{2.9}$$

with weights given by

$$w(\mathbf{z}_i, \mathbf{z}_{(j)}) = \frac{K(\frac{d(\mathbf{z}_i, \mathbf{z}_{(j)})}{\lambda})}{\sum_{h=1}^{k} K(\frac{d(\mathbf{z}_i, \mathbf{z}_{(h)})}{\lambda})}, \tag{2.10}$$

where $K(.)$ is a kernel function (triangular, Gaussian etc.) and λ is a tuning parameter. Note that $\hat{\pi}_{is}^T = (\hat{\pi}_{is1}, \cdots, \hat{\pi}_{isk_s})$ is a vector of estimated probabilities.

If one uses all the available neighbors that is $k = n$, then λ is the only and crucial tuning parameter. The imputed estimate of Z_{is} is the value of $c \in \{1, \ldots, k_s\}$ that has the largest value. In other words, the weighted imputation estimate of a categorical missing value Z_{is} is

$$\hat{Z}_{is} = \arg\max_{c=1}^{k_s} \hat{\pi}_{isc}, \tag{2.11}$$

If the maximum is not unique one value is selected at random (Chapter 5).

2.6.3. A Pearson Correlation Strategy

As an alternative we consider a strategy that uses the dummy variables directly. Starting from the matrix of dummies $\mathbf{Z}^D$ we use the Pearson correlation coefficient between dummy variables as association measure. The weighting scheme remains the same. The imputation is again determined by

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\mathbf{z}_i, \mathbf{z}_{(j)})\mathbf{z}_{(j)sc}.$$

Although $\hat{\pi}_{isc}^T = (\hat{\pi}_{is1}, \ldots, \hat{\pi}_{isk_s})$ might not be a vector of probabilities, simple standardization by setting

$$\tilde{\pi}_{isc} = \hat{\pi}_{isc} / \sum_{r=1}^{k_s} \hat{\pi}_{isr}$$

yields a vector of probabilities that can be used to determine the mode. The method can be seen as an adaptation of the weighting method proposed by Tutz and Ramzan (2015). Only small modification are needed to apply the method to the dummy variables. One is that the missing of an observation refers to a set of variables, namely all the dummies that are linked to a missing value. For this method the Gaussian kernel and the Euclidean distance are used throughout. The method is denoted by $\texttt{wNNSel}_{\texttt{dum}}$ (Chapter 5, 6).

For multi-categorical data, the simple method which uses dummy variables and the classical correlation coefficient, showed the best performance. For binary data, both procedures $\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ yield similar results, whereas, for multi-categorical data $\texttt{wNNSel}_{\texttt{dum}}$

yields smaller imputation errors. The $\texttt{wNNSel}_{\texttt{dum}}$ method outperforms in simulations as well as in real data application all competitors (Chapter 5).

2.7. Extension to Mixed Type Data

One has to deal with a combination of continuous and nominal variables in many real world applications, therefore the methods to impute mixed data become more important. Since multiple imputation techniques fail for high-dimensional missing data, the nonparametric single imputation methods are gaining more popularity. We propose an improved version of the popular nonparametric nearest neighbors method which uses information only on potentially relevant neighbors (Chapter 6). More specifically, we introduce a distance function that is appropriate for mixed data. It is an extension of Tutz and Ramzan (2015) and uses information on association among variables. A particular advantage of the proposed method is that it simultaneously takes into account the similarities between samples and the relationships between covariates. Furthermore, a Pearson correlation approach based on dummy variables for the imputation of mixed data is also proposed.

2.7.1. Available Distances for Mixed Type Data

A straightforward approach to handle mixed types of variables is to split the variables into types and confine the analysis to the dominant type (Anderberg, 1973). The judgment of *dominant* type may depend on several factors such as the number of variables in each type, background theory, some variable may be of interest to the analyst etc. It is not a recommended practice since it discards the important and relevant information. Moreover, one may be tempted to ignore the differences among different types of variables and use a distance function which is suitable for one type but inappropriate for other types of the variables.

Another simple way to deal with the mixture type of variables is to convert one type of covariate to another, while retaining the maximum possible information, and, then use a distance measure which is suitable for the selected type of covariates. Anderberg (1973, p. 94) argued over the natural question which type should be chosen as the single type for the analysis. For instance, metric variables can be converted into binary variables using a fixed level. Consequently, many observations would be more similar to each other, resulting in the reduction of the influence of metric variables. Alternatively, the nominal variables can be converted into binary variables (see, for example, (Frank and Todeschini, 1994, p. 92)). In general, the conversion of variables may result in a loss of important information contained in the data (Tarsitano and Falcone, 2011).

Some measures for computing the distance between the observations of mixed type data, have been proposed in the literature. When the data contains a mixture of variable types, the distance between observations i and j is calculated as $d_{ij} = w_1 d_{ij}^1 + w_2 d_{ij}^2 + w_3 d_{ij}^3$,

where d^1, d^2, d^3 are the distances calculated separately for the binary, multi-categorical and continuous variables, respectively. The w_1, w_2, w_3 are weighting coefficients in the aggregate distance measure. An important issue with this type of distance is how to determine the weights. One may use an equal or proportional weight for each type of variables or a priori judgment could also be used. The Gower's distance (Gower, 1971) is a weighted average of three different measure of dissimilarity where the weights are the proportions of each type of variable. It is a widely used distance measure based on a general similarity measure and can be used for different types of variables.

2.7.2. Weighted Distance for Mixed Type Data

Let $\boldsymbol{\mathcal{R}} = (R_{is})$ be the $n \times (p+m)$ data matrix with p continuous and m categorical covariates defined by $\boldsymbol{\mathcal{R}} = (\boldsymbol{X}, \boldsymbol{Z})$, where $\boldsymbol{X} = (x_{ig})$, $g = 1, \cdots, p$, with x_{ig} denoting the i^{th} observation of the g^{th} continuous covariate, and $\boldsymbol{Z} = (z_{il})$, $l = 1, \cdots, m$, with z_{il} denoting the i^{th} observation of the l^{th} categorical covariate. Let $\boldsymbol{O} = (o_{is})$ be the corresponding $n \times (p+m)$ matrix with $o_{is} = 1$ if R_{is} is observed, otherwise $o_{is} = 0$.

The categorical observations z_{il} in the data matrix, can take values $c_l \in \{1, \ldots, k_l\}, l = 1, \ldots, m$, where k_l is the number of categories that the l^{th} attribute can take. It is to note that the value of $k_l = 2$ shows that the l^{th} variable is binary whereas $k_l > 2$ shows a multi-categorical variable. Then the i^{th} row of the data matrix $\boldsymbol{\mathcal{R}}$ can be written as $\boldsymbol{R}_i^T = (\boldsymbol{x_i}^T, \mathbf{z_i}^T)$ with $\boldsymbol{x_i}^T = (x_{i1}, \cdots, x_{ip})$ and $\mathbf{z_i}^T = (z_{i1}, \cdots, z_{im})$. For the computation of distances, the categorical variables are transformed into binary variables. Thus the observation z_{il} becomes a vector, $\mathbf{z}_{il}^T = (z_{il1}, \ldots, z_{ilk_s})$ with $z_{ilr} = 1$ if $Z_{il} = r$. The dummy vectors $\mathbf{z}_{il}^T$ for a nominal variable with four categories can be written as

category	z_{il1}	z_{il2}	z_{il3}	z_{il4}
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Thus, the i^th row of the transformed matrix $\boldsymbol{\mathcal{R}}^D$ has the form $[(x_{i1}, \ldots, x_{ip}), (\mathbf{z}_{i1}^T, \cdots, \mathbf{z}_{im}^T)]^T$ with dummy vectors $\mathbf{z}_{il}$, $l = 1, \cdots, m$.

Let R_{is} be a missing entry in the data matrix $\boldsymbol{\mathcal{R}}$, that is $O_{is} = 0$. Then the distance between the i-th and the j-th rows specific for a missing value in the sth covariate is defined by

$$d(\boldsymbol{R}_i, \boldsymbol{R}_j) = \left(\gamma_1 \sum_{g=1}^{p} |x_{ig} - x_{jg}|^q I_{(o_{ig}=1)} I_{(o_{jg}=1)}.C(\delta_{sg}) + \gamma_2 \sum_{l=1}^{m} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|^q I_{(o_{il}=1)} I_{(o_{jl}=1)}.C(\delta_{sl}) \right)^{1/q}, \quad (2.12)$$

The weights γ_1 and γ_2 can be determined on the basis of a *prior* judgment. But if the investigator has no knowledge about which type of covariates are important or to be prioritized, the assessment is not be feasible. A simpler approach is to use an equal weighting for all data types (Chiodi, 1990), that is, to use $\gamma_1 = \gamma_2 = 0.5$. Romesburg (2004) used weights that are proportional to the number of variable in each type. The proportional weighting would be the right choice if all the variables are supposed to be equally important, irrespective of the scale on which they are measured. We propose to consider $\gamma_1,\ \gamma_2$, as tuning parameters and select their value by cross validation. We select tuning parameters on a three-dimensional grid, that is the combination of γ_1, λ and m yielding the smallest imputation error. The procedure for cross validation is given in Chapter 6.

One important issue in the computation of distances in equations (2.8 and 2.12) is how to compute the association among categorical variables as the usual Pearson coefficient of correlation is not suitable for categorical covariates. The measures of association for two nominal or categorical variables are typically based on the χ^2-statistic which tests the independence of variables in contingency tables.

2.7.3. Weighted Imputation by Nearest Neighbors

Using information on all nearest neighbors does not necessarily provide better imputation results and for larger data it would also consume more time. Alternatively, a weighted nearest neighbors method based on kernel function is proposed. More specifically, the weights we are using are defined by

$$w(\boldsymbol{R}_i, \boldsymbol{R}_j) = \frac{K(\frac{d(\boldsymbol{R}_i, \boldsymbol{R}_j)}{\lambda})}{\sum_{l=1}^{k} K(\frac{d(\boldsymbol{R}_i, \boldsymbol{R}_j)}{\lambda})}, \quad (2.13)$$

Let a value is missing in the i^{th} row of the data matrix $\boldsymbol{\mathcal{R}}$. There are two possibilities: (i) metric covariate has the missing value (ii) a categorical covariate has the missing value. One finds the k nearest neighbor observation vectors $\boldsymbol{R}_k$ based on the distances

$$\boldsymbol{R}^D_{(1)}, \ldots, \boldsymbol{R}^D_{(k)} \quad \text{with} \quad d(\boldsymbol{R}_i, \boldsymbol{R}_{(1)}) \leq \cdots \leq d(\boldsymbol{R}_i, \boldsymbol{R}_{(k)})$$

where the distances are computed using equation (2.12). The computation of the imputed value depends on the type of variable and computed differently for metric and categorical variable.

Imputing Categorical Missing Value

Let z_{is} be the missing value in the i^{th} row and s^{th} categorical variable. For the imputation of the values z_{is}, we use the weighted estimator

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\boldsymbol{R}_i, \boldsymbol{R}_{(j)}) z_{(j)sc},$$

where $c = 1, \ldots, k_s$. The weighted imputation estimate of a categorical missing value z_{is} is

$$\hat{z}_{is} = \arg\max_{c=1}^{k_s} \hat{\pi}_{isc},$$

i.e. the value of $c \in \{1, \ldots, k_s\}$ with highest value of $\hat{\pi}$.

Imputing Continuous Missing Value

Let x_{is} be the missing value in the i^{th} row and s^{th} continuous variable. The weighted imputation estimate of continuous missing value, x_{is}, is defined by

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\boldsymbol{R}_i, \boldsymbol{R}_{(k)}) x_{(j)s}$$

2.8. Bootstrap Inference

Imputation is not the ultimate goal of any analysis. Missing values are generally not ignorable and hence imputed using some imputation technique. It is not compulsory that an estimate with smaller imputation error also performs better in downstream analysis like regression analysis etc. Therefore, it is customary to compare the performance of imputed data in further analysis. Under some basic assumptions, multiple imputation provides valid inferences and therefore remains the standard approach (Rubin, 1987). In the case of large samples, this approach gives nice properties under some assumptions including normality. However when these assumption are not met, the confidence intervals and inferences based on them should be use carefully (Nielsen, 2003). In single imputation we obtain one estimate for each missing value, and when we take the asymptotics of this estimate like in GLM, it is probably wrong because we are treating the imputed values as if

they have been complete. They are not reflecting the uncertainty in the variance estimates of the regression coefficients.

Bootstrap is a popular and computer intensive method which is often used for estimating the sampling distribution of estimators. Efron (1979) introduced this concept for independently identically distributed sample, and Rao and Wu (1988) provided the extensions for complex designs. They are valid for complete data only. When the data has been imputed to fill the missing values, the usual bootstrap methods should be used carefully as they may provide invalid results (Shao and Sitter, 1996). More specifically, if the estimated missing values are also treated like the true values, then the variance of the imputation estimate is underestimated because the expansion in variance due to imputation has been ignored. Alternatively, to get asymptotically valid variances and estimators of the population parameters the bootstrap sample should be imputed in the similar way as the original data (Shao and Sitter, 1996). A fractional bootstrap resampling method was proposed by Bello (1994) to obtain a *representative* sample relative to the proportion of missing values in the original data.

In particular, we investigate the performance of the inferences based on the imputed data by weighted nearest neighbors including selection of predictors (`wNNSel`). We present analytic techniques for inference from a dataset in which missing values have been replaced by nearest neighbors imputation method. In particular, we use bootstrap sampling to estimate the parameters and their confidence intervals. The confidence intervals for the regression coefficients are constructed using bootstrap sampling. A simple and easy to use bootstrap algorithm that combines the nearest neighbors imputation with bootstrap resampling estimation to obtain valid bootstrap inference in a linear regression model is suggested (Chapter 7). More specifically, imputing the bootstrap samples in the exact same way as original data was imputed produces correct bootstrap estimates. Simulation results show the performance of our approach in different data structures.

2.9. Missing Data and Classification

In the particular case of classification, incomplete data in either the training set or test set or in both sets affect the prediction accuracy of learned classifiers. A naive imputation method can affect the performance of a classifier constructed using that data set as a training sample. An imputed value is always a guess at the actual value. The imputation variation is natural due to the uncertainty about the actual value. Imputation can lead to an underestimation or overestimation of the uncertainty of the estimate. If the uncertainty is underestimated, the classifier trained with the imputed data set will overfit the training data and produce invalid outputs. On the other hand, if the uncertainty is overestimated, the classifier trained with the imputed data set will underfit the training data and provide poor prediction (Gheyas and Smith, 2010).

The literature relating to the classification with missing data usually focuses on analyzing and comparing a particular imputation method with the others for a specific proportion of

missing values under a particular missing mechanism. Luengo et al. (2012) used a variety of imputation techniques, and different classification methods by splitting them into three categories. Their findings proved the importance of imputation over case deletion. Furthermore, none of the imputation method could universally outperform the others regarding the accuracy of classification. Our objective is to assess whether the classification accuracy is affected with randomly generated missing values comparing to the same classifier trained and tested using the complete data set (Chapter 8).

2.10. Multiple Imputation in High-Dimensional Data

Due to the advancement and rapid growth in technology, the collection of high-dimensional data is no longer a tedious task. For *single imputation* in high-dimensional settings several approaches are available. An algorithm to impute missing data in high-dimensional settings, which is based on the popular random forests technique was proposed by Stekhoven and Bühlmann (2012). Random forests avoid overfitting by bootstrap aggregation of multiple regression trees. Moreover, the accuracy of predictions is enhanced by combining the predictions across all the trees (Breiman, 2001). An adaptation of this method and its comparison to parametric imputation methods was given by Shah et al. (2014). The results of their studies showed that the method is more efficient and typically produces confidence interval that are narrower than standard MI approaches. Four different variants of the nearest neighbors imputation method were developed by Liao et al. (2014). (See also Chapter 3, 4, 5 and 6). But all of these methods do not properly account for the uncertainty of imputation, Deng et al. (2016) regarded them as *improper* in the sense of Rubin (1987).

Multiple imputation is one of the most popular methods for handling missing values, as propagated by Rubin (1987). It is a technique to generate more than one plausible value for each missing value in data. It can provide valid results even when the missing mechanism is missing at random (MAR) (He and Belin, 2014). Multiple imputation (MI) accounts for the uncertainty due to imputation by explicitly using the estimates obtained for the different imputed datasets. Application of multiple imputation involves three stages:

(i) *Imputation*, that is, each missing value of the data is imputed $M \geq 2$ times. This stage results in M complete data sets.

(ii) *Analysis*, that is, each complete imputed dataset is analyzed independently using standard statistical techniques for complete data.

(iii) *Pooling*, that is, the estimates of the M analyses are combined into one set of parameter estimates. When pooling the estimates one accounts for the missing data uncertainty and sampling variation.

To get an overall inference, estimation results obtained for each of the imputed data set are combined across all the imputed data sets. Rubin (1987) provided some rules to combine the estimated parameters and their standard errors. (see also, Little and Rubin, 2014, Rubin, 2004). In recent years, MI has emerged as the leading approach for imputing missing data

with lots of advancements in methodology and software (Harrel and Zhou, 2007; Horton and Kleinman, 2007).

Issues in Existing MI Methods

In general, the *multiple imputation* methods suffer from the curse of dimensionality and seem not to work very well. When the number of predictors is less than the sample size, many software packages provide an easy application of MI methods, for example, the `R` packages `mice` (van Buuren and Groothuis-Oudshoorn, 2011) and `Amelia` (Honaker et al., 2011). When the number of predictors is less than but close to the sample size, the existing methods can still be applied to get imputation results but may not perform well (Zhao and Long, 2016). However, when the number of predictors exceeds the sample size, the available software packages typically fail. Some researchers have suggested to use regularized regression methods, which allow for variable selection before building the final imputation model (Long and Johnson, 2015). For a multivariate normal model, Song and Belin (2004) presented a method to extract the common factors in high-dimensional data so that the number of parameters to be estimated is reduced. To impute missing data in high-dimensional data structures, Zhao and Long (2016) proposed the Bayesian lasso regression for multiple imputation with normally distributed data. The approach is computationally very demanding and becomes infeasible for an increasing number of variables with missing data. Another adaptation of `mice` for high-dimensional setting was given by Deng et al. (2016). But, as mentioned in the paper, the theoretical properties of `mice` are still not well established.

Although MI methods have been very popular and the most widely used approach for imputing missing data, the validity of MI is based on various assumptions. For example, most of the MI methods assume that the missing data follow the MAR mechanism (Little and Rubin (2002)). Another major issue is the correct (or at least close to the true model) specification of the imputation model (Rubin, 1996, Akaike, 1974). Another drawback of conditional MI is that it may not converge to the correct posterior distribution, similar to the Gibbs sampler.

2.10.1. MI using Nearest Neighbors

Nearest neighbors (NN) is a popular non-parametric technique that has been widely used for single value imputation (Troyanskaya et al., 2001). Many variants have been proposed in the literature as improvements over the original kNN method, for example, Ouyang et al. (2004), Kim et al. (2004), Scheel et al. (2005), Johansson and Hakkinen (2006), Bø et al. (2004), and Zhang et al. (2008). An adaptation of nearest neighbor imputation was given by Verboven et al. (2007) for the imputation of missing values sequentially. They successfully applied the method to gene expression data, and Branden and Verboven (2009) provided a robust version of their method. Tutz and Ramzan (2015) proposed a weighted nearest

neighbor approach to impute missing data. It was shown that the method provides better imputation results when compared to existing approaches (Chapter 3). The approach is particularly useful when the data suffers from the curse of dimensionality (Faisal and Tutz, 2017c). See also, Chapter 4. Since these methods impute only one plausible value for each missing value, they share all the drawbacks of single imputation.

The objective is to demonstrate that NN methods can also be used for multiple imputation, which has advantages, in particular when the number of covariates is large when compared to the sample size (e.g., genetics studies). In particular we propose two methods:

- A sequential NN imputation method (multiple imputation using sequential nearest neighbors-`miNNseq`). Our first approach is based on imputing the missing values in a sequential order. The missing values are imputed one at a time using the `wNNSel` method. After imputing one missing value, it is considered as observed to obtain an updated matrix which is used for the imputation of the next missing value. Thus each imputed value contributes to the imputation of others. For each data set the first value to be imputed is selected at random.
- A combination of the bootstrap and NN (multiple imputation using nearest neighbors and bootstrapping-`miNNboot`) Alternatively, we make use of the famous bootstrap algorithm to impute missing values. More specifically, we propose to select M bootstrap samples, with replacement, of the same size n from the original missing data matrix. Then we impute missing values for each of the bootstrap sample using the `wNNSel` method. When using bootstrap the sampling as well as the imputation uncertainty is taken into account.

The proposed methods (Chapter 9) are compared with the widely used MI method Multivariate Imputation by Chained Equations (MICE). We report the results of imputation errors, and investigate the performance of inference based on the imputed data by means of the accuracy of the estimated coefficients and the confidence intervals of the estimated coefficients.

3. Improved Methods for the Imputation of Missing Data by Nearest Neighbor Methods

Abstract

Missing data raise problems in almost all fields of quantitative research. A useful nonparametric procedure is the nearest neighbor imputation method. Here we present improved versions of this method. First, a weighted nearest neighbor imputation method based on L_q distances is proposed. It is demonstrated that the method tends to have a smaller imputation error than other nearest neighbor estimates. We then consider weighted- neighbor imputation methods that use distances for selected covariates. The careful selection of distances that carry information about the missing values yields an imputation tool that can outperform competing nearest neighbor methods . This approach performs well, especially when the number of predictors is large. The methods are evaluated in simulation studies and with several real data sets from different fields.

Keyword: Kernel function, Weighted nearest neighbors, Cross-validation, Weighted imputation, MCAR

3.1. Introduction

Missing data have always been a challenge to researchers. Since the 1980s, many techniques to impute missing data have been proposed, for example, Little and Rubin (2002) and Schafer (2010). Broadly speaking, the methods for filling in an incomplete data matrix can be divided into two main categories, single imputation and multiple imputation (Little and Rubin, 2002). A well-known and computationally simple method for the imputation of missing data is mean substitution. However, its disadvantage is that the correlation structure among the predictors is ignored. An alternative is the *nearest neighbors* (NNs) approach, which uses observations in the neighborhood to impute missing values. NNs as a nonparametric concept in discrimination dates back to Fix and Hodges (1951). The approach has been successfully used to impute data in gene expression (Troyanskaya et al., 2001; Atkeson et al., 1997; Hastie et al., 1999), machine learning (Batista and Monard, 2002), medicine (Waljee et al., 2013), forestry (Eskelson et al., 2009; Hudak et al., 2008), and compositional data (Hron et al., 2010).

An important aspect in missing data imputation is the pattern of missing values because it determines the selection of an imputation procedure. Little and Rubin (2002) defined three categories of missing data, missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR). MCAR refers to data in which the probability of an observation being missing does not depend on the variable itself or on any other variable in the data set (Little and Rubin, 2002; Allison, 2001). Most imputation methods assume the data to be at least MAR, if not MCAR, and so does the NN method. Before considering NN approaches in detail, a few general remarks on the method are warranted. The main approach for imputation is multiple imputation because it yields valid inferences under several basic conditions. In contrast to NN approaches, multiple imputation provides consistent estimates under MAR conditions. However, NN approaches are useful in high-dimensional problems in which multiple imputation cannot be applied and complete cases do not generate enough data.

Several NN approaches to imputation have been considered in the literature. Hastie et al. (1999) proposed using weights based on the Euclidean distance for NN selection. Troyanskaya et al. (2001) compared the k nearest neighbor imputation (KNNimpute) with the mean imputation and singular-value decomposition (SVD) techniques for gene expression data. Their simulation studies showed that the method performs well compared to mean imputation and SVD approaches, as also demonstrated by Troyanskaya et al. (2003). In a comparative study of single imputation methods, Malarvizhi and Thanamani (2012) found that median or standard deviation substitution perform better than mean substitution. A further comparison of KNNimpute with mean, ordinary least squares (OLS) and partial least squares (PLS) imputation methods in microarray data was that of Nguyen et al. (2004). In that study, the good performance of KNNimpute was demonstrated.

Several alternative procedures have been proposed that rely on basic concepts to impute values by building averages over qualifying neighbors, as described in Ouyang et al. (2004), Kim et al. (2004), Sehgal et al. (2005), and Scheel et al. (2005). Liew et al. (2011) and

Moorthy et al. (2014) reviewed the available methods and algorithms, with a focus on gene expression data. Johansson and Hakkinen (2006) proposed *WeNNI*, which utilizes continuous weights in the NN imputation procedure. Bø et al. (2004) and Wasito and Mirkin (2005) proposed a NN procedure with a modification based on the least squares method. Other variants include the local least squares (LLSimpute) method of Kim et al. (2005), the sequential local least squares (SLLSimpute) method of Zhang et al. (2008), and the iterative LLS (ILLSimpute) of Cai et al. (2006).

A drawback of NN methods is that their performance depends on k. For example, KNNimpute typically performs well when k is between 5 and 10, but the performance deteriorates for larger values of k (Yoon et al., 2007). Here we propose a localized approach to missing data imputation that uses a weighted average of NNs based on L_q distances. For the high-dimensional case, we propose a new distance that explicitly uses the correlation among variables. The method automatically selects the relevant variables that contribute to the distance and thus does not depend on k.

The paper is organized as follows: In Section 3.2, the L_q distance is used to define a weighted imputation estimate. In a simulation study, the weighted approach is compared to the unweighted approach. In Section 3.3, the weighted imputation with the selection of predictors is introduced and compared to alternative imputation techniques. In Section 3.4 ,several applications using real data sets are demonstrated.

3.2. Weighted Neighbors

When using NNs to impute data, two preliminary steps must be carried out. First, one has to define the NNs, that is, how they are computed and which distance measures are to be used. Second, one has to determine how these NNs are used to obtain an imputed value. The choices to be made in these two steps are considered in the following sections.

3.2.1. Distances and Computation of Nearest Neighbors

Let n observations on p covariates be collected. The corresponding $n \times p$ data matrix is given by $\mathbf{X} = (x_{is})$, where x_{is} denotes the ith observation of the sth variable. Let $\mathbf{O} = (o_{is})$ denote the corresponding $n \times p$ matrix of dummies with entries

$$o_{is} = \begin{cases} 1 & \text{if } x_{is} \text{ was observed} \\ 0 & \text{for missing value.} \end{cases}$$

Distances between two observations $\mathbf{x}_i$ and $\mathbf{x}_j$, represented by rows in the data matrix, can be computed by using the L_q metric for the observed data, given by

$$d_q(\mathbf{x}_i, \mathbf{x}_j) = [\frac{1}{a_{ij}} \sum_{s=1}^{p} |x_{is} - x_{js}|^q I(o_{is} = 1) I(o_{js} = 1)]^{1/q}, \tag{3.1}$$

where $a_{ij} = \sum_{s=1}^{p} I(o_{is} = 1)I(o_{js} = 1)$ denotes the number of valid components in the computation of distances. The indicator function $I(a)$, which is used in the definition, has the value 1, if a is true and 0 otherwise. The computation of distances does not use all the components of the vectors but only those for which observations in both vectors are available. The components included in the computation of neighbors are given by $C_{ij} = \{s : I(o_{is} = 1)I(o_{js} = 1) = 1\}$. The distances define the NNs when imputing a specific value. It should be noted that the number of components varies over the sample because C_{ij} depends on the specific observation for which imputations are to be made.

Similar concepts to define distances and therefore NNs were described by Hastie et al. (1999), Troyanskaya et al. (2001), Myrtveit et al. (2001), and Kim et al. (2005). In those studies the Euclidean distance was $q = 2$. Alternatives to compute distances in gene expression studies are based on the Pearson correlation (Dudoit et al., 2002, Bø et al., 2004) and on covariance estimates (Sehgal et al., 2005).

3.2.2. Imputation Procedure

Imputation for Fixed Number of Neighbors

Consider the imputation for $\mathbf{x}_i$ in component s, that is, $o_{is} = 0$. The imputation estimate for $\mathbf{x}_i$ is based on the k NNs in the reduced data set. The k NNs are determined from the corresponding $(\tilde{n} \times p)$-dimensionally reduced data set $\tilde{\mathbf{X}} = (x_{ij}, o_{is} = 1)$ to obtain

$$\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(k)} \quad \text{with} \quad d(\mathbf{x}_i, \mathbf{x}_{(1)}) \leq \cdots \leq d(\mathbf{x}_i, \mathbf{x}_{(k)}),$$

where $\mathbf{x}_{(j)}^T = (x_{(j)1}, \ldots, x_{(j)p})$ denotes the jth NNs. The *imputation value for a fixed k* is then

$$\hat{x}_{is} = \frac{1}{k} \sum_{j=1}^{k} x_{(j)s}. \tag{3.2}$$

Thus, the missing value in the sth component of observation vector $\mathbf{x}_i$ is replaced by the average of the corresponding values of the k NNs. The accuracy of the method is mainly determined by the number of neighbors that are used. Simple rules use a fixed number of neighbors that is chosen by the experimenter. However, it is advantageous to consider the number of neighbors as a tuning parameter that is chosen in a data-driven manner, for example, by cross-validation (see the next section).

Imputation by Weighting

A disadvantage of the imputation based on the k NNs is that the value of the first NN has the same importance as the kth NN. A more appropriate method uses weights that account for the distance of the observations. We consider a weighted average of neighbors based

on distances that are determined by kernel functions. The *weighted imputation estimate* considered here has the form

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\mathbf{x}_i, \mathbf{x}_{(j)}) x_{(j)s} \tag{3.3}$$

with weights

$$w(\mathbf{x}_i, \mathbf{x}_{(j)}) = \frac{K(d(\mathbf{x}_i, \mathbf{x}_{(j)})/\lambda)}{\sum_{l=1}^{k} K(d(\mathbf{x}_i, \mathbf{x}_{(l)})/\lambda)}, \tag{3.4}$$

where $K(.)$ is a kernel function (for example tricube, Gaussian) and λ is a tuning parameter. For small λ the weights decrease very strongly with distance, whereas for $\lambda \to \infty$ all neighbors are of equal weight. To make λ the crucial tuning parameter, a large number of potential neighbors should be used. In the extreme case, if $k = \tilde{n}$, the window-width λ will be the only tuning parameter.

Alternative weights were described by Troyanskaya et al. (2001). They used the weighted average over the k NNs based on the Euclidean distance, with the weights determined by the inverse of that distance.

3.2.3. Choice of Tuning Parameters by Cross-Validation

An important issue in *weighted nearest neighbors imputation* techniques is the selection of the tuning parameter λ. One option is to choose the same method that is commonly used when selecting the optimal number of NNs, namely, cross-validation, (see, for example, Dudoit et al. (2002)). In addition, some R packages provide cross-validation as a tool to choose the optimal number of NNs (Waljee et al., 2013). The concept is to artificially delete some of the values in the data matrix and to compute the mean squared imputation error (MSIE) or the mean absolute imputation error (MAIE) for the artificially deleted data. The value for which the value of the MSIE or MAIE is the smallest is then chosen.

Specifically, we generate completely at random (MCAR) m^* artificially missing values from the available data $\{x_{is} : o_{is} = 1\}$. Let $\mathbf{X}^*$ denote the $n \times p$ data matrix that contains the originally missing values ($\{x_{is} : o_{is} = 0\}$) and the artificially missing values ($\{x^*_{is} : o^*_{is} = 0\}$). The MAIE for these observations is then defined by

$$\text{MAIE}(\mathbf{X}^*) = \frac{1}{m^*} \sum_{x_{is}: o^*_{is}=0} |x^*_{is} - x^*_{is(imputed)}|. \tag{3.5}$$

The procedure is repeated R times yielding the averaged value

$$\mathrm{MAIE}_{\mathrm{CV}} = \frac{1}{R}\sum_{r=1}^{R}\mathrm{MAIE}(\mathbf{X}_r^*),$$

where $\mathbf{X}_r^*$ denotes the matrix with both types of missing values in the rth replication. Similarly, the cross-validated MSIE is computed as follows

$$\mathrm{MSIE}(\mathbf{X}^*) = \frac{1}{m^*}\sum_{x_{is}:o_{is}^*=0}(x_{is}^* - x_{is(imputed)}^*)^2 \tag{3.6}$$

and yields the corresponding $\mathrm{MSIE}_{\mathrm{CV}}$.

In summary, the cross-validation is performed as follows:

1. For a specific value of λ;
   a. Artificially delete m^* value/s in the data matrix.
   b. Impute these missing values and calculate MSIE or MAIE.
   c. Repeat a and b, R times, for example, $R = 100$, to obtain $\mathrm{MSIE}_{\mathrm{CV}}$ or $\mathrm{MAIE}_{\mathrm{CV}}$.

2. Repeat steps **a-c** for all values of λ and choose the parameter with the minimum value of $\mathrm{MSIE}_{\mathrm{CV}}$ or $\mathrm{MAIE}_{\mathrm{CV}}$ as the optimal λ.

Within the procedure, the number of values that are deleted artificially has to be chosen. We used several values, ranging from the deletion of only one observation to 10% of the data. In our simulations the resulting values of λ were very similar. Therefore, in the following we chose to delete 10% of the data.

3.2.4. Performance Measures

Once the tuning parameter has been chosen, the performances of the different imputation methods are compared on the basis of the MSIE) and the MAIE computed from the original and imputed values in the same way as for the cross-validation (Troyanskaya et al., 2001; Junninen et al., 2004; Hudak et al., 2008). In simulation studies, the evaluation is simple. If $\{x_{is} : o_{is} = 0\}$ are the missing values in the data matrix $\mathbf{X}$, then the corresponding imputed values are computed using the λ selected by cross-validation. The performance is then measured by computing the MAIE or MSIE , using the missing observations and the imputed values.

In the case of real data, the original values are not known. Thus, instead, the performance is evaluated by considering part some of the observations $\{x_{is} : o_{is} = 1\}$ as missing and computing the corresponding imputed values to obtain the MAIE or MSIE. The number of values considered as missing in the evaluation of the performance depends upon the sample size.

3.2.5. Evaluation of Weighted Neighbors

The correlation structure of the data plays an important role in the selection of the imputation method (see, for example,Feten et al. (2005)). Therefore, in this section we investigate the dependence of the method on the correlation between variables and compare the un-weighted and weighted NN versions of imputation in a simulation study.

Imputation Based on a Fixed Number of Nearest Neighbors

In the simulated data, observations were drawn from a p-dimensional multivariate normal distribution with a mean of $\mathbf{0}$ and a pairwise correlation of ρ. We used five values for the correlation, $\rho \in \{= 0.1, 0.3, 0.5, 0.7, 0.9\}$. For each value of ρ, $S = 200$ samples, each of size $n = 50$, were drawn for $p = 10$ covariates from $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is a correlation matrix with the pairwise correlations ρ among the covariates. The un-weighted NN was used to impute (i) 5%, and (ii) 25% missing values (MCAR) in each simulation setting. Fixed numbers of NN $k \in \{1, \ldots, 40\}$, were used to estimate the missing values. The performance of the procedure in terms of the MSIE and MAIE is shown in Fig. 3.1 for all values of k and for selected values of ρ (0.3, 0.5, 0.9).

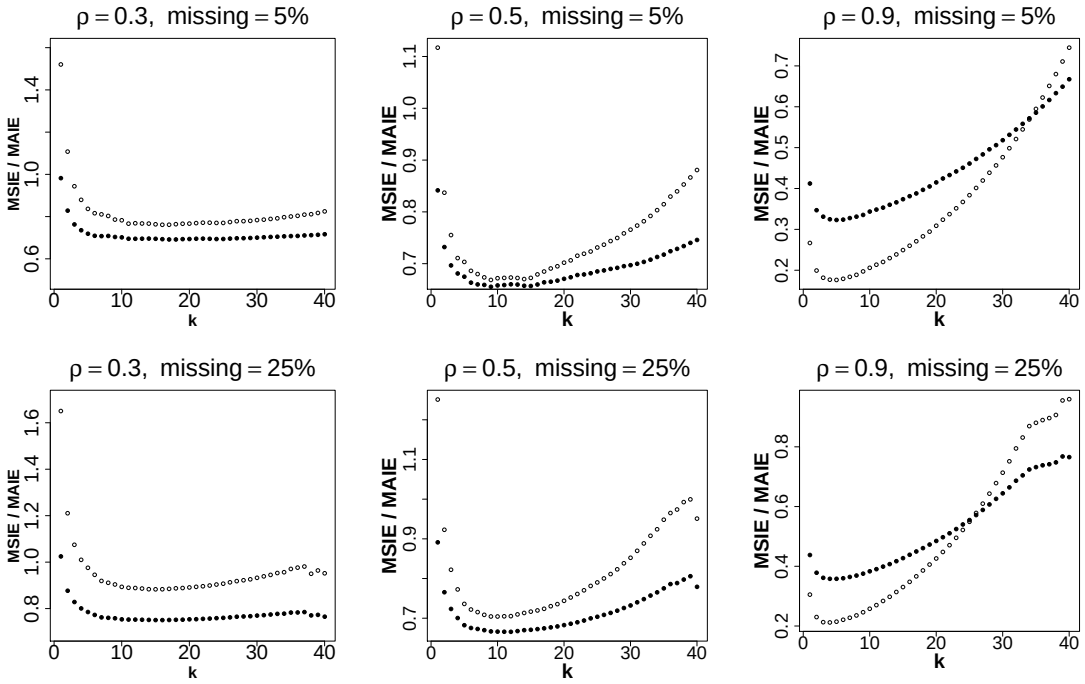


Figure 3.1.: Illustration of simulation study: Mean Squared Imputation Error (MSIE) and Mean Absolute Imputation Error (MAIE) for un-weighted nearest neighbors using L_2 metric for fixed values of k (number of nearest neighbors), with 5% missing data (Upper panel) and 25% missing data (lower panel). Circles represent MSIE and solid circles represent MAIE.

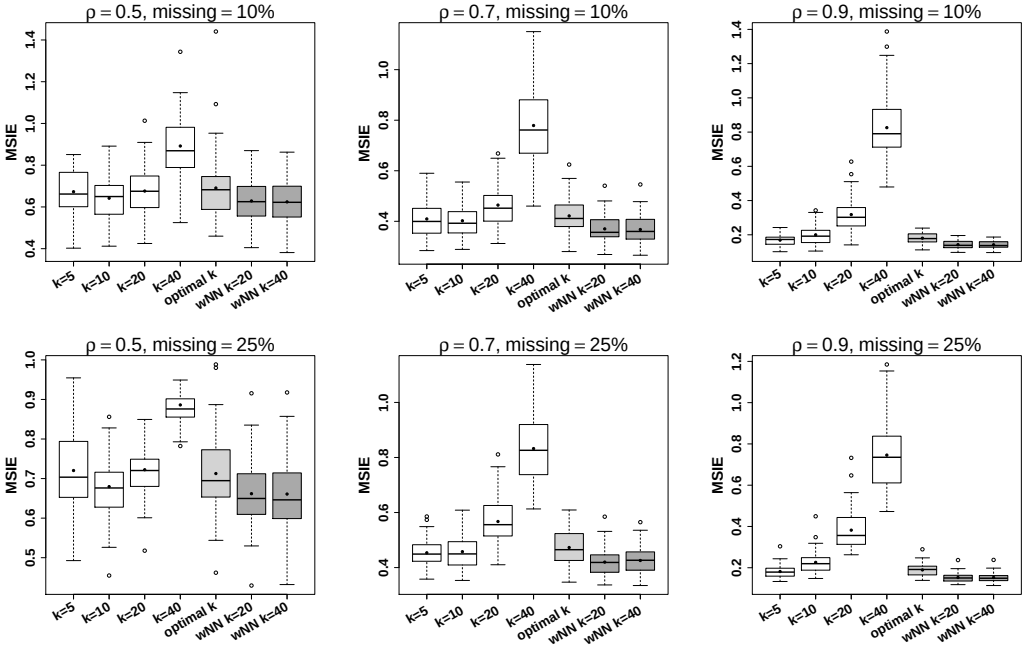
Figure 3.1 shows that for a low correlation, for example, $\rho = 0.3$, the MSIE/MAIE is high for small values of k and decreases with the increasing value of k but remains almost stable for $k \geq 10$. If small values of k are excluded, the performance does not strongly depend on the chosen number of the next neighbors. The situation changes if the correlation among the variables is stronger. In that case there is a distinct minimum and the performance deteriorates if k increases. It is worth noting that the performance is much better when the variables are correlated. For $\rho = 0.3$ and 5% missing values, the best obtained value is approximately 0.7 whereas for $\rho = 0.9$ and 5% missing values the best value is below 0.2. Imputation works much better if the variables are correlated since only information from the other variables is available. Therefore, in the following, we focus on correlated data.

Weighted Imputation

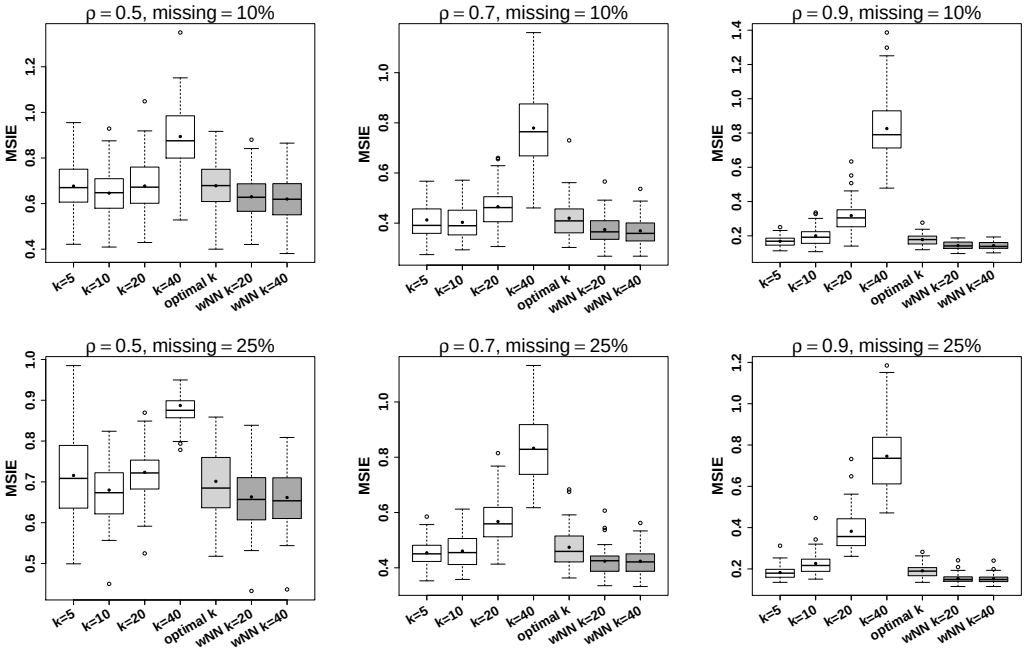
In the next simulation the weighted nearest neighbor method, henceforth denoted as `wNN`, is compared to an un-weighted imputation. Again $S = 200$ samples, each of size $n = 50$ were drawn for $p = 15$ covariates from $N(\mathbf{0}, \mathbf{\Sigma})$. The values of the pairwise correlation used in $\mathbf{\Sigma}$ were $\rho = 0.5,\ 0.7$, and 0.9. In each sample we considered 10% and 25% of the total values as MCAR. The distance (3.1) with q=1 (L_1 distance) and q=2 (L_2 or the Euclidean distance) was used to impute the missing data. The optimal value of the tuning parameter λ for weights (3.4) was chosen by cross-validation based on the MSIE and MAIE. As these showed similar patterns, only the MSIE results are presented. The weights were calculated using a Gaussian kernel because simulations had shown that the performance of the Gaussian kernel was slightly better than that of an alternative kernel, such as the triangular one.

Figure 3.2 shows the MSIEs for the NN imputation method with fixed values of $k = 5, 10, 20, 40$, for a k chosen by cross-validation, and for the weighted approach with $k = 20, 40$ as the maximal neighbors. The boxplots of MSIE with L_1 distance are shown in the upper panel of Fig. 3.2, with the L_2 distance in the lower panel. As seen in the figure, the `wNN` imputation works well when the data are highly correlated. The MSIE for $\rho = 0.5$, 10% missing, is larger than 0.6, whereas for $\rho = 0.9$, 10% missing, it is below 0.2 (Fig. 3.2(a)). Similar results are found obtained for the L_2 distance (Fig. 3.2(b)). Overall, L_1 and L_2 distances yield nearly the same MSIEs in all data settings.

In addition to the dependence on the correlation structure, we investigated whether cross-validation is able to obtain a proper value for k, the number of NNs to be used. The boxplots show that, although certain specific values of k perform better, the performance of the procedure that selects k by cross-validation (denoted by optimal k) is also acceptable. The other point of interest is the proposed weighted imputation (`wNN`). Its performance is always better than that of the imputation for a fixed value of k, when the window width and the number of NNs are chosen by cross-validation. The detailed results are shown in Table 3.1, where for each scenario the best-performing method is shown in boldface. For $n = 50$, the differences are slight when allowing for 10, 20, or 40 NNs in the weighting procedure. Nevertheless, for 40 NNs the performance tends to be slightly better. In summary, weighted



(a) L_1 distance



(b) L_2 distance

Figure 3.2.: Boxplots of mean squared imputation error for NN Imputation using L_1 distance (a) and L_2 distance (b). Imputation is done by using un-weighted/fixed values of $k = 5$, 10, 20, 40 nearest neighbors (white boxes), for optimal value of k selected by cross-validation (light grey boxes), and weighted approach (`wNN`) with maximum $k = 20$, 40 using Gaussian kernel. Solid circles within boxes show the mean values.

imputation is an attractive alternative that, computationally, is not more demanding than imputation with a fixed k.

3.3. Weighted Neighbors Including the Selection of Predictors

3.3.1. Selection of Dimensions

Traditionally, the distances are computed from all the available components of the observations. However, as will be shown, in high-dimensional settings imputation suffers from the curse of dimensionality. Therefore, we propose to compute distances from selected dimensions only. Since imputation is successful especially when the predictors are highly correlated, selection of the dimensions will be linked to the correlation between predictors. In the following, an extended version of the weighted NN imputation method is given that uses only the correlated predictors to compute the distances.

Consider the imputation for $\mathbf{x}_i$ in component s ($o_{is} = 0$). When computing distances from the reduced data set $\{\mathbf{x}_j, o_{js} = 1\}$, an additional weight is used in the distances. Specifically, for distance L_q one computes component-specific distances by

$$d_{q,C}(\mathbf{x}_i, \mathbf{x}_j) = \{\frac{1}{a_{ij}} \sum_{l=1}^{p} |x_{il} - x_{jl}|^q I(o_{il} = 1) I(o_{jl} = 1) C(r_{sl})\}^{1/q}, \tag{3.7}$$

where r_{sl} is the empirical correlation between covariates s, l and $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the correlations into weights. The transformation is constructed such that only the covariates that are strongly correlated with component s strongly contribute to the computation of distances, while components that are not correlated do not contribute to the distance. A convex function $C(.)$ that can be used is defined by

$$C(r) = \begin{cases} \frac{|r|}{1-c} - \frac{c}{1-c} & \text{if } |r| > c \\ 0 & \text{if } |r| \leq c. \end{cases} \tag{3.8}$$

It is linear in the absolute value of the correlation. If $|r_{sj}| \leq c$, the component s does not contribute to the distance. If $|r_{sj}| > c$, then the weights increase linearly with increasing correlation and have a weight of 1 if $|r| = 1$. The threshold parameter c can be chosen as fixed, for example by $c = 0$, or can be considered as an additional tuning parameter from the range $[0, 1)$. A smoother function, which can also be used, is the power function

$$C(r) = |r|^m \tag{3.9}$$

Table 3.1.: MSIE for NN imputation with weighted and un-weighted approach using $L_1$and L_2 distances

			n=50							
			Fixed /un-weighted					wNN		
Distance	ρ	missing (%)	$k=5$	$k=10$	$k=20$	$k=40$	optimal k	$k=10$	$k=20$	$k=40$
L_1	0.5	10	0.6798	0.6487	0.6819	0.8968	0.6894	0.6360	0.6239	**0.6189**
		25	0.7128	0.6772	0.7245	0.8893	0.7120	0.6733	0.6587	**0.6582**
	0.7	10	0.4241	0.4148	0.4836	0.8180	0.4324	0.3765	0.3677	**0.3673**
		25	0.4576	0.4595	0.5694	0.8459	0.4679	0.4211	0.4180	**0.4168**
	0.9	10	0.1681	0.1974	0.3139	0.7901	0.1781	0.1451	0.1424	**0.1417**
		25	0.1861	0.2305	0.3895	0.7582	0.1944	0.1540	**0.1521**	0.1528
L_2	0.5	10	0.6821	0.6523	0.6828	0.8976	0.6855	0.6404	0.6276	**0.6220**
		25	0.7092	0.6765	0.7247	0.8905	0.6966	0.6717	0.6589	**0.6548**
	0.7	10	0.4233	0.4142	0.4837	0.8182	0.4312	0.3751	0.3703	**0.3686**
		25	0.4572	0.4624	0.5695	0.8463	0.4728	0.4268	0.4226	**0.4198**
	0.9	10	0.1678	0.1975	0.3141	0.7900	0.1769	0.1438	0.1423	**0.1416**
		25	0.1869	0.2305	0.3896	0.7582	0.1941	0.1547	0.1525	**0.1523**

			n=100							
			Fixed /un-weighted					wNN		
			$k=10$	$k=20$	$k=40$	$k=80$	optimal k	$k=20$	$k=40$	$k=80$
L_1	0.5	10	0.6240	0.6180	0.6619	0.8964	0.6272	0.6030	0.5992	**0.5983**
		25	0.6449	0.6393	0.7011	0.9091	0.6598	0.6243	0.6218	**0.6213**
	0.7	10	0.3885	0.4046	0.4846	0.8443	0.4001	0.3651	0.3625	**0.3610**
		25	0.4045	0.4260	0.5339	0.8349	0.4185	0.3820	0.3790	**0.3788**
	0.9	10	0.1591	0.1950	0.3125	0.7936	0.1567	0.1329	0.1315	**0.1312**
		25	0.1664	0.2154	0.3821	0.8067	0.1620	0.1343	0.1338	**0.1336**
L_2	0.5	10	0.6222	0.6196	0.6637	0.8967	0.6368	0.6038	0.5992	**0.5990**
		25	0.6446	0.6417	0.7017	0.4090	0.6554	0.6268	0.6241	**0.6237**
	0.7	10	0.3884	0.4054	0.4846	0.8446	0.3965	0.3668	**0.3623**	0.3634
		25	0.4044	0.4263	0.5351	0.8353	0.4214	0.3830	0.3795	**0.3790**
	0.9	10	0.1585	0.1945	0.3130	0.7936	0.1586	0.1321	0.1316	**0.1312**
		25	0.1673	0.2157	0.3822	0.8068	0.1616	0.1350	0.1342	**0.1341**

with natural number m as an additional tuning parameter. However, a problem with this weighting scheme is that r_{sl} cannot be computed, since missing values are present in the data. Therefore, we use a simple first-step imputation with five un-weighted NNs and compute the correlations from the observed and imputed data. The actual imputation is then carried out by using the correlations computed in the first step. It should be noted that, depending on the scale level, alternative dependence measures might be more appropriate. For variables that have been measured on an ordinal scale, Spearman's correlation coefficient can be used. For categorical variables, a standardized chi-squared measure, such as Cramer's V-statistic, is an option.

The weighted L_q distances with the selection of dimensions includes the calculation of an additional tuning parameter. Thus, in addition to the tuning parameters for the kernels, λ, an appropriate value c for the convex function (3.8) or an integer m for the convex function (3.9) must be found. The parameters are chosen in the same way as λ in Section 3.2.3, namely, by cross-validation. For the simultaneous selection of the tuning parameters (λ, c) or (λ, m) cross-validation is computed on a two-dimensional grid of values. We refer to this method as `wNNSelect`. The linear function is characterized by c, and the power function by the parameter m.

3.3.2. Evaluation of the Method with Selected Weighted Neighbors

The simulation study in Section 3.2.5 showed that for highly correlated predictors weighted nearest neighbors (`wNN`) perform better than un-weighted distances. Moreover, the results showed that the distances L_1 and L_2 have very similar performances. In this section, the performance of the weighted approach with the selection of predictors is assessed. We consider, in particular, two structures of the correlation matrix, blockwise correlation and autoregressive (AR) type correlation.

Blockwise Correlation Structure

Let the data matrix $\mathbf{X}_{(n\times p)}$, be partitioned into $\mathbf{X} = [\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}]$ such that each element of $\mathbf{X}^{(q)} = [\mathbf{x}_1, \cdots, \mathbf{x}_{p_q}]$, $q = 1, 2, 3$, is a $n \times 1$ random vector. The partitioned correlation matrix has the form

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{\Sigma}_{11} & \vdots & \mathbf{\Sigma}_{12} & \vdots & \mathbf{\Sigma}_{13} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \mathbf{\Sigma}_{21} & \vdots & \mathbf{\Sigma}_{22} & \vdots & \mathbf{\Sigma}_{23} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \mathbf{\Sigma}_{31} & \vdots & \mathbf{\Sigma}_{32} & \vdots & \mathbf{\Sigma}_{33} \end{pmatrix}.$$

The matrices $\mathbf{\Sigma}_{ii}$ are determined by the pairwise correlations ρ_w of the elements of $\mathbf{X}^{(q)}$, $q = 1, 2, 3$, that is, all components have (*within*) correlation ρ_w. The matrices $\mathbf{\Sigma}_{ij}, i \neq j$, contain the pairwise (*between*) correlations (ρ_b) between all the elements of $\mathbf{X}^{(i)}$ and $\mathbf{X}^{(j)}$.

Autoregressive-Type Correlation Structure

The second correlation structure we consider is the exponential or *autoregressive* type. An AR correlation matrix of order one is defined by

$$\mathbf{\Sigma} = (\rho^{|i-j|}),$$

for $i, j = 1, \ldots, p$, where ρ denotes the pairwise correlation between predictors and p the number of variables.

Comparison of the Fixed and Selected Values of the Tuning Parameters

In the weighted imputation method considered in Section 3.2 (`wNN`), one tuning parameter λ for the kernel weights had to be chosen. In the selection of predictors method (3.7, an additional tuning parameter is needed, c or m. In the following we first investigate whether the data- driven selection of the tuning parameters (λ, c) or (λ, m) by cross-validation is possible.

We generated $S = 200$ samples of size $n = 50$ for $p = 30$ predictors drawn from a multivariate normal distribution with $N(\mathbf{0}, \mathbf{\Sigma})$. The structure of the correlation matrix $\mathbf{\Sigma}$ was of the blockwise or AR type. In the former, the predictors were chosen in three blocks of 10 predictors each, that is, the $p \times 1$ random vector $\mathbf{X} = [\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}]$, has components $\mathbf{X}^{(1)} = [X_1, \ldots, X_{10}]$, $\mathbf{X}^{(2)} = [X_{11}, \ldots, X_{20}]$, and $\mathbf{X}^{(3)} = [X_{21}, \ldots, X_{30}]$. The variables within each block correlated strongly, with $\rho_w = 0.70, 0.90$, but were nearly uncorrelated with the variables in the other blocks ($\rho_b = 0.10$). In the AR type structure, $\rho = 0.9$. In each sample, 10% and 25% of the total values were replaced by values that were MCAR.

The following six data settings were used in the simulation study:.

Setting	Structure	Correlation	missing
1	Blockwise	ρ_w=0.7, ρ_b=0.1	10%
2			25%
3		ρ_w=0.9, ρ_b=0.1	10%
4			25%
5	Autoregressive	$\rho = 0.9$	10%
6			25%

In the simulation study, we used the convex functions (3.8) and (3.9) at fixed levels and selected the tuning parameters by cross-validation. For the tuning parameter c in the linear function (3.8), a grid of 20 values from the interval $[0, 1)$ was considered. Therefore, all predictors whose correlation with the target predictor was less than or equal to the specified c, were discarded during computation of the distances. For the power function (3.9), the tuning parameters $m \in \{1, 2, \cdots, 8\}$ was used. The number of NN k was set to the maximum of available neighbors. The large value of k was chosen because the imputation procedure automatically selects the relevant neighbors.

In the simulation study, we used both convex functions (3.8) and (3.9) at fixed levels and selected the tuning parameters by cross-validation. For the tuning parameter c in the linear function (3.8) we considered a grid of 20 values from the interval $[0, 1)$. Therefore, all the predictors whose correlation with the target predictor was less than or equal to the specified c, were discarded during the computation of the distances. For the power function (3.9), we used the tuning parameters $m \in \{1, 2, \cdots, 8\}$. The number of nearest neighbors k was set to the maximum of available neighbors. The large value of k was chosen since the imputation procedure automatically selects the relevant neighbors.

A visual comparison of the wNN imputation and the imputation with selected dimensions is given in Figure 3.3. It shows the performance for fixed tuning parameters c and m as well as the performance if the tuning parameters are selected by cross-validation ($\texttt{wNNSelect}_c$ and $\texttt{wNNSelect}_m$). For ease of presentation, only the results for the values $c \in \{0.2,\ 0.3,\ 0.4,\ 0.5\}$ and $m \in \{2, 4, 6\}$ are shown. The method with the selection of predictors (`wNNSelect`) clearly performs much better than the simple weighted NN imputation without the selection of distances (`wNN`). Again, the results from L_1 and L_2 distances do not differ significantly. In addition, good results are obtained with methods involving the data-driven selection of tuning parameters c and m. For both types of correlation matrices, these methods it yield smaller MSIEs than for obtained with fixed c and m. Other simulation settings (data not shown) produced similar results. Therefore, in the following, the tuning parameter selected by double grid cross-validation is used.

Dependence on the Number of Predictors

That the selection of sub-spaces may yield better results was shown in Figure 3.3. In the following, we determine how strongly an imputation based on distances computed from the whole set of predictors suffers when the number of predictors increases. Based on the results of the simulation study in the previous sub-section, we use cross-validation to select tuning parameters on a double grid (λ, c) and (λ, m).

We use S=200 samples of size $n = 50$ with the numbers of predictors given by $p \in \{$ 5,10,20,30,40,50,60,70,80,90,100$\}$ and $n = 100$ with $p \in \{$5,10,20,30,40,50,60,70,80, 90,100,120,150$\}$ from $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the AR(1) correlation matrix with $\rho = 0.9$.

For comparison, we use established NN methods that are available in the `R environment` (R Core Team, 2013). In the package `impute` from Bioconductor, NN are used to impute

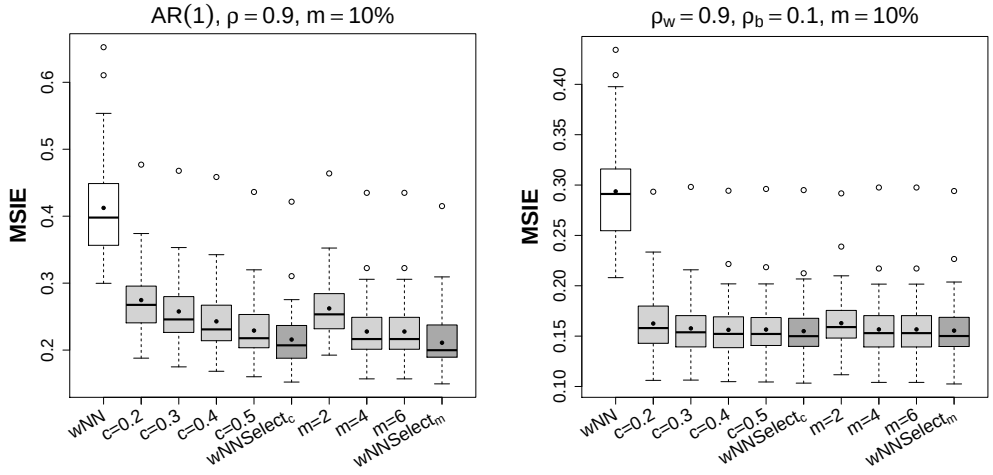


Figure 3.3.: Comparison of weighted imputation (white boxes), weighted with selection of predictors at fixed $c = 0.2, 0.3, 0.4, 0.5$; $m = 2, 4, 6$ (light grey boxes) and with values of tuning parameters chosen by cross-validation ($\texttt{wNNSelect}_c$ and $\texttt{wNNSelect}_m$, dark grey boxes) based on L_2 distance in Autoregressive (left) and blockwise (right) correlation structure. Solid circles within boxes show the mean values.

the missing expression values of continuous variables (Hastie et al., 2013). The function `kNN` from the `R` package `VIM` (Templ et al., 2016) deals with data consisting of continuous, count, binary, and categorical variables. By default, it uses the Gower distance (Gower, 1971) for the computation of distances, but other metrics are also supported. Waljee et al. (2013) used this package in their studies on clinical data. The package `imputation` (Wong, 2013) imputes the missing data by applying the algorithm provided by Troyanskaya et al. (2001). However, the package does not always yield estimates for all missing data and by default replaces values that were not estimated with zeros.

Therefore, we chose the `impute` function from Bioconductor (Hastie et al., 2013) and the `kNN` function from package `VIM` (Templ et al., 2016) as benchmarks (shown as BIO and VIM, respectively, in Figure 3.4). Both methods use a predefined or fixed number of NNs (k). To obtain comparable results, we selected the number of NNs by using the cross-validation algorithm discussed in Section 3.2.3. For weighted imputation without the selection of variables, the tuning parameter λ was chosen by cross-validation, with m^* corresponding to approximately 10% of the data and $R = 10$. The values of the tuning parameters (λ, c) and (λ, m) were chosen by cross-validation on a double grid, with m^* corresponding to 10% and $R = 10$ for weighted imputation including the selection of predictors.

Figure 3.4 (left panel) shows the average MSIEs for the imputation methods under consideration. It is seen that for all imputation methods using all the predictors the performance is poor as the numbers of predictors increases. Although more information is available because more predictors are observed, the performance deteriorates. By contrast, methods that allow for the selection of predictors make use of the additional information. They perform better for large numbers of predictors and distinctly outperform both the benchmarks

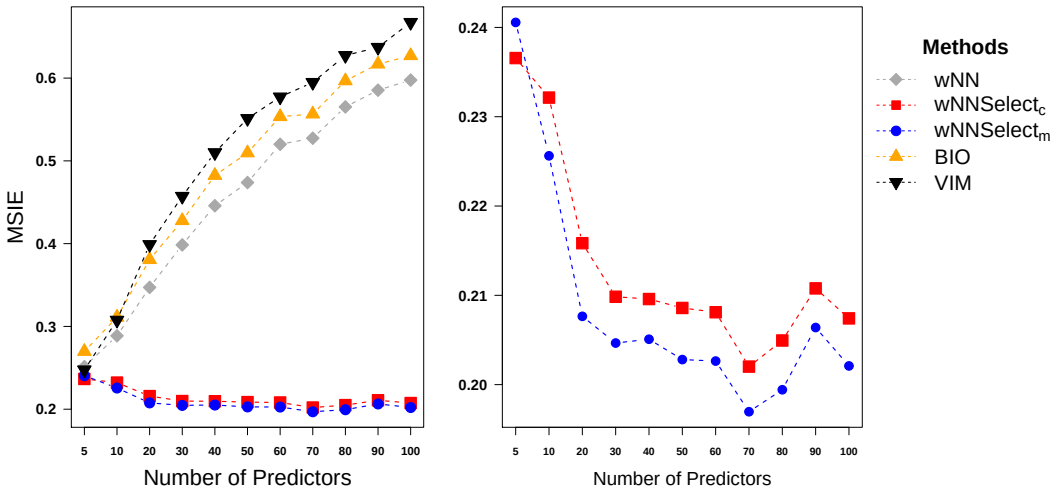


Figure 3.4.: Comparison of average MSIEs at different number of predictors, n=50 using AR(1) correlation matrix with $\rho = 0.9$

BIO and VIM and the weighted neighbors without selection (wNN) method. It is seen that the convex function (3.9) in particular yields good results in terms of the MSIEs even when the number of predictors is large.

Comparison of Weighted Imputation and Benchmarks

In the following, the performances of the new methods (`wNN` and `wNNSelect`) and the benchmark methods are investigated more systematically. We consider $S = 200$ samples of size $n = 50, 100$ from $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the correlation matrix with a blockwise or an AR structure.

For the blockwise correlation with $n = 50$ (setting 1), the predictors were again given in three blocks of 10 predictors each, that is, the $p \times 1$ random vector $\mathbf{X} = [\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}]$ has components $\mathbf{X}^{(1)} = [X_1, \ldots, X_{10}]$, $\mathbf{X}^{(2)} = [X_{11}, \ldots, X_{20}]$, and $\mathbf{X}^{(3)} = [X_{21}, \ldots, X_{30}]$. The variables within each block were strongly correlated with $\rho_w = 0.90$, but nearly uncorrelated with the variables in the other blocks ($\rho_b = 0.10$, for $i, j = 1, 2, 3, \forall i \neq j$). Similarly, for $n = 100$ (setting 2), the predictors were chosen in three blocks of 15 predictors each, that is, the $p \times 1$ random vector $\mathbf{X} = [\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \mathbf{X}^{(3)}]$ has comonents $\mathbf{X}^{(1)} = [X_1, \ldots, X_{15}]$, $\mathbf{X}^{(2)} = [X_{16}, \ldots, X_{30}]$, and $\mathbf{X}^{(3)} = [X_{31}, \ldots, X_{45}]$. In settings 3 and 4, the AR type $\mathbf{\Sigma}$ is of order one, with $\rho = 0.9$.

The simulation settings are as follows:

Setting	Structure	n	p	Correlation
1	Blockwise	50	30	ρ_w=0.9, ρ_b=0.1
2		100	45	
3	Autoregressive	50	30	ρ=0.9
4		100	40	

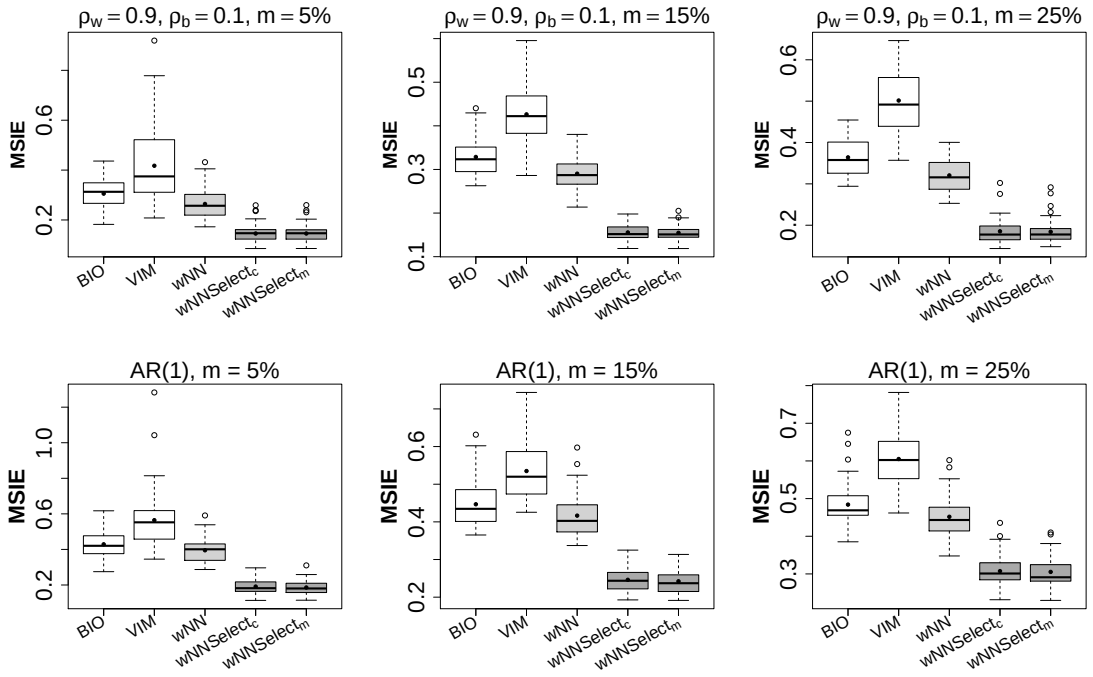


Figure 3.5.: Comparison of `wNNSelect` (dark grey boxes), `wNN` (light grey boxes) imputation using L_2 metric, and two benchmarks (white boxes); `impute` from Bioconductor (shown as BIO) & `VIM` (shown as VIM) using Blockwise (upper panel) and Autoregressive (lower panel) correlation structure.

The percentage of missing values in each data setting was set to 1%, 5%, 10%, 20%, 25%, and 30% of the total values. These values were generated completely at random (MCAR). The tuning parameters for the proposed weighting function `wNN` and `wNNSelect` were chosen by cross-validation with m^*, corresponding to approximately 10% of the sample size. For the benchmark methods, the number of NN (k) was chosen based on the same cross-validation algorithm.

Figure 3.5 compares the proposed weighted imputation methods with the existing methods under settings 1 and 3, with 5%, 15%, and 25% missing data. For the blockwise correlation structure (Fig. 3.5: upper panel for data setting 1), the weighting functions with L_2 metric and selection (shown as $\texttt{wNNSelect}_c$ and $\texttt{wNNSelect}_m$) and without the selection of predictors (shown as wNN) yield MSIEs that are smaller than the benchmarks (shown as BIO and VIM). The best MSIE is obtained by the weighted selection of predictors (shown

as $\texttt{wNNSelect}_c$ and $\texttt{wNNSelect}_m$). The convex functions (linear (3.8) and power (3.9)) used for the selection of predictors provide nearly the same MSIEs. For the AR correlation structure (Fig. 3.5: lower panel for data setting 3), the best MSIE is obtained near 0.1 (5% missing), near 0.2 (15% missing), and near 0.3 (25% missing). Therefore, for the AR correlation the average MSIE increases as the number of missing values increases. Similar results were obtained for settings 2 and 4 for other percentages of missing values, (see Table 3.2). In each data setting, the smallest MSIE value is shown in boldface.

As seen from Table 3.2, for the blockwise correlation with $n = 50$ (upper left panel) and 1% missing data the smallest MSIE is obtained using the weighted selection of predictors $\texttt{wNNSelect}_m$. The resulting value (0.1325) is much smaller than those obtained for existing NN imputation methods (0.2814 by `BIO` and 0.3399 by `VIM`). It is also apparent that the selection of dimension (`wNNSelect`) reduces the MSIEs when only the relevant information from selected predictors is used. The results for $n = 100$ are similar (Table 3.2: lower left panel). There is little difference concerning the alternative convex functions used in the weighting procedure for the blockwise structure with 1% to 20% missing data. However, when the percentage of missing data is greater than 20% of the data, the weighted function ($\texttt{wNNSelect}_m$) with the power function (3.9) yields the best MSIEs for both correlation structures.

In the case of an autoregressive correlation with $n = 50$ (Table 3.2: upper right panel), the weighted selection of predictors with convex function (3.9), denoted by $\texttt{wNNSelect}_m$, provides smaller MSIEs (for example, 0.1645 for 1% and 0.3409 for 30% missing values). Both of these values are much smaller than those obtained for `wNN` and the benchmark imputation methods. Moreover, the best MSIE is that resulting from the method with the selection of predictors when using the convex function (3.9), namely, $\texttt{wNNSelect}_m$. Similar results are found for $n = 100$ (Table 3.2: lower right panel).

A comparative view of the performance for the different levels of missing percentages for blockwise (lower panel) and AR (upper panel) correlations is shown in Figure 3.6. To make them comparable, all the plots were drawn on the same scale. It can be seen that the MSIEs become larger as the percentages of the missing values also become larger. The curve is flatter when the data are generated from a blockwise correlation structure. Overall, the proposed weighting functions distinctly outperform the available methods for all data settings; this is particularly the case for the methods $\texttt{wNNSelect}_c$ and $\texttt{wNNSelect}_m$.

To determine whether the performance of the weighted NNs approach suffers when the data are not drawn from a normal distribution, we consider a small simulation study with an alternative distribution. Figure 3.7 shows the results of data drawn from a multivariate t-distribution. The AR (1) correlation with $\rho = 0.9$ and the blockwise correlation with ρ_b=0.9, ρ_w=0.1 for $n = 50$ and $p = 30$ were used. Only the results of $S = 200$ samples with 5% MCAR values are shown. The weighted NN method can be seen to dominates the other procedures also in this case.

All the above simulation results were generated under the assumption that the data are MCAR. However, as suggested by a reviewer, it is also interesting to investigate the per-

Table 3.2.: Comparison of MSIE for weighted L_2 NN imputation with benchmarks

		Blockwise Correlation					**AR(1) Correlation**				
		weighted			Benchmark		*weighted*			Benchmark	
n	missing	`wNN`	`wNNSelect`$_c$	`wNNSelect`$_m$	*BIO*	*VIM*	`wNN`	`wNNSelect`$_c$	`wNNSelect`$_m$	*BIO*	*VIM*
50	1%	0.2534	0.1660	**0.1325**	0.2814	0.3399	0.3533	**0.1645**	0.1653	0.3931	0.4283
	5%	0.2646	0.1465	**0.1465**	0.3027	0.3450	0.3939	0.1869	**0.1842**	0.4260	0.4428
	10%	0.2759	0.1542	**0.1542**	0.3123	0.3701	0.4060	0.2164	**0.2108**	0.4336	0.4672
	15%	0.2876	0.1545	**0.1544**	0.3274	0.4056	0.4177	0.2445	**0.2411**	0.4519	0.4920
	20%	0.2920	0.1619	**0.1616**	0.3335	0.4392	0.4353	0.2766	**0.2703**	0.4698	0.5201
	25%	0.3122	0.1792	**0.1789**	0.3586	0.4791	0.4554	0.3118	**0.3086**	0.4886	0.5462
	30%	0.3229	**0.1952**	0.1961	0.3695	0.5061	0.4689	0.3445	**0.3409**	0.5056	0.5762
100	1%	0.2046	**0.1275**	0.1277	0.2338	0.2533	0.3734	**0.1515**	0.1518	0.3862	0.4197
	5%	0.2067	**0.1225**	0.1228	0.2301	0.2761	0.3867	0.1678	**0.1650**	0.4077	0.4434
	10%	0.2119	**0.1259**	0.1261	0.2377	0.3051	0.3869	0.1866	**0.1823**	0.4094	0.4774
	15%	0.2181	**0.1309**	0.1310	0.2463	0.3371	0.4024	0.2119	**0.2074**	0.4242	0.5032
	20%	0.2239	**0.1332**	0.1333	0.2521	0.3735	0.4116	0.2338	**0.2298**	0.4357	0.5342
	25%	0.2344	0.1378	**0.1378**	0.2643	0.4140	0.4255	0.2649	**0.2615**	0.4505	0.5653
	30%	0.2416	**0.1434**	0.1435	0.2747	0.4560	0.4395	0.3018	**0.2981**	0.4640	0.6057

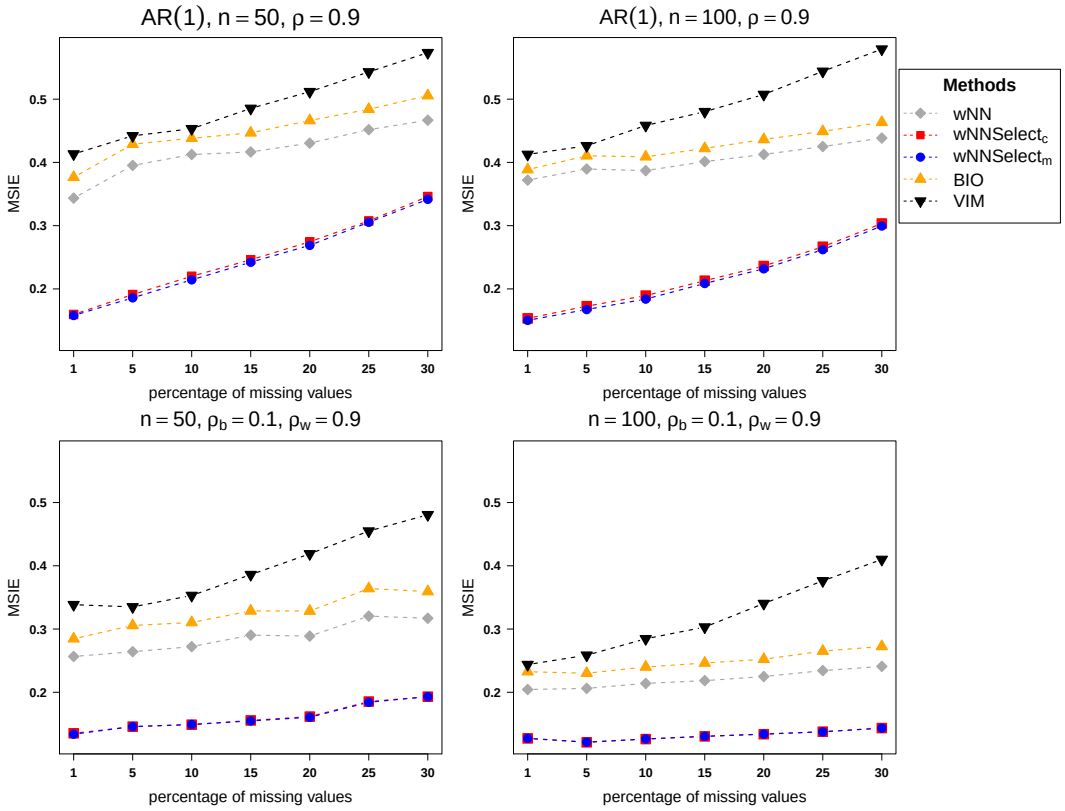


Figure 3.6.: Comparison of `wNNSelect` at different percentages of missing data in autoregressive (upper panel) and blockwise (lower panel) correlation structure. The tuning parameters (λ, c) and (λ, m) selected by cross-validation.

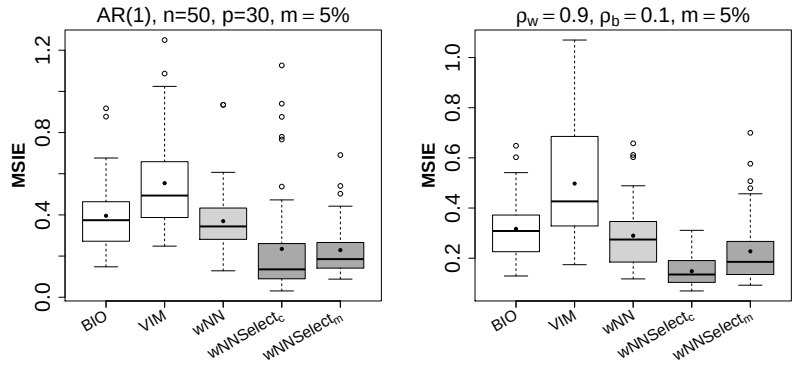


Figure 3.7.: Boxplots of MSIEs, $S = 200$ samples were drawn from multivariate t-distribution with 3 degrees of freedom using autoregressive (right) and blockwise (left) correlation structures. The tuning parameters (λ, c) and (λ, m) for weighted NN approaches were selected by cross-validation.

formance of the proposed approach for non-MCAR data. We consider a small simulation scenario in which the assumption of MCAR is violated. Figure 3.8 shows the results for $S = 200$ samples with $n = 50$ and $p = 30$ drawn from a multivariate normal distri-

bution with $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the correlation matrix of the blockwise or AR structure. We used the AR(1) correlation with $\rho = 0.9$ and the blockwise correlation with ρ_w=0.9, ρ_b=0.1. The missing values in variable x_i were generated by the logistic model $\pi_{(x_i)}(x_j) = \exp(-1.5 + 0.8x_j)/[1 + \exp(-1.5 + 0.8x_j)]$ for $i \neq j$. The percentage of missing values generated by the model was 11.80%. The tuning parameters (λ, c) and (λ, m) for weighted NN approaches were selected by cross-validation based on 5% randomly chosen observations. Figure 3.8 shows the dominance of the weighted procedure with selection also when the MCAR assumption is violated.

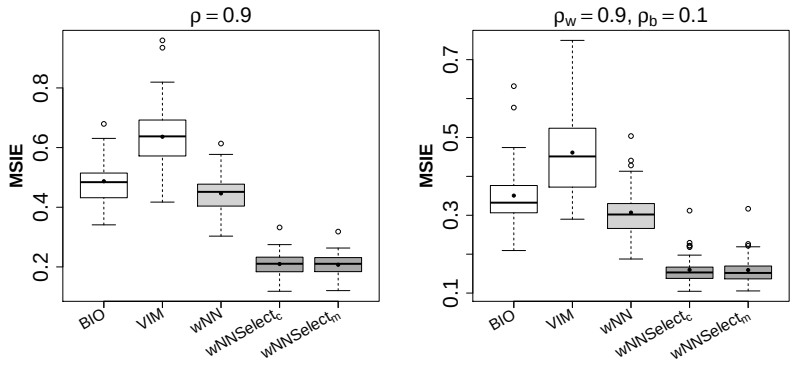


Figure 3.8.: Boxplots of MSIEs for MAR missing pattern, $S = 200$ samples were drawn from multivariate normal distribution using autoregressive (right) and blockwise (left) correlation structures.

Imputation methods are typically used as a first step to obtain complete data. The data are then used to investigate underlying structures. Here we provide the results of another small simulation study in which we asked whether the improvement achieved with the new methods carries over to the estimation of regression parameters. Thus, we fitted a linear regression model $\mathbf{y} = \mathbf{X}\beta + \epsilon$, where $\mathbf{X}$ is a $n \times p$ data matrix in which the missing values have been imputed. We used $\mathbf{X}$ drawn from $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ is the correlation matrix of blockwise or AR structure. The imputed data matrices were the same as those used in the previous simulation study (Table 3.2), with settings 1, 2 ($n = 50, 100$ and $p = 45$ with the blockwise correlation) and settings 3, 4 ($n = 50, 100$ and $p = 40$ with the AR correlation). The components of the parameter vector β were chosen from a uniform distribution, $unif(-3, 3)$, and the error term from a standard normal distribution. We computed the mean squared errors (MSE) of the estimated regression coefficients by using $\text{MSE}(\hat{\beta}) = \sum_S (\hat{\beta} - \beta)^2 / S$.

The average MSEs are shown in Table 3.3, with the standard errors shown in parentheses. The results are given only for $n = 100$ and missing value proportions of $5\%, 15\%, and 25\%$. The minimal value in each setting is shown in boldface. As seen from the table, in all the settings, the weighted approach provides distinctly smaller MSEs for $\hat{\beta}$, especially when dimensions are selected.

Table 3.3.: MSE($\hat{\beta}$) and their standard errors for NN imputation methods

	Autoregressive				
Missing (%)	`wNN`	$\texttt{wNNSelect}_c$	$\texttt{wNNSelect}_m$	BIO	VIM
5	0.5633	**0.3480**	0.3494	0.5917	0.6613
	(0.1856)	(0.1027)	(0.1039)	(0.1735)	(0.2091)
15	1.1145	0.8108	**0.7977**	1.1492	1.3829
	(0.3207)	(0.2348)	(0.2309)	(0.2938)	(0.3951)
25	1.4919	1.2800	**1.2769**	1.5652	1.7576
	(0.4221)	(0.3801)	(0.3881)	(0.3891)	(0.4444)
	Blockwise				
Missing (%)	`wNN`	$\texttt{wNNSelect}_c$	$\texttt{wNNSelect}_m$	BIO	VIM
5	0.4567	0.3739	**0.3725**	0.4838	0.5865
	(0.1332)	(0.0960)	(0.0961)	(0.1413)	(0.1880)
15	0.9692	0.7440	**0.7421**	1.0430	1.2332
	(0.2551)	(0.2213)	(0.2202)	(0.3319)	(0.3711)
25	1.4954	**1.2634**	1.2683	1.5667	1.8762
	(0.4898)	(0.3813)	(0.3860)	(0.3583)	(0.4353)

3.4. Case Studies

In this section, we use several real data sets to compare imputation methods. Two of the data sets were from genetics and two were from non-genetic studies. The complete data sets, without any missing values, were used so as to allow a comparison of the results.

3.4.1. Gene Expression Data

In particular gene expression data often contain missing values. Imputation of missing values in gene expression data was considered in several studies, see Brock et al. (2008), Wasito and Mirkin (2005), Troyanskaya et al. (2003), Nguyen et al. (2004), Hastie et al. (1999), and Dudoit et al. (2002). We use gene expression data on two different types of human tumor namely lymphomas and leukemia. The data sets can be downloaded from the website http://www.gems-system.org/.

DLBCL Data

The gene expression data on lymphomas contain data from 77 patients and 6,817 genes. The two groups in the study were patients with *diffuse large B-cell lymphomas* (DLBCLs; n=58 patients) and *follicular lymphomas* (FLs; n=19 patients). Data from the 100 most significant genes, identified by applying the `samr` package (Tibshirani et al., 2011) from `R`

(R Core Team, 2013), were used. This package provides a tool to identify differentially expressed genes.

Leukemia Data

The second data set contains data from three types of leukemia: acute myelogenous leukemia (AML), acute lympboblastic leukemia (ALL) B-cell, and ALL T-cell. The complete data consists describes gene expression with respect to 5,328 variables of 72 patients. In our analysis, the data from all 72 patients for the 100 most significant genes, identified by applying the `R` package `samr` (Tibshirani et al., 2011), were used.

3.4.2. Non-gene Expression Data

LSVT Voice Rehabilitation Data Set

Lee Silverman Voice Treatment (LSVT) Global, a company specializing in voice rehabilitation, assists people with Parkinson's disease (PD). The data set was originally collected to determine the most parsimonious feature subset that aids in the prediction of the binary response. The data are composed of a range of biomedical speech signal- processing algorithms from 14 people diagnosed with PD and undergoing LSVT. The original study used 310 algorithms (predictors) to characterize 126 speech signals (samples). The response variable is binary, acceptable vs. unacceptable phonation during rehabilitation. More information on the data can be found in Tsanas et al. (2014). The data can be downloaded from the UCI Machine Learning Repository.

LIBRAS Movement Database

The data set contains 15 classes of 24 instances each, with each class referring to a hand movement type in LIBRAS. The hand movement is represented as a bidimensional curve performed by the hand during a certain period of time. The curves were obtained from videos of hand movements, with using the Libras LIBRAS performances from four different people during two sessions. Each video corresponds to only one hand movement and is roughly 7 s long. The total data consists of 360 instances of 90 numeric attributes. More information on the data can be found in Dias et al. (2009). The data set is available on the UCI Machine Learning Repository.

All the variables were standardized before processing. The NN imputation techniques were applied to these four data sets by artificially setting 5% of the observations as MCAR). The missing values were imputed using weighted NNs with the selection of variables (`wNNSelect`) with the tuning parameters chosen by cross-validation. For the benchmark methods, the number of NNs was also selected by cross-validation. The procedure was repeated 10 times for each of the methods. The resulting averaged MSIEs are shown in Table 3.4. The

smallest MSIE for each data set is shown in boldface, ; standard errors are enclosed in parentheses.

In all of these case studies, the minimum value of the average MSIE was obtained using one of the new imputation methods. For the DLBCL data, the minimum, 0.5993, was obtained for the L_1 metric with the convex function (3.9) shown as $\texttt{wNNSelect}_m$. For the `impute` function from Bioconductor , the MSIE was 0.72679;, for the `VIM` procedure it was 0.7889. The results for the leukemia data were similar, with 0.7279 as the minimal average MSIE for $\texttt{wNNSelect}_m$ based on the L_2 metric. For LVST and the LIBRAS movement data sets, the smallest MSIE was obtained for $\texttt{wNNSelect}_m$, using the L_2 metric. As shown in Table 3.4, for all of the data sets the proposed imputation procedure yields better results than the Bioconductor and the `VIM` packages.

Table 3.4.: Average MSIE and their standard errors for NN imputation methods using real data sets

	L_1 **metric**		L_2 **metric**		**Benchmark**	
DATA	$\texttt{wNNSelect}_c$	$\texttt{wNNSelect}_m$	$\texttt{wNNSelect}_c$	$\texttt{wNNSelect}_m$	**BIO**	**VIM**
DLBCL	0.6424	**0.5993**	0.6504	0.6078	0.7267	0.7889
	(0.0953)	(0.0891)	(0.0984)	(0.0912)	(0.1009)	(0.1405)
Leukemia	0.7620	0.7461	0.7620	**0.7292**	0.8709	0.8272
	(0.0887)	(0.0841)	(0.0816)	(0.0883)	(0.0966)	(0.1363)
LVST	0.4401	0.4427	0.4142	**0.4087**	0.5821	0.6662
	(0.0912)	(0.0979)	(0.0882)	(0.0898)	(0.0875)	(0.1405)
LIBRAS	0.0799	0.0752	0.0353	**0.0350**	0.2903	0.5966
	(0.0206)	(0.0201)	(0.0069)	(0.0068)	(0.0375)	(0.0794)

3.5. Concluding Remarks

The main objective of this study was to introduce improved NN procedures for the imputation of missing values. The simulation results showed that the proposed *weighted imputation estimate* performs better than the fixed or un-weighted approach. The use of kernel functions to generate weights decreases the imputation error. Simulation results, not presented here, suggest that the Gaussian kernel yields smaller MSIEs than other kernel functions. To cope with the problem of high-dimensional data, we proposed an extended imputation method with a weighted selection of predictors. Especially for highly correlated data, the proposed NN imputation procedure shows promising results, also when there is a high proportion of missing data. Both the simulation studies and the real data sets confirmed these results. As demonstrated in the simulations, the performance depends on the correlations between covariates. If the correlation is low or absent, the NN imputation methods will not perform better than simple mean value imputation.

Although the methods improve traditional NN-based imputation methods, they share the drawbacks of all NN methods. These include the fact that the valid inference obtained using some of the multiple imputation methods is not available; this will be a topic of future research. An advantage of the methods is that they also can be used in high-dimensional settings in which the number of covariates is very large compared to the number of observations. In these settings, a complete case analysis, which also yields consistent estimates, is typically infeasible because too many cases have to be omitted, rendering inefficient estimates. This was the situation in the real data sets used herein to illustrate the methods.

Several choices have to be made when using the extended NN methods. One refers to the definition of the distance. We compared the L_1 and L_2 metrics and found that the performance of the L_2 metric was typically slightly better than that of the L_1 metric. The L_1 metric might perform better if many outliers are in the data but we did not find a distinct dominance, not even for distributions that tend to produce more extreme values (results not shown). Moreover, especially for the real data, in which the true distribution is was unknown, the L_2 metric yielded distinctly better results than the L_1 metric. We considered two functions that transform the correlations into weights, one depending on the tuning parameter $c \in [0, 1)$ and the other depending on natural numbers m. The performance was better when m (Table 3.2) was used. Although the version with the tuning parameter $c \in [0, 1)$ was slightly superior for the t-distribution (Figure 3.7), the version with the tuning parameter m dominated consistently in the real data sets (Table 3.4)

4. Missing Value Imputation for Gene Expression Data by Tailored Nearest Neighbors

Abstract

High dimensional data like gene expression and RNA-sequences often contain missing values. The subsequent analysis and results based on these incomplete data can suffer strongly from the presence of these missing values. Several approaches to imputation of missing values in gene expression data have been developed but the task is difficult due to the high-dimensionality (number of genes) of the data. Here an imputation procedure is proposed that uses weighted nearest neighbors. Instead of using nearest neighbors defined by a distance that includes all genes the distance is computed for genes that are apt to contribute to the accuracy of imputed values. The method aims at avoiding the curse of dimensionality, which typically occurs if local methods as nearest neighbors are applied in high-dimensional settings. The proposed weighted nearest neighbors algorithm is compared to existing missing value imputation techniques like mean imputation, KNNimpute and the recently proposed imputation by random forests. We use RNA-sequence and microarray data from studies on human cancer to compare the performance of the methods. The results from simulations as well as real studies show that the weighted distance procedure can successfully handle missing values for high-dimensional data structures where the number of predictors is larger than the number of samples. The method typically outperforms the considered competitors.

Keyword: High-dimensional data, Missing values, Weighted nearest neighbors, Gene expression data

4.1. Introduction

DNA microarray data are important in a wide range of application areas such as biology, genetics and medicine. They typically come in the form of large matrices with missing values affecting most of the genes. Missing values can affect up to 90% of the genes (Ouyang et al., 2004). The main causes for the occurrence of missing values are artifacts on the microarray, dust or scratches on the slide, hybridization failures, image noise and corruption, and insufficient resolution. Additionally, the robotic methods used to create data can be responsible. Standard statistical methods are designed for the analysis of complete data matrices, but fail if the data matrix contains missing observations. Therefore, methods for data dimension reduction and clustering techniques tend to suffer (Sehgal et al., 2004). One option to address the problem of missing values is to simply remove the missing observations by discarding the corresponding cases. However, this procedure entails loss of valuable information and can lead to a selection bias, in particular if the number of missing values is large (Tuikkala et al., 2008). The situation calls for methods to impute the missing values before using standard statistical analysis tools.

A number of methods to estimate or impute missing values have been proposed in literature, and a continuous effort is in progress to develop improved imputation methods. The simplest approach consists of replacing the missing gene expression values with zeros or mean/average values (Alizadeh et al., 2000). A distinct disadvantage of this method is that the information on the correlation among genes is ignored. It has been demonstrated that imputation methods perform better than both the simple deletion of whole observations and the replacement of the missing data by zeros (ZeroImpute) or average values (MeanImpute), see, for example, Troyanskaya et al. (2001). Stekhoven and Bühlmann (2012) proposed a random forests approach to impute missing values in mixed type data matrices. Their study showed that missForest can successfully handle missing values for continuous as well as categorical and mixed data with complex interactions and non-linear relations. More recently, Städler and Bühlmann (2014) presented the imputation method Missingness Pattern Alternating Imputation and l_1-penalty (MissPALasso), which assumes that data matrices are generated by a normal distribution. The approach uses an EM-type algorithm and a lasso penalty to impute missing values.

The use of the information provided by nearest neighbors for the imputation of missing values has been proposed by Troyanskaya et al. (2001). The KNN imputation approach not only outperforms the previously used methods for completing data matrices with zeros or means, but also provides better accuracy than singular value decomposition (SVD) imputation approaches. Various versions of this concept have been developed (see for example Kim et al. (2004); Brás and Menezes (2007)). Bø et al. (2004) proposed specialized procedures that utilize the correlation between genes and between arrays based on the principle of least squares and k-nearest neighbors methods. Dudoit et al. (2002) used nearest neighbor methods that use the information of correlation among genes to impute the missing gene expression data in a tumor classification study. For protein expression data, Jung et al. (2006) applied KNN methods and estimated the missing values by using the mean, the weighted mean and the median of the nearest neighbors. Their studies show

that the mean is better than the median and the weighted mean. Moreover, in comparison of distance measures, the Euclidean distance performed slightly better than the Mahalanobis and the Chebyshev distances (see also Jung et al., 2005).

More recently, Tutz and Ramzan (2015) proposed an improved imputation method that uses nearest neighbors computed from selected predictors. The selection of predictors is linked to the correlation between the predictor and the variable that has to be imputed. It was shown in simulation studies that the method performs well in high-dimensional settings. The following paper will focus on a version of the corresponding `wNNSelect` algorithm. The objective is to investigate its performance in data situations arising in genetics. Therefore, we use simulated data that mimic the structure found in real data sets, and use these real data sets themselves with with randomly generated missing data. In addition, it is investigated how this imputation method compares to the recently proposed random forest imputation method. These competitors were not considered by Tutz and Ramzan (2015).

The paper is organized as follows: Section 2 describes the details of the nearest neighbor approach, the cross-validation procedure used for the selection of tuning parameters and the competitors for imputation. The simulation results and application to the real gene expression data are included in Section 3. Finally, some concluding remarks are given in Section 4.

4.2. Materials and Methods

In this section, we describe the weighted nearest neighbors approach that uses the selected variables (genes) for imputation, and give a brief overview on alternative approaches to impute missing genes expression data.

4.2.1. Nearest Neighbors Approaches to Imputation

Given n microarray experiments, each of which contains the mRNA expressions of p genes, the data can be organized into an $n \times p$ matrix of gene expression values. Let x_{is} denote the expression value of gene (column) s in sample (row) i. In the following we refer to a particular gene with a missing value (MV) as the target gene. The set of genes with available information for the estimation of the MV of the target gene is considered as the set of candidate genes.

Let $\boldsymbol{X} = (x_{is})$ be the $n \times p$-dimensional data matrix, where x_{is} denotes the expression level of the ith sample on the sth gene. Let $\boldsymbol{O} = (o_{is})$ denote the corresponding $n \times p$ matrix of dummies with entries

$$o_{is} = \begin{cases} 1 & \text{if } x_{is} \text{ was observed} \\ 0 & \text{for missing value.} \end{cases}$$

For two rows of the data matrix, $\boldsymbol{x}_i^T = (x_{i1}, \ldots, x_{ip})$ and $\boldsymbol{x}_j^T = (x_{j1}, \ldots, x_{jp})$, the distances that define the nearest neighbors are computed by

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{1}{a_{ij}} \sum_{s=1}^{p} (x_{is} - x_{js})^2 I_{(o_{is}=1)} I_{(o_{js}=1)} \right)^{1/2}, \tag{4.1}$$

where $I(.)$ denotes the indicator function and $a_{ij} = \sum_{s=1}^{p} I(o_{is} = 1) I(o_{js} = 1)$ is the number of valid components in the computation of distances.

In nearest neighbor imputation, if observation x_{is} is to be imputed, one computes the k nearest neighbors of $\boldsymbol{x}_i$ denoted by $\boldsymbol{x}_{(1)}, \ldots, \boldsymbol{x}_{(k)}$. The imputed value is obtained by using the sth components of the nearest neighbors $\boldsymbol{x}_{(j)}^T = (x_{(j)1}, \ldots, x_{(j)p})$ by computing $\hat{x}_{is} = \sum_{j=1}^{k} x_{(j)s}/k$. Therefore, the missing value in the sth component of the observation vector $\boldsymbol{x}_i$ is replaced by the average of the corresponding values of the k nearest neighbors. In k-nearest neighbor imputation the number of the nearest neighbors, k, is a tuning parameter that can be chosen, for example, by cross validation.

A modified form of nearest neighbor imputation uses weights instead of a fixed number of nearest neighbors. The corresponding imputed value is given by

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\boldsymbol{x}_i, \boldsymbol{x}_{(j)}) x_{(j)s}, \tag{4.2}$$

where $w(.,.)$ is a weight function typically chosen as kernel functions, that is,

$$w(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{K\Big(d(\boldsymbol{x}_i, \boldsymbol{x}_j)/\lambda\Big)}{\sum_l K(d\Big(\boldsymbol{x}_i, \boldsymbol{x}_j)/\lambda\Big)}$$

with kernel function $K(.,.)$. In weighted imputation, the tuning parameter is the window width λ, which replaces the number of nearest neighbors.

4.2.2. Nearest Neighbor Based on Selected Genes

In high-dimensional settings, the computation of distances over the whole feature space tends to suffer from the curse of dimensionality. The strategy that is proposed here is to select the features that actually carry information on the missing values. It has already been shown by Troyanskaya et al. (2001) that the correlation between gene expression profiles plays a major role in the imputation of missing values. Variables can contribute to the accuracy of the estimated value only if they are correlated to the variable that is to be imputed. If there is no correlation between these two components, the variable contains no information and the nearest neighbor imputation will not perform better than simple mean imputation.

The `wNNSelect` algorithm proposed here explicitly links the selection of genes or features used in the imputation method to the correlation between the variable that is used to impute and the target variable, that is, the variable that is missing and is to be imputed. The link is obtained by replacing the simple distance function by a weighted distance, where the weights are determined through the strength of correlation. More precisely, when imputing a value x_{is}, the `wNNSelect` algorithm uses the distance function

$$d_{\text{Sel}}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(\frac{1}{a_{ij}} \sum_{l=1}^{p} |x_{il} - x_{jl}|^q I_{(o_{il}=1)} I_{(o_{jl}=1)} w_{\text{corr}}(r) \right)^{1/q} \tag{4.3}$$

The correlation driven weight, which is included in (4.3), has the form $w_{\text{corr}}(r_{sl}) = |r_{sl}|^{\omega}$, where r_{sl} is the empirical correlation between gene s and gene l. Therefore, if the lth variable is uncorrelated with the target variable s the variable is not used when computing distances. If the correlation is strong, no matter positive or negative, the variable contributes to the computation of the distance. The `wNNSelect` procedure can be described as follows:

1. Locate a missing value x_{is} in sample i and gene s.

2. Compute d_{Sel} of expression vectors $\boldsymbol{x}_i, \boldsymbol{x}_j$ using (4.3).

3. Rank samples (rows) based on d_{Sel}.

4. Calculate corresponding weights based on kernel function $w(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

5. Compute missing value $\hat{x}_{is}$ using (4.2).

6. Seek the next missing value in $\boldsymbol{X}$ and repeat steps (2-5) until all missing values in $\mathbf{X}$ have been imputed.

The power ω in the function $w_{\text{corr}}(r) = |r|^{\omega}$ is considered an additional parameter. Therefore, one has two tuning parameters, λ, which determines the weighting of nearest neighbors, and ω, which determines the impact of the correlation strength on the computation of distances. In simulation studies we found that in the computation of weights (equation 4.2) the Gaussian kernel is to be preferred, and in the definition of the distance d_{Sel} (equation 4.3) the value $q = 2$ yields better results. Moreover, $w_{\text{corr}}(r)$ is based on the correlation matrix which requires a complete data matrix. One can use an initial unweighted five nearest neighbor imputation or mean imputation to get complete data set. Alternatively, one can use the complete data only, when computing the correlation between components, i.e. using only $n_{sl} = \sum_{=1}^{n} I(o_{is} = 1)I(o_{il} = 1)$ pairs where n_{sl} is the number of valid components in the computation of correlation. It should be mentioned that we used the standard estimator for the correlation. There are alternatives available, for example, the shrinkage estimator of Schäfer et al. (2005).

4.2.3. Choice of Tuning Parameters

The weighted nearest neighbors imputation requires the tuning parameter λ and ω. In the following the cross-validation algorithm used for the selection of the optimal values of these parameters is presented. The main idea is to randomly delete some values that were actually observed and impute these values to obtain a measure of imputation error. The process is repeated for various values of the tuning parameters in order to find the values that have minimal imputation error. More specifically, the cross-validation procedure is described in Algorithm 4.1.

Algorithm 4.1 Cross-validation for `wNNSelect`

Require: $\mathbf{X}$ an $n \times p$ matrix, number of validation sets t, range of suitable values for tuning parameters $\mathcal{L}$ and $\mathcal{W}$
  $\mathbf{X}^{\text{cv}} \leftarrow$ initial imputation using unweighted 5-nearest neighbors
  for $t = 1, \ldots, \mathcal{T}$ **do**
    $\mathbf{X}^{\text{cv}}_{\text{miss,t}} \leftarrow$ artificially introduce missing values to $\mathbf{X}^{cv}$
    for $\omega \in \mathcal{W}$ **do**
      for $\lambda \in \mathcal{L}$ **do**
        $\mathbf{X}^{\text{cv}}_{\texttt{wNNSelect},t} \leftarrow$ imputation of $\mathbf{X}^{\text{cv}}_{\text{miss,t}}$ using `wNNSelect` procedure
        $\psi_{(\lambda,\omega),t} \leftarrow$ imputation error of `wNNSelect` procedure for λ & ω
      end for
    end for
    Determine $(\lambda, \omega)_{\text{best}} \leftarrow$ argmin $\frac{1}{T}\sum_{t=1}^{T} \psi_{(\lambda,\omega),t}$
  end for
  $\mathbf{X}^{\text{imp}} \leftarrow$ `wNNSelect` imputation of $\mathbf{X}$ using $(\lambda, \omega)_{\text{best}}$

Let $\psi_{(\lambda,m)}$ be a measure of imputation error for specific values of λ and ω. One can use, for example, the mean of the absolute differences between imputed and true values (MAE), or the mean of the squared imputation errors (MSIE). We use the MSIE to obtain the optimal values with $\lambda \in \mathcal{L} = (0, 1]$, $m \in \mathcal{M} = \{2, 4, 6\}$ and a split into five validation sets.

4.2.4. Overview of Competing Methods

The performance of the `wNNSelect` imputation method is compared to existing methods like Zero imputation, mean substitution, KNN imputation and imputation using random forests (RF). We describe these methods briefly in the following.

Zero and Mean imputation

The easiest method for filling missing values in gene expression data is to replace them by zeros or average values (Alizadeh et al., 2000). The replacement of missing values with

their corresponding mean values is beneficial only if the data have weak correlation (see for example, Feten et al., 2005). Therefore, the approach is less attractive but is included as a baseline procedure.

KNN Imputation

The KNN method estimates the missing data by selecting genes whose expression values are most similar to the gene of interest (the one with missing values). Let x_{is}, the expression for sample i and gene s, be a missing value. Then the neighbor or candidate genes are defined on the basis of a gene similarity or a distance measure, for example, the Euclidean distance or the Pearson correlation. The next step includes the estimation of the missing gene entry by using the available observed values of the selected nearest genes. The predicted value is the weighted average of the expression values of gene s in k nearest samples (Troyanskaya et al., 2001). To evaluate the performance of this method, the function `impute` from the Bioconductor package `impute` (Hastie et al., 2013) and the function `kNN` from the R package `VIM` (Templ et al., 2016) are used. The procedure requires the value of k (number of nearest neighbors) to be selected in advance. In our simulation studies, we used cross-validation for selecting the optimal values of k.

Imputation by Random Forests

Random forests were introduced by Breiman (2001) as an extension of classification and regression trees (CART). The approach uses bootstrap samples of the data to build various trees. It can accommodate not only mixed types of predictors with nonlinear and complex relationships, but also deal with high-dimensional data ($p \gg n$). A random subset of the variables is used as candidates at each split. To build the trees, bootstrap aggregation (bagging) as well as random variable selection can be used. A popular choice for the split size is to use the square root of the number of available predictors (Stekhoven and Bühlmann, 2012). The recently introduced *missForest* procedure uses the versatile and flexible random forests model to obtain an estimate of a missing value. It is an iterative procedure that uses initial imputations using mean imputation or another imputation method, and then improves the imputed data matrix on successive iterations. A random forest model is developed for each predictor with a missing value by using the rest of the predictors, and this model is used to estimate the missing value of that predictor. The imputed data matrix is updated, and the difference between previous and new imputation is assessed at the end of each iteration. The whole process is repeated until a specific criterion is met. In our evaluations we used the `R` package `missForest` (Stekhoven and Bühlmann, 2012).

4.3. Results and Discussion

4.3.1. Data

The investigation of the performance of various imputation methods is based on real data sets, which we used in two ways. In Section 4.3.2 we use the data structure found in these data sets to simulate data and missing values. In each setting, $S = 200$ samples are drawn from a multivariate normal distribution with the mean vectors and covariance matrices determined by the corresponding estimates for the real data sets. In Section 4.3.3, we use the real data sets themselves and investigate the performance of imputation methods based on some randomly deleted values.

RNA-sequence data typically consists of read counts obtained by mapping (alignment). Mapping of sequence reads of the reference genome is a fundamental step in the RNA-seq data analysis, for example, by counting the aligned sequence reads with HT-seq (Anders et al., 2014). Different mapping methods are available for this purpose e.g., the STAR-aligner, which is available in the software package STAR, (Dobin et al., 2013). It is important to distinguish between *zero count* and a *missing* value in RNA-Seq data. A transcript that is found in one observation by mapping to a reference genome but not found in another observation is seen as a *zero count* rather than a missing value.

The first data set we consider is taken from the ReCount repository (Frazee et al., 2011), and consists of RNA-Seq data of 12984 genes for 60 individuals from the HapMap cohort from Montgomery et al. (2010). The second data is taken from the same repository and was used by Pickrell et al. (2010). It consists of 12984 genes of 69 Nigerian HapMap individuals from a study of expression quantitative trait loci (eQTLs).

Filtering or removing irrelevant genes is a common practice before analyzing these types of data e.g. prior to cluster analysis or genetic network analysis (Tritchler et al., 2009). We used filtered data sets for the implementation of our methods. For each RNA-seq dataset, the evidence of differentially expressed transcripts was measured by the informative/non-informative (I/NI) value, which can be used to extract the transcripts with a desired significance level (I/NI threshold) (Klambauer et al., 2013). We used the I/NI threshold 0.1 and obtained 8.86 % (=1150) of the total 12984 genes for the Montgomery data and 9.2% (=1195) of the total 12984 genes for the Pickrell data.

The third data set was published and provided by Khan et al. (2001). In a sample of 63 individuals, the expression of genes in four types of small round blue cell tumours of childhood (SRBCT) were collected. The original sample contained 6567 clones, of which 3789 were known genes and 2778 were ESTs. These types were neuroblastoma (NB), rhabdomyosarcoma (RMS), Burkitt lymphoma, a subset of non-Hodgkin lymphoma (BL), and the Ewing family of tumours (EWS). Khan et al. (2001) used the sample red intensity (ri) as a criteria for filtering, and omitted all those genes for which the the sample ri was less than 20. The filtered data set contained 2308 gene expression profiles and is freely available from the supplementary website http://research.nhgri.nih.gov/microarray/Supplement/.

4.3.2. Simulation and Evaluation

In this section, we compare the imputation methods by using simulated data that are generated in accordance with the real gene expression data structures. First, we selected 300 genes out of the differentially expressed genes by using the *R* package `DEXUS` (Klambauer et al., 2013) to obtain simulation scenarios. Let μ and $\mathbf{\Sigma}$ denote the estimated mean vector and covariance matrix of the data respectively. Then, samples of sizes $n \in \{60, 69, 63\}$, which correspond to the sizes of the real data sets, and $p = 300$ predictors were drawn from a multivariate normal distribution with mean vector μ and covariance matrix $\mathbf{\Sigma}$. The simulation procedure was as follows: starting with the complete data matrix, we marked some values as missing completely at random with a probability π, used the imputation method to estimate the missing entries, and compared the imputed and the complete matrix by computing MSIEs. The procedure was performed $S = 200$ times for each missing probability π.

For the assessment of the performance of imputation methods, the MSIE is used, which is defined by

$$\text{MSIE} = \frac{1}{n^{miss}} \sum_{x_{is}:o_{is}=0} (x_{is} - \hat{x}_{is})^2, \tag{4.4}$$

where x_{is} is the true value in the complete data matrix, $\hat{x}_{is}$ is the imputed value and n^{miss} is the number of missing values in the data matrix.

The performance of the weighted nearest neighbors method is compared with other imputation approaches including KNN and random forests (RF). The random forest method was chosen as a competitor because recent studies have shown that it has smaller imputation error than other approaches, see Stekhoven and Bühlmann (2012) and Waljee et al. (2013).

When using the `missForest` method, one needs to specify the number of trees (n_{tree}) and the number of variables tried at each node (m_{try}). As suggested by one reviewer, we investigated the impact of the choice of these tuning parameters. For the number of trees we used $n_{tree} \in \{50, 100, 1000\}$, and for the number of variables tried at each node we used $m_{try} \in \{10, \sqrt{p} = 17, p/3 = 100, 150\}$. The choice was motivated by Breiman who suggested the use of $p/3$ for regression, and $\sqrt{p}$ for classification. The resulting boxplots of MSIEs are shown in Figure 4.1. For simplicity, only the results for the Montgomery data are shown because the other two data showed similar patterns. It is seen that for all values of m_{try} an increase of the value of n_{tree} has a minor effect on the MSIEs . To reduce computational effort we chose $n_{tree} = 100$ as recommended by Stekhoven and Bühlmann (2012). An increase of m_{try} definitely reduces the error, but for $m_{try} > p/3$ the improvement is negligible. In the final tables we show the results for the values $m_{try} \in \{\sqrt{p}, p/3\}$.

For KNN imputation, we use the `impute` function from Bioconductor and the `kNN` function from package `VIM` (shown as BIO and VIM in Figure 4.2). Since the method needs a prespecified value of k (number of nearest neighbors), we used cross validation to find the

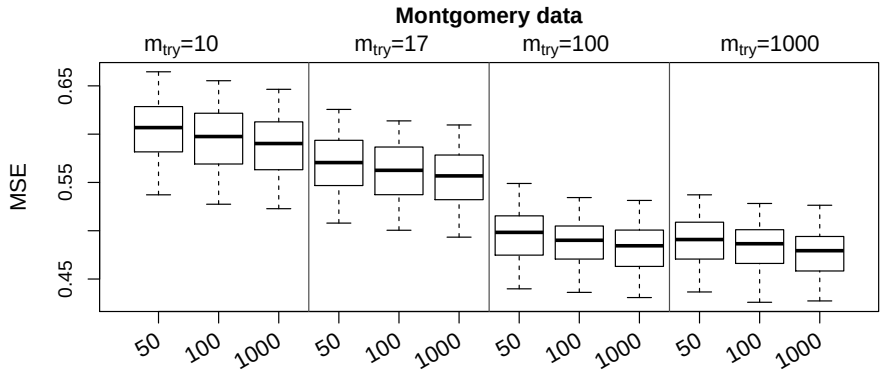


Figure 4.1.: Random Forest imputation: Comparison of MSIEs at different level of n_{tree} and m_{try} using simulated data with $\pi = 0.10$ missing values.

optimal k for all settings. Although the optimal parameters can also be chosen by using initial simulations, see, for example, Brock et al. (2008), we used cross-validation because it was also used for $wNNSelect$.

The weighted nearest neighbors method also requires the selection of a suitable kernel function for computing the weights in equation (4.2) and a value of q in the distance equation (4.3). The selection of the kernel function and q were investigated in detail by Tutz and Ramzan (2015). It was found that the Gaussian kernel typically performs better than other choices. We also investigated alternative kernel functions for gene expression data, and the results were similar. Therefore, throughout the simulations and the evaluation of methods in real data sets, we use the Gaussian kernel and $q = 2$ to compute distance in equation (4.3).

The simulation results for three data sets, with missing probabilities $\pi = 0.05$, 0.10, 0.15, 0.20, 0.25, are shown in Figure 4.2.

Each boxplot represents the imputation error (MSIEs) over 200 replications. From the results shown in Figure 4.2 it is evident that the weighted method (`wNNSelect`) has the smallest MSIE, followed by the random forests (RF) and the KNN imputation method.

Effect of Sample Size

In this section we examine the performance of `wNNSelect` when the sample size is very low as was recommended by one of the reviewers. We set $n = 20$, and implement the same imputation procedure as in the previous section. More specifically, we select a random sample of size $n = 20$ from the data and use its structure to obtain simulated data sets. The process is repeated five times and the average results are given. Figure 4.3 shows the MSIEs for $n = 20$ and $\pi = 0.10$. It is seen that also for small sample sizes the weighted imputation method outperforms the competitors. It is remarkable that `wNNSelect` yields

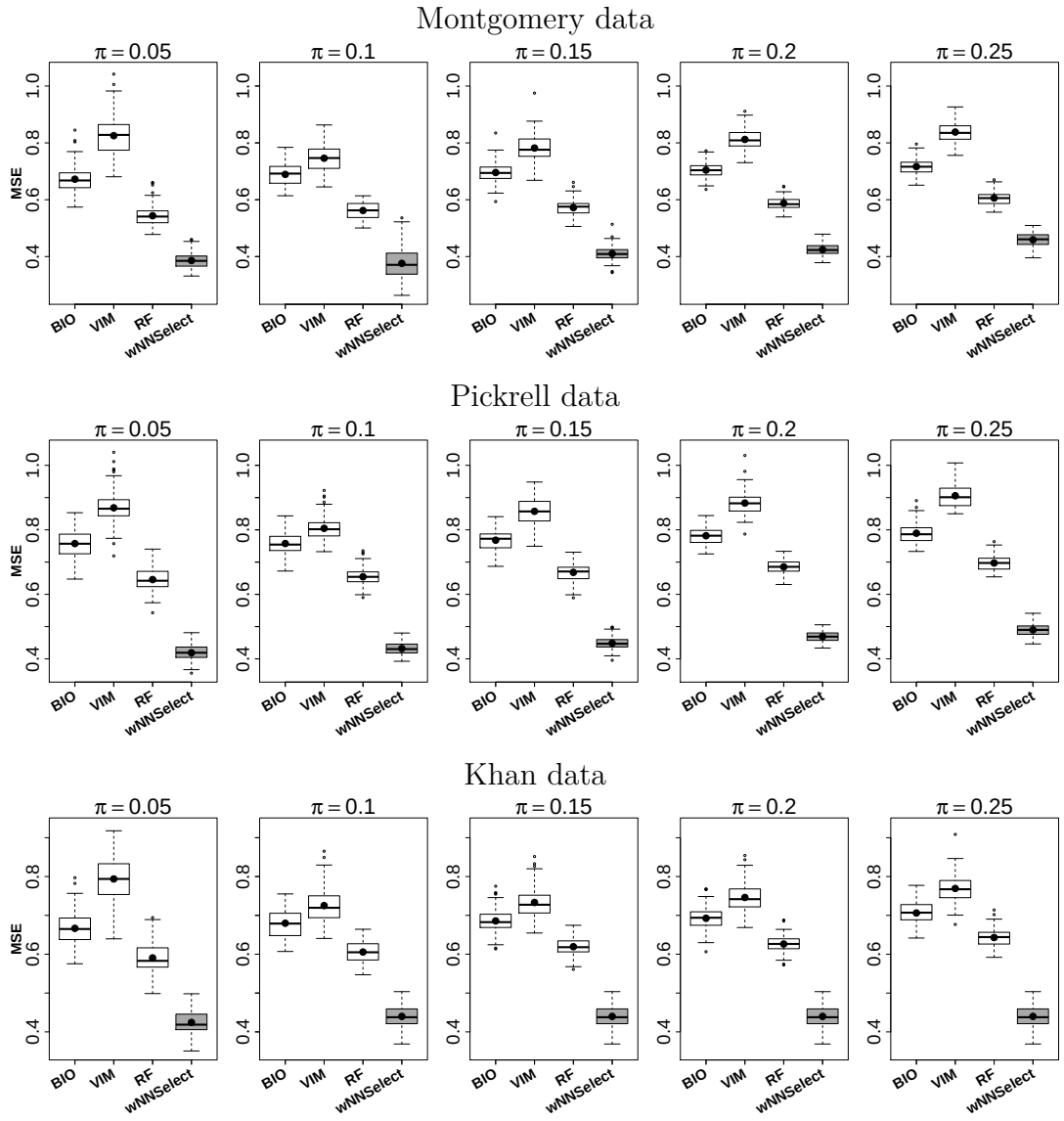


Figure 4.2.: Illustration of simulation study: Boxplots of MSIEs for different values of π using KNN (`BIO` and `VIM`), Random Forests (`RF`) and weighted imputation (`wNNSelect`) methods. Solid circles within boxes show mean values.

MSIEs that are not only smaller in magnitude but also have less variation. We show the results for only one setting because the settings with different probabilities yielded similar results.

4.3.3. Real Data Sets

The simulation results in Section 4.3.2 show that the `wNNSelect` method has the smallest MSIE in all the data settings. Now we apply this approach directly to the real gene

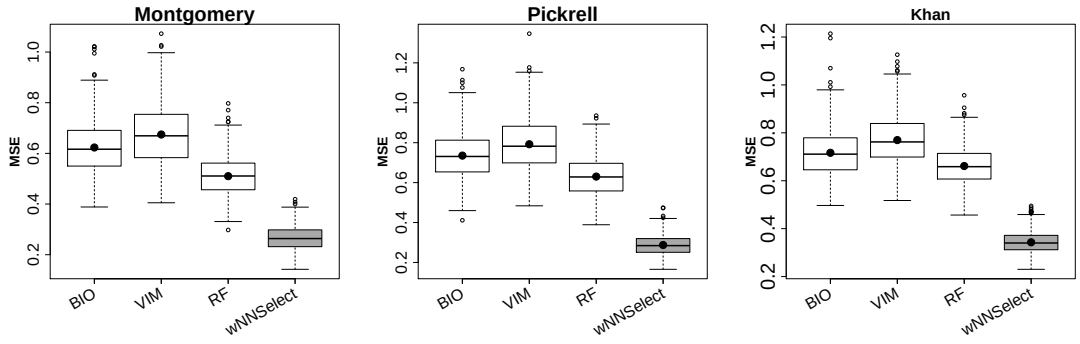


Figure 4.3.: Illustration of simulation study for $n = 20$: Boxplots of MSIEs for $\pi = 0.10$ using KNN (`BIO` and `VIM`), Random Forests (`RF`) and weighted imputation (`wNNSelect`) methods. Solid circles within boxes show mean values.

expression data and compare the performance with existing approaches. We use the complete gene expression data matrices, which have no missing values, so that the performance of the imputation methods can be compared by using the true values. For this purpose, a part of the values is deleted completely at random with a probability of π. We chose $\pi = 0.05, 0.10, 0.15, 0.20, 0.25$. The tuning parameters λ and ω when using `wNNSelect` are chosen by cross-validation. The value of k in the KNN imputation is also determined by cross-validation. For the random forest method, we use $m_{try} \in \{\sqrt{p}, p/3\}$ and $n_{tree} = 100$. All the imputation results represent the average over five independent runs for all three data sets for each missing probability.

Effect of Missing Data

Figure 4.4 shows the violin plots of MSIEs for each imputation method and different probabilities of missing values.

The average MSIE results for all three gene expression data sets are shown in Table 4.1, with the smallest MSIEs in each setting given in boldface.

It is seen that `wNNSelect` yields the best results. The mean and zero imputation methods show the highest MSIEs, with zero imputation performing slightly better because the data were standardized to zero mean and unit variance. Mean imputation performs worst as it ignores the correlation between genes. The random forest method is a strong competitor, but nevertheless, the nearest neighbor method with selected distances shows the smallest MSIEs for all values of π in all three data sets. The random forest method is a strong competitor even though the nearest neighbor method with selected distances still shows the smallest MSIEs for all values of π in all three data sets.

For most methods, the performance does not depend strongly on the probability of missing values π (at least for the values of π considered here). In particular, `BIO` and `VIM` are rather stable over different values of π. The strongest dependence is seen for `Random Forests` and `wNNSelect`. For these, an increase in π also means an increase in MSIEs.

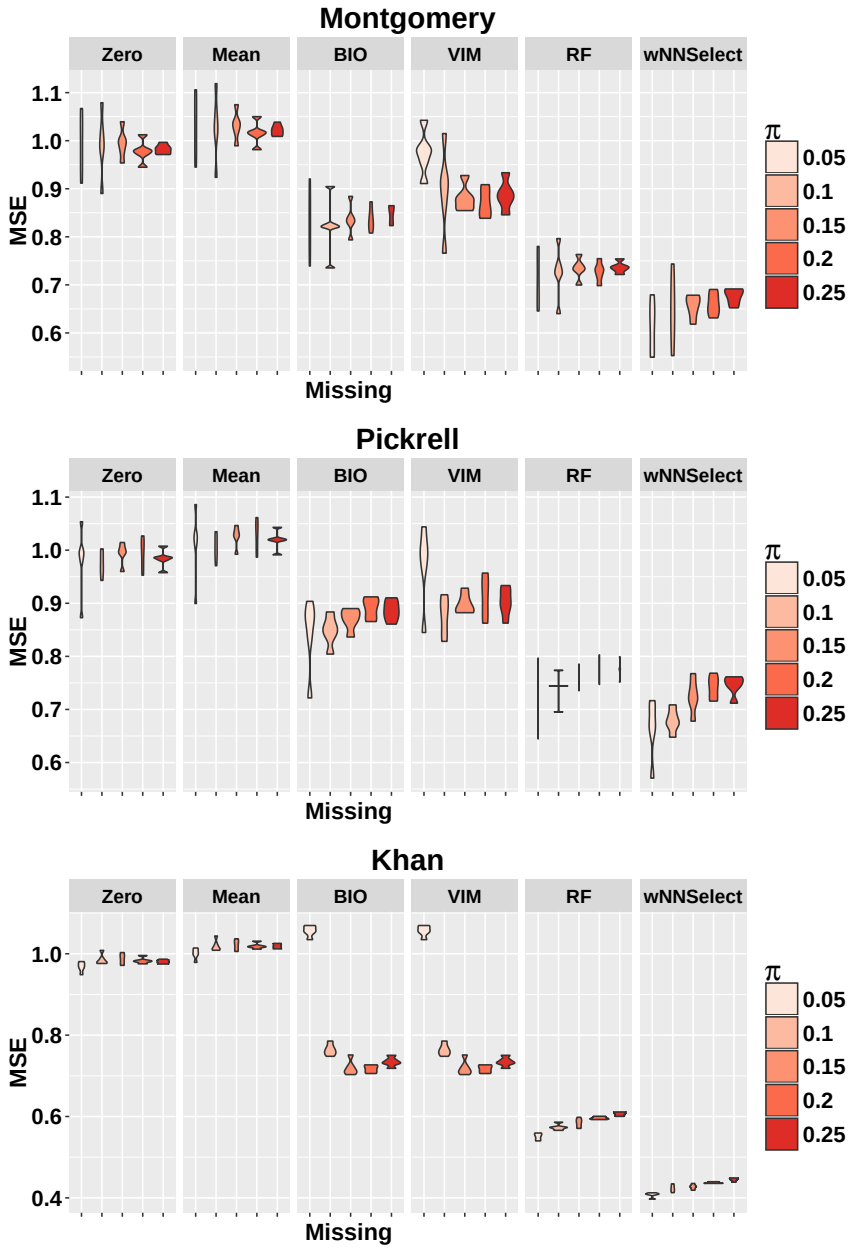


Figure 4.4.: Effect of missing probability: Comparison of MSIEs with varying probability of missing values (π), for the real data sets using different imputation methods.

4.4. Concluding Remarks

Several imputation methods that are available for completing gene expression data matrices are compared in terms of the accuracy of the imputed values. We investigated the performance of the weighted nearest neighbor imputation method for different numbers of missing data and small sample sizes. For this purpose three different real data sets were used. The results show that for gene expression and RNA-seq data, the weighted nearest neighbors

Table 4.1.: Average MSIEs using real data sets. The results shown are average of five independent runs for each probability of missing data (π) = 0.05, 0.10, 0.15, 0.20, 0.25.

					RF (m_{try} =)		
π	ZERO	MEAN	BIO	VIM	$\sqrt{p}$	$p/3$	`wNNSelect`
	Montgomery data						
0.05	0.99	1.03	0.83	0.97	0.73	0.71	**0.61**
0.10	0.99	1.03	0.82	0.90	0.75	0.73	**0.65**
0.15	0.99	1.03	0.84	0.88	0.76	0.73	**0.66**
0.20	0.98	1.02	0.84	0.87	0.75	0.73	**0.66**
0.25	0.98	1.02	0.85	0.89	0.76	0.74	**0.68**
	Pickrell data						
0.05	0.98	1.01	0.84	0.97	0.77	0.74	**0.66**
0.10	0.97	1.01	0.85	0.88	0.77	0.74	**0.68**
0.15	0.99	1.03	0.87	0.90	0.79	0.77	**0.72**
0.20	0.99	1.03	0.89	0.91	0.81	0.78	**0.74**
0.25	0.98	1.02	0.89	0.90	0.81	0.78	**0.74**
	Khan data						
0.05	0.97	1.00	1.06	1.06	0.59	0.55	**0.41**
0.10	0.99	1.02	0.76	0.76	0.62	0.57	**0.42**
0.15	0.99	1.02	0.72	0.72	0.63	0.58	**0.43**
0.20	0.98	1.02	0.72	0.72	0.64	0.60	**0.44**
0.25	0.98	1.02	0.73	0.73	0.64	0.61	**0.45**

method with selected distances can handle missing values effectively. In the examples considered, it outperforms its competitors consistently even when the sample size is very low, and shows even better performance than Random Forests, which are quite effective tools to impute missing values.

5. Nearest Neighbor Imputation for Categorical Data by Weighting of Attributes

Abstract

Missing values are a common phenomenon in all areas of applied research. While various imputation methods are available for metrically scaled variables, methods for categorical data are scarce. An imputation method that has been shown to work well for high-dimensional metrically scaled variables is the imputation by nearest neighbor methods. In this paper, we extend the weighted nearest neighbors approach to impute missing values in categorical variables. The proposed method, called $\texttt{wNNSel}_{\texttt{cat}}$, explicitly uses the information on association among attributes. The performance of different imputation methods is compared in terms of the proportion of falsely imputed values. Simulation results show that the weighting of attributes yields smaller imputation errors than existing approaches. A variety of real data sets is used to support the results obtained by simulations.

5.1. Introduction

Categorical data are important in many fields of research, examples are surveys with multiple-choice questions in the social sciences (Chen and Shao, 2000), single nucleotide polymorphisms (SNPs) in genetic association studies (Schwender, 2012) and tumor or cancer studies (Eisemann et al., 2011). It is most likely that some respondents/patients do not provide the complete information on the queries, which is the most common reason for missing values. Sometimes, also the information may not be recorded or included into the database. Whatever the reason, missing data occur in all areas of applied research. Since for many statistical analyses a complete data set is required, the imputation of missing values is a useful tool. For categorical data, although prone to contain missing values, imputation tools are scarce.

It is well known that using the information from complete cases or available cases may lead to invalid statistical inference (Little and Rubin, 2014). A common approach is to use an appropriate imputation model, which accounts for the scale level of the measurements. When the data are categorical the log-linear model is an appropriate choice (Schafer, 1997). The

simulation studies of Ezzati-Rice et al. (1995) and Schafer (1997) showed that it provides an attractive solution for missing categorical data problems. However, its use is restricted to cases with a small number of attributes (Erosheva et al., 2002) since model selection and fitting becomes very challenging for larger dimensions.

A non-parametric method called hot-deck imputation has been proposed as an alternative (Rubin, 1987). This technique searches for the complete cases having the same values on the observed variables as the case with missing values. The imputed values are drawn from the empirical distribution defined by the former. The method is well suited even for data sets with a large number of attributes (Cranmer and Gill, 2013). A variant, called approximate Baysian bootstrap, works well in situations where the standard hot-deck fails to provide proper imputation Rubin and Schenker (1986). But the hot-deck imputation may yield biased results irrespective of the missing data mechanism (Schafer and Graham, 2002), and it may become less likely to find matches if the number of variables is large (Andridge and Little, 2010).

Another popular non-parametric approach to impute missing values is the nearest neighbors method (Troyanskaya et al., 2001). The relationship among attributes is taken into account when computing the degree of nearness or distance. The method may easily be implemented for high-dimensional data. However, the k-nearest neighbors (kNN) method, originally developed for continuous data, cannot be employed without modifications to non-metric data such as nominal or ordinal categorical data (Schwender, 2012). As the accuracy of the kNN method is mainly determined by the distance measure used to calculate the degree of nearness of the observations, one needs different distance formula when data are categorical. Some existing methods to impute attributes are based on the mode or weighted mode of nearest neighbors (Liao et al., 2014).

Schwender (2012) suggested a weighted kNN method to impute *categorical* variables only, that uses the Cohen or Manhattan distance for finding the nearest neighbors. The imputed value is calculated by using weights that correspond to the inverse of the distance. One limitation of this approach is that it can handle only variables that have the same number of categories. Also the value of k, which strongly affects the imputation estimates, is needed. There are some methods for imputing mixed data that can also be used for categorical data, for example, see Liao et al. (2014), Stekhoven and Bühlmann (2012). The latter transform the categorical data to dichotomous data and use the classical k-nearest neighbors method to the standardized data with mean 0 and variance 1. The imputed data are re-transformed to obtain the estimates. However, it has been confirmed by several studies that rounding may lead to serious bias, particularly in regression analysis (Allison (2005), Horton et al. (2003)).

For categorical data one has to use specific distances or similarity measures, which are typically based on contingency tables. Commonly used distance measures include the simple matching coefficient, Cohen's kappa κ_c (Cohen, 1960)), and the Manhattan or L_1 distance. The Euclidean or variants of the Minkowski distance give an equal importance to all the variables in the data matrix when computing the distance. But for a larger number of variables, the equal weighting ignores the complex structure of correlation/association among

these variables. As will be demonstrated, better distance measures are obtained by utilizing the association between variables. More specific, we propose a weighted distance that explicitly takes the association among covariates into account. Strongly associated covariates are given higher weights forcing them to contribute more strongly to the computation of the distances than weakly associated covariates.

The paper is organized as follows: Section 5.2 reviews some available measures to calculate the distance for nominal data. The improved measure, which uses information on association among attributes, is introduced. A weighted nearest neighbors procedure is described in Section 5.3. In Section 5.4, the existing methods to impute missing categorical data, and the measure of performance used for comparison are described. In section 5.5 the performance of different imputation methods is compared in simulation studies. Section 5.6 gives applications of the proposed method to some real data sets.

5.2. Methods

At the core of nearest neighbor methods is the definition of the distance. In contrast to continuous data, computation of distance or similarity between categorical data is not straightforward since categorical values have no explicit notion of ordering. We first consider distances for categorical variables that can be used to impute missing values.

5.2.1. Distances for Categorical Variables

Let data be collected in a $(n \times p)$-matrix $\mathbf{Z} = (Z_{is})$, where Z_{is} is the i^{th} observation on the s^{th} attribute. Let $\mathbf{z}_i^T = (Z_{i1}, \cdots, Z_{ip})$ denote the ith row or observation vector in the data matrix $\mathbf{Z}$. The categorical observations Z_{is} in the data matrix $\mathbf{Z}$ can take values from $\{1, \ldots, k_s\}, s = 1, \ldots, p$, where k_s is the number of categories of the s^{th} attribute. Distances can use the original variables $Z_{is} \in 1, \ldots, k_s$ or the vector representation.

Simple Matching of Coefficients (SMC)

The distance uses the original values Z_{is}. This method simply considers the matching of the values of the variables, that is, whether they are the same or not Sokal and Michener (1958). The SMC distance between two observation vectors $\mathbf{z}_i, \mathbf{z}_j$ is defined by

$$d_{SMC}(\mathbf{z}_i, \mathbf{z}_j) = \sum_{s=1}^{p} I(Z_{is} \neq Z_{js})$$

where $I(.)$ is an indicator function defined by

$$I(.) = \begin{cases} 1 & \text{if } Z_{is} \neq Z_{js} \\ 0 & \text{otherwise.} \end{cases}$$

Minkowski's Distance

The Minkowski's distance can be used for re-coded categorical variables.

For the computation the categorical variable Z_{is} is transformed into binary variables. Let $\mathbf{z_{is}}^T = (z_{is1}, \ldots, z_{isk_s})$ be the dummy vector built from Z_{is} with components being defined by

$$z_{isc} = \begin{cases} 1 & \text{if } Z_{is} = c, \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mathbf{Z}^D$ denote the matrix of dummies which is obtained from the original data matrix. Thus, the ith row of the matrix $\mathbf{Z^D}$ has the form $(\mathbf{z}_{i1}^T, \cdots, \mathbf{z}_{ip}^T)^T$ with dummy vectors $\mathbf{z}_{is}$, $s = 1, \cdots, p$.

The dummy vectors $\mathbf{z_{is}}^T$ for a nominal variable with four categories can be written as

category	z_{is1}	z_{is2}	z_{is3}	z_{is4}
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

By using the dummy vectors from $\mathbf{Z^D}$ the Minkowski's distance between two rows $\mathbf{z}_i, \mathbf{z}_j$ of $\mathbf{Z}$ is given by

$$d_{Cat}(\mathbf{z}_i, \mathbf{z}_j) = \left(\sum_{l=1}^{p} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|^q \right)^{1/q}, \tag{5.1}$$

By choosing $q = 2$, one obtains the Euclidean distance

$$d_{Cat}(\mathbf{z}_i, \mathbf{z}_j) = \left(\sum_{l=1}^{p} \sum_{c=1}^{k_l} (z_{ilc} - z_{jlc})^2 \right)^{1/2},$$

and for $q = 1$ one obtains the Manhattan or L_1 distance

$$d_{Cat}(\mathbf{z}_i, \mathbf{z}_j) = \sum_{l=1}^{p} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|.$$

Thus the Euclidean distance and Manhattan distance are two special forms of the Minkowski's distance. It should be noted that it has a strong connection to the matching coefficient distance. When $q = 1$, the Minkowski's distance uses the number of matches between the two covariates, ant the distance is equal to the simple matching coefficient (SMC) distance.

5.2.2. Selection of Attributes by Weighted Distances

The Euclidean or variants of Minkowski distance give an equal importance to all the variables in the data matrix. When the number of variables is large and they are correlated/associated, it is useful to give unequal weights to covariates when calculating the distance. We present a weighted distance which explicitly takes the association among covariates into account. More specifically, highly associated covariates will contribute more to the computation of the distance than less associated covariates.

For a concise definition we distinguish between cases that were observed in the corresponding component and missing values, only the former contribute to the computation of the distance. Let us again consider the data matrix $\mathbf{Z}$ with dimension $n \times p$. Let $\mathbf{O} = (o_{is})$ denote the $n \times p$ matrix of dummies, with $o_{is} = 0$ if the value is missing, and $o_{is} = 1$ if the the value is available in the data matrix $\mathbf{Z}$. Let now Z_{is} be a specific missing entry in the data matrix $\mathbf{Z}$, that is, $o_{is} = 0$. For the computation of distances we use the corresponding matrix of dummy variables $\mathbf{Z}^D$.

We propose to use as distance between the i-th and the j-th observation

$$d_{CatSel}(\mathbf{z}_i, \mathbf{z}_j) = \left(\frac{1}{a_{ij}} \sum_{l=1}^{p} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|^q I(o_{il} = 1) I(o_{jl} = 1) C(\delta_{sl}) \right)^{1/q}, \tag{5.2}$$

where $I(.)$ denotes the indicator function and $a_{ij} = \sum_{l=1}^{p} I(o_{il} = 1) I(o_{jl} = 1)$ is the number of valid components in the computation of distances. The crucial part in the definition of the distance is the weight $C(\delta_{sl})$. $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the measure of association between attributes s and l, denoted by δ_{sl}, into weights.

It is worth noting that the distance is now specific to the s^{th} attribute, which is to be imputed. For $C(.)$ we use the power function $C(\delta_{sl}) = |\delta_{sl}|^{\omega}$. So the attributes that have a higher association with the s^{th} attribute are contributing more to the distance and vice versa. The higher the value of association, the more it contributes to the computation of the distance. Note also that only the *available* pairs with $I(o_{il} = 1) I(o_{jl} = 1)$ are used for the computation of the distance. In the following we describe some of association among variables that account for the number of categories each variable can take.

5.2.3. Measuring Association Among Attributes

One important issue in the computation of distances in equation (5.2) is how to compute the association among categorical variables as the usual Pearson coefficient of correlation is not suitable for categorical covariates. In this section, we briefly describe some measures that are used to calculate the association between categorical measurements.

The measures of association for two nominal or categorical variables are typically based on the χ^2-statistic which tests the independence of variables in contingency tables.

Consider the association among attribute s and l where attribute s has $i = 1, ..., k_s$ categories and attribute l has $j = 1, ..., k_l$ categories. The two attributes s and l can be presented in the form of an $k_s \times k_l$ contingency table (Figure 5.1).

		Attribute l						
		1	2	$\cdots$	j	$\cdots$	k_l	Total
Attribute s	1	n_{11}	n_{12}					$n_{1.}$
	2	n_{21}	n_{22}					$n_{2.}$
	$\cdots$			$\ddots$				
	i				n_{ij}			$n_{i.}$
	$\cdots$					$\ddots$		
	k_s						$n_{k_s k_l}$	
	Total	$n_{.1}$	$n_{.2}$		$n_{.j}$			n

Figure 5.1.: Contingency table with $k_s \times k_l$ cells

In the contingency table, n_{ij} is the number of observations ($Z_s = i$, $Z_l = j$) and $n = n_{..}$ is the total number of values. The χ^2-statistic between attributes s and l is defined as

$$\chi^2_{sl} = \sum_{i,j} \frac{(n_{ij} - \frac{n_{i.}n_{.j}}{n})^2}{\frac{n_{i.}n_{.j}}{n}},$$

where $n_{i.}$ and $n_{.j}$ are the row and columns totals respectively and n is the total number of observations in the contingency table.

Association measures based on the χ^2-statistic are, in particular, the ϕ-coefficient, Pearson's contingency Coefficient and Cramer's V.

Phi coefficient (ϕ)

For nominal variables with only two categories, i.e. $k_s = k_l = 2$, a simple measure of association is the ϕ-coefficient

$$\phi_{sl} = \sqrt{\frac{\chi^2_{sl}}{n}}.$$

Pearson's coefficient of contingency ($\mathcal{PCC}$)

For $k_s = k_l = k$, that is, the variables have the same number of categories, Pearson's coefficient of contingency is computed as

$$C_{sl} = \sqrt{\frac{\chi^2_{sl}}{\chi^2_{sl} + n}}.$$

It can be corrected to reach a maximum value of 1 by dividing by the factor $\sqrt{(k-1)/k}$, where k is the number of rows($r = k_s$) or columns ($c = k_l$) as both are equal. Pearson's corrected coefficient (PCC) of contingency is given by

$$\text{PCC}_{sl} = \frac{C_{sl}}{\sqrt{(k-1)/k}}.$$

It is suitable only when the number of categories of both covariates are the same.

Cohen's kappa (κ_c)

For $k_s = k_l = k$, another useful measure of association was given by Cohen (1960),

$$\kappa_{sl} = \frac{p_0 - p_e}{1 - p_e},$$

where $p_0 = n_{ii}/n = n_{jj}/n$ are the proportions of units with perfect agreement, which are the diagonal elements in the contingency table and $p_e = \sum_{i=j}^{k} \frac{n_{i.}n_{.j}}{n}$ is the expected proportion of units under independence.

Cramer's V

If the covariates have different number of categories ($k_s \neq k_l$), Cramer's V is an attractive option Cramér (1946). It is defined by

$$\text{Cramer's V} = \sqrt{\frac{\chi^2_{sl}/n}{min(k_s - 1, k_l - 1)}},$$

where $n = k_s \times k_l$ is the total number of cells in the contingency table.

In this paper, we choose Cramer's V as the measure of association (δ_{sl}) to be used in (5.2) because its value lies between 0 and 1; and it can be used for unequal number of categories of the attributes as well. The corresponding method is denoted by $\texttt{wNNSel}_{\texttt{cat}}$.

5.3. Using Nearest Neighbors to Impute Missing Values

Classical nearest neighbor approaches fix the number of neighbors that are used. We prefer to use weighted nearest neighbors by using weights that are defined by kernel functions. Uniform kernels yield the classical approach, however, smooth kernels typically provide better results.

Let Z_{is} be a missing value in the $n \times p$ matrix of observations. The k nearest neighbor observation vectors are defined by

$$\mathbf{z}^D_{(1)}, \ldots, \mathbf{z}^D_{(k)} \quad \text{with} \quad d(\mathbf{z}_i, \mathbf{z}_{(1)}) \leq \cdots \leq d(\mathbf{z}_i, \mathbf{z}_{(k)})$$

where $\mathbf{z}^D_{(i)}$ are rows from the matrix $\mathbf{Z}^D$, and $d(\mathbf{z}_i, \mathbf{z}_{(k)})$ is the computed distance using equation (5.2). It is important to mention that the row $\mathbf{z}^D_{(i)}$ is composed of values of dummy variables of the form $(\mathbf{z}^T_{(i)1}, \cdots, \mathbf{z}^T_{(i)p})^T$, where $\mathbf{z}^T_{(i)s} = (z_{is1}, \cdots, z_{isk_s})$ are the dummy values.

For the imputation of the value Z_{is} we use the weighted estimator

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\mathbf{z}_i, \mathbf{z}_{(j)}) \mathbf{z}_{(j)sc}, \tag{5.3}$$

with weights given by

$$w(\mathbf{z}_i, \mathbf{z}_{(j)}) = \frac{k(\frac{d(\mathbf{z}_i, \mathbf{z}_{(j)})}{\lambda})}{\sum_{h=1}^{k} k(\frac{d(\mathbf{z}_i, \mathbf{z}_{(h)})}{\lambda})}, \tag{5.4}$$

where $k(.)$ is a kernel function (triangular, Gaussian etc.) and λ is a tuning parameter. Note that $\hat{\pi}^T_{is} = (\hat{\pi}_{is1}, \cdots, \hat{\pi}_{isk_s})$ is a vector of estimated probabilities.

If one uses all the available neighbors that is $k = n$, then λ is the only and crucial tuning parameter. The imputed estimate of Z_{is} is the value of $c \in \{1, \ldots, k_s\}$ that has the largest value. In other words, the weighted imputation estimate of a categorical missing value Z_{is} is

$$\hat{Z}_{is} = \arg\max_{c=1}^{k_s} \hat{\pi}_{isc}, \tag{5.5}$$

If the maximum is not unique one value is selected at random. The proposed `wNNSel`$_{\texttt{cat}}$ procedure can be described as follows:

1. Locate a missing value Z_{is} in sample i and attribute s in the data matrix.
2. Compute d_{CatSel} between observation vectors $\mathbf{z}_i, \mathbf{z}_j$ using equation (5.2).

3. Rank samples (rows) of the matrix $\mathbf{Z}^D$ based on d_{CatSel}.
4. Compute the corresponding weights based on kernel function $w(\mathbf{z}_i, \mathbf{z}_j)$ using equation (5.4).
5. Compute the s probabilities by using equation (5.3)
6. Compute the missing value $\hat{z}_{is}$ by using equation (5.5).
7. Seek the next missing value in $\mathbf{Z}$ and repeat steps (2-6) until all missing values in $\boldsymbol{Z}$ have been imputed imputed.

In this weighted nearest neighbor ($\texttt{wNNSel}_{\texttt{cat}}$) method, one has to deal two tuning parameters, λ in (5.4) and ω to be used in distance equation (5.2). These tuning parameters have to be specified

5.3.1. A Pearson Correlation Strategy

As an alternative we consider a strategy that uses the dummy variables directly. Starting from the matrix of dummies $\mathbf{Z}^D$ we use the Pearson correlation coefficient between dummy variables as association measure. The weighting scheme remains the same. The imputation is again determined by

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\mathbf{z}_i, \mathbf{z}_{(j)}) \mathbf{z}_{(j)sc}.$$

Although $\hat{\pi}_{isc}^T = (\hat{\pi}_{is1}, \ldots, \hat{\pi}_{isk_s})$ might not be a vector of probabilities, simple standardization by setting

$$\tilde{\pi}_{isc} = \hat{\pi}_{isc} / \sum_{r=1}^{k_s} \hat{\pi}_{isr}$$

yields a vector of probabilities that can be used to determine the mode. The method can be seen as an adaptation of the weighting method proposed by Tutz and Ramzan (2015). Only small modification are needed to apply the method to the dummy variables. One is that the missing of an observation refers to a set of variables, namely all the dummies that are linked to a missing value. For this method the Gaussian kernel and the Euclidean distance are used throughout. The method is denoted by $\texttt{wNNSel}_{\texttt{dum}}$.

5.3.2. Cross Validation

The imputation procedure $\texttt{wNNSel}_{\texttt{cat}}$ requires pre-specified values of the tuning parameters λ and ω. In this section we present a cross validation algorithm that automatically choses those values for which the imputation error is minimum.

To estimate the tuning parameters ω and λ, we set some available values $(o_{is}) = 1$ in the data matrix as missing $(o_{is}) = 0$. The advantage of this step is that these values are used to

estimate the tuning parameters. The procedure of cross validation to find optimal tuning parameters is given as algorithm 1.

Algorithm 5.1 Cross-validation for `wNNSelcat`

Require: $\mathbf{Z}$ an $n \times p$ matrix, number of validation sets t, range of suitable values for tuning parameters $\mathcal{L}$ and $\mathcal{W}$

$\mathbf{Z}^{\text{cv}} \leftarrow$ initial imputation using unweighted 5-nearest neighbors
for $t = 1, \ldots, \mathcal{T}$ **do**
  $\mathbf{Z}^{\text{cv}}_{\text{miss,t}} \leftarrow$ artificially introduce missing values to $\mathbf{Z}^{cv}$
  for $\omega \in \mathcal{W}$ **do**
    for $\lambda \in \mathcal{L}$ **do**
      $\mathbf{Z}^{\text{cv}}_{wNNSel_{cat},t} \leftarrow$ imputation of $\mathbf{Z}^{\text{cv}}_{\text{miss,t}}$ using $\texttt{wNNSel}_{\texttt{cat}}$ procedure
      $\psi_{(\lambda,\omega),t} \leftarrow$ imputation error (PFC) of $\texttt{wNNSel}_{\texttt{cat}}$ procedure for λ & ω
    end for
  end for
  Determine $(\lambda, \omega)_{\text{best}} \leftarrow \text{argmin } \frac{1}{T} \sum_{t=1}^{T} \psi_{(\lambda,\omega),t}$
end for
$\mathbf{Z}^{\text{imp}} \leftarrow \texttt{wNNSel}_{\texttt{cat}}$ imputation of $\mathbf{Z}$ using $(\lambda, \omega)_{\text{best}}$

5.4. Evaluation of Performance

This section briefly describes some available methods to impute missing categorical data. Then the performance of imputation methods is compared by using the mean proportion of falsely imputed categories (PFC) given by

$$\text{PFC} = \frac{1}{m} \sum_{Z_{is}:o_{is}=0} I(Z_{is} \neq \hat{Z}_{is}),$$

where $I(.)$ is an indicator function which takes the value 1 if $Z_{is} \neq \hat{Z}_{is}$ and 0 otherwise, ω is the number of missing values in the data matrix, Z_{is} is the true value and $\hat{Z}_{is}$ is its imputed value.

5.4.1. Existing Methods

In this section, we briefly review some existing procedures for the imputation of categorical missing data.

Mode Imputation

This is perhaps the simplest and fastest method to fill the incomplete categorical data. The missing values of an attribute are replaced by the attribute with maximal occurrence

in that variable, that is, the mode. The approach is very simple and totally ignores the association or correlation among the attributes.

k-Nearest Neighbors Imputation

In this method, k neighbors are chosen based on some distance measure and their average is used as an imputation estimate. The method requires the selection of a suitable value of k, the number of nearest neighbours, and a distance metric. The function `kNN` in the R package `VIM` (Templ et al., 2016) can impute categorical and mixed type of variables.

An adaptation of the k nearest neighbors algorithm proposed by Schwender (2012) can impute missing genotype or categorical data. The procedure selects k nearest neighbors based on distance measures (Cohen, Pearson, or SMC). The weighted average of the k nearest neighbors is used to estimate the missing value, where the weights are defined by the inverse of the distances. The limitation of this method is that it offers imputation only for variables having an equal number of categories. We use the function `knncatimpute` from R package `scrime` Schwender and Fritsch (2013) to apply this method.

Random Forests

In recent years, random forest Breiman (2001) have been used in various areas including imputation of missing values. The imputation of missing categorical data by random forests is based on an iterative procedure that uses initial imputations using mode imputation and then improves the imputed data matrix on successive iterations. A random forest model is developed for each predictor with a missing value by using the rest of the predictors and the model to estimate the missing value of that predictor. The imputed data matrix is updated and the difference between previous and new imputation is assessed at the end of each iteration. The whole process is repeated until a specific criterion is met (Stekhoven and Bühlmann (2012); Rieger et al. (2010); Segal (2004); Pantanowitz and Marwala (2009)).

The main advantages of random forests include the ability to handle high-dimensional mixed-type data with non-linear and complex relationships, and robustness to outliers and noise (Hill (2012), Rieger et al. (2010)). The `missForest` package in the statistical programming language `R` offers this approach (Stekhoven, 2013).

5.5. Simulation Studies

This section includes preliminary simulations to check if the suggested distance measure contributes to better imputation or not. Using simulated data we compare our method with three existing methods in the situations with binary or multi-categorical variables only, and mixed (binary and multi-categorical) variables.

5.5.1. Binary Variables

In our fist simulation study we investigate the performance of imputation methods using simulated binary data. We generate $S = 200$ samples of size $n = 100$ for $p = 30$ predictors drawn from a multivariate normal distribution with $N(\mathbf{0}, \mathbf{\Sigma})$. The correlation matrix $\mathbf{\Sigma}$ has an autoregressive type of order 1 with $\rho = 0.8$. The data are converted to binary variables by defining positive values as the first category and negative as the second. In each sample randomly selected values are declared as missing with proportion of 10%, 20% and 30%. The missing values are imputed using mode imputation, random forests (`RF`) and the proposed weighted nearest imputation methods. In the weighted nearest imputation methods ($\texttt{wNNSel}_{\texttt{cat}}$), the distance (5.2) is computed for $q = 1, 2$. We use the Gaussian and the triangular kernel functions for each value of q (shown as `Gauss.q1`, `Gauss.q2`, `Tri.q1`, and `Tri.q2` in Figure 5.2).

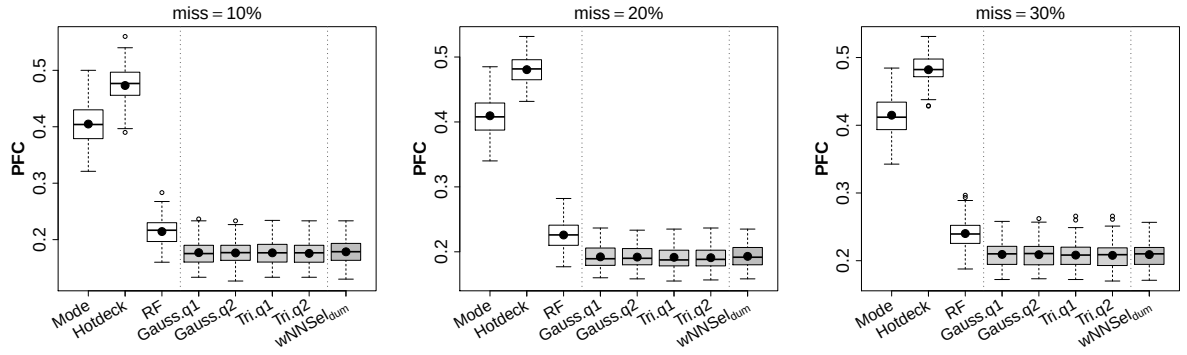


Figure 5.2.: Simulation study for binary data: boxplots of PFCs for MCAR missing data with $n = 100$, $p = 30$. Solid circles within boxes show mean values.

The resulting PFCs are shown in Figure 5.2. It is clear from the figure that the weighted imputation methods ($\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$) yield smaller error than mode, hot deck and random forest imputation methods. The highest errors are obtained by hot deck and mode imputation. It seems that the selection of the kernel function and the value of q do not affect the results. To have a closer look into the results, the average values are given table 5.1. The smallest value in each row is boldfaced. The average values of the PFCs are nearly equal for $q = 1, 2$ when using the triangular kernel. For the Gaussian kernel, $q = 2$ produces slightly smaller imputation errors. Overall the ($\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$) procedures give similar results. We skip the Hotdeck method in our further simulations as it produced poor imputations.

5.5.2. Multi-Categorical Variables

In this section, we investigate the performance of imputation methods using multi-categorical data. We generate $S = 200$ samples of size $n = 100$ for $p = 10, 50$ predictors drawn from a multivariate normal distribution with $N(\mathbf{0}, \mathbf{\Sigma})$. The correlation matrix $\mathbf{\Sigma}$

Table 5.1.: Comparison of imputation methods using Binary data

miss	MODE	Hotdeck	RF	wNNSel$_{cat}$				wNNSel$_{dum}$
				Gauss.q1	Gauss.q2	Tri.q1	Tri.q2	
10%	0.4049	0.4731	0.2143	0.1769	0.1764	0.1766	**0.1756**	0.1784
20%	0.4095	0.4806	0.2259	0.1923	0.1919	0.1916	**0.1909**	0.1930
30%	0.4149	0.4820	0.2401	0.2093	0.2089	0.2083	**0.2079**	0.2090

has an autoregressive type of order 1 with $\rho = 0.9$. These values are then converted to the desired number of categories. In each sample, $miss = 10\%, 20\%, 30\%$ of the total values were replaced by missing values completely at random (MCAR).

In the case where all attributes have an equal number of categories ($k_s = k$), another benchmark proposed by Schwender (2012), is also considered. To compare the performance, the proportion of falsely imputed categories (PFC) is computed for each imputation method. We distinguish the cases when the probabilities are the same for all categories and the case when they are not equal.

Effect of the Number of categories

$k_s = k$ (the number of categories is the same for all the attributes)

We construct categories from the continuous data by setting cut points. In the first simulation setting, we assume that all the categories within each attribute have the same probability. For example, for an attribute having four categories $k_s = 4$, the quartiles Q_1, Q_2, Q_3 are used as cut points, where Q_1, Q_2, Q_3 are the usual lower quartile, median and upper quartile respectively, which divide the data into an equal four parts. So in this case, $\pi_1 = \pi_2 = \pi_3 = \pi_4 = 0.25$. In general, to create c categories of a variable one needs $c - 1$ cut points.

In our second simulation setting, the number of categories (k_s) of all the attributes is the same but the categories within each attribute may have the unequal probabilities ($\pi_c \neq 1/k$). The purpose is to investigate whether π_c do have any effect on the imputation results.

We use $q = 1, 2$ in the distance calculation of wNNSel$_{cat}$ method to get L_1 and L_2 metrics (shown as wNNSel$_{cat}$q$_1$ and wNNSel$_{cat}$q$_2$ in Figure 5.3). The tuning parameters are chosen by cross validation and these optimal values, λ_{opt} and m_{opt}, are used to estimate the final imputed values. Using dummy variable method, the missing values are imputed and shown as wNNSel$_{dum}$ in Figure 5.3.

It is seen from Figure 5.3 (upper panel), that for $\pi_c = 1/k$, wNNSel$_{dum}$ method yields the smallest imputation errors followed by wNNSel$_{cat}$. For the wNNSel$_{cat}$ method, both values of q produce similar results. The method by Schwender (2012) is also used as benchmark

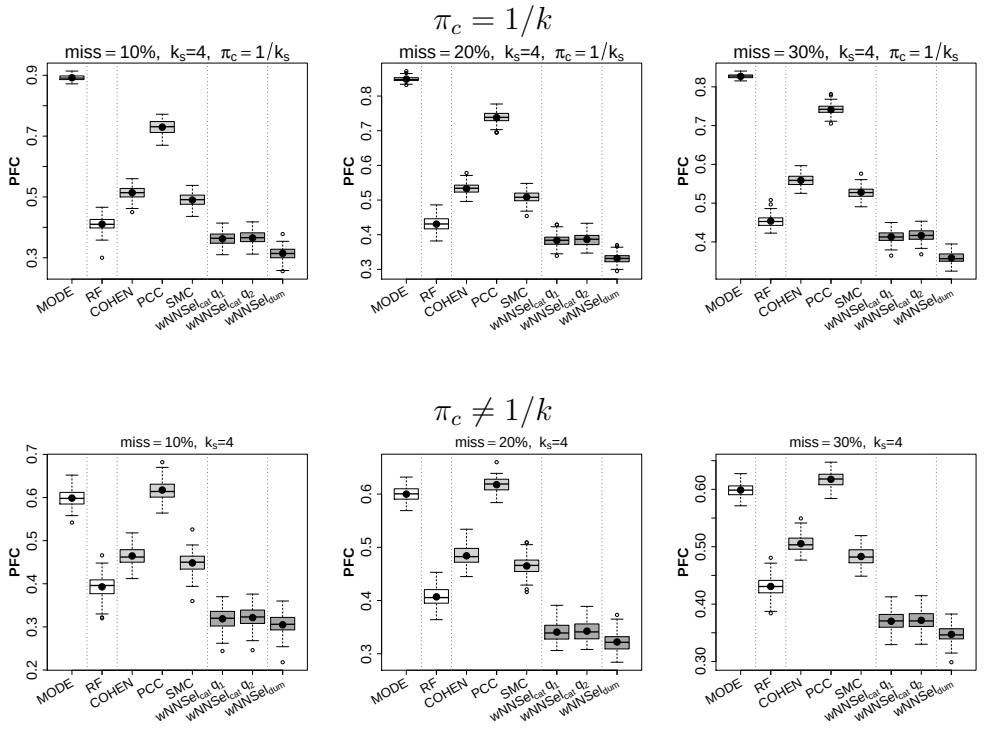


Figure 5.3.: Simulation study for multi-categorical data (the number of categories is the same for all the attributes): Boxplots of PFCs for MCAR missing pattern with $k_s = 4$, $S = 200$ samples of size $n = 100, p = 50$ were drawn from a multivariate normal distribution using autoregressive correlation structures to form the categories. Solid circles within boxes show mean values. Upper row shows when the probability of occurrence of each category is the same and lower row for probability of occurrence of each category is not same.

as all the attribute have an equal number of categories. We used Cohen, PCC and SMC distances to compute the nearest neighbors for this method (light-gray boxes in Figure 5.3). The PCC distance gives higher imputation errors than the Cohen and SMC distances which yield almost similar results. The same findings can be seen for $\pi_c \neq 1/k$, in the lower panel of Figure 5.3. Overall, the replacement of missing values by the mode yields the highest errors followed by KNN and random forests, while weighted nearest neighbors imputation with weighting as proposed here provides the smallest errors.

$k_s \neq k$ (the number of categories is different for the attributes)

In this simulation setting, we explore whether the $\texttt{wNNSel}_{\texttt{cat}}$ method works in the situation when the attributes have an unequal number of categories ($k_s \neq k$). Following the same process as in the previous subsections, we use $k_s = \{3, 4\}$ and $k_s = \{3, 4, 5\}$ for the predictors in this case. Furthermore, the probability of occurrence of each category (π_c) is not same i.e., $\pi_c \neq 1/k_s$. We set $n = 100$, $p = 50$ and $miss = 10\%, 20\%, 30\%$ values are deleted at random. The rest of the procedure of imputing missing values is the same.

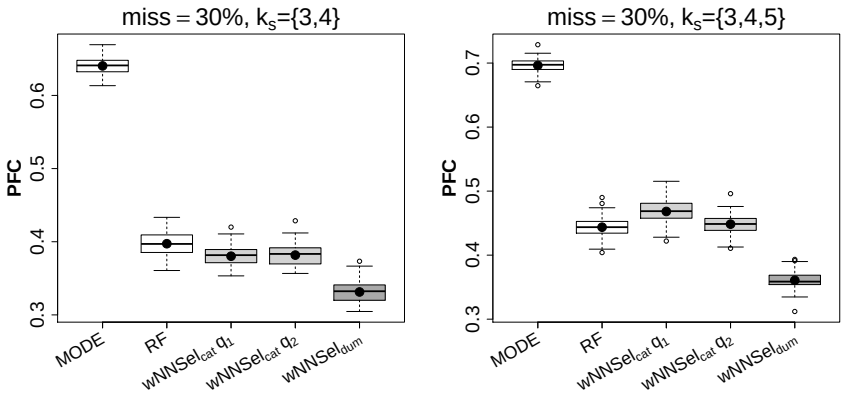


Figure 5.4.: Simulation study for multi-categorical data (the number of categories is different for the attributes): Boxplots of PFCs for MCAR missing pattern with $k_s \neq k$, $S = 200$ samples were drawn from multivariate normal distribution using autoregressive correlation structures to form the categories. Solid circles within boxes show mean values.

The resulting PFCs for $miss = 30\%$ only are shown in Figure 5.4. The left panel shows the results for $k_s = \{3, 4\}$ and the right panel for $k_s = \{3, 4, 5\}$. It is to be noted that the KNN method of Schwender (2012) is not applicable in these settings. Clearly, the mode imputation shows the highest errors followed by random forests. It is interesting that the random forest method perform pretty well and yields similar results as the $\texttt{wNNSel}_{\texttt{cat}}$ method. Here again, the smallest errors are obtained by the $\texttt{wNNSel}_{\texttt{dum}}$ method in both settings considered. The detailed results for other the settings are shown in Table 5.2.

Table 5.2.: Comparison of imputation methods using multi-categorical simulated data

	miss	MODE	RF	$\texttt{wNNSel}_{\texttt{cat}}$		$\texttt{wNNSel}_{\texttt{dum}}$
				$q = 1$	$q = 2$	
$k_s = \{3, 4\}$	10%	0.6377	0.3578	0.3290	0.3291	**0.3198**
	20%	0.6385	0.3767	0.3524	0.3533	**0.3087**
	30 %	0.6404	0.3973	0.3803	0.3817	**0.3314**
$k_s = \{3, 4, 5\}$	10%	0.6989	0.4066	0.4227	0.3969	**0.3198**
	20%	0.6974	0.4212	0.4399	0.4205	**0.3392**
	30 %	0.6963	0.4437	0.4682	0.4484	**0.3607**

5.5.3. Mixed (Binary and Multi-Categorical) Variables

As shown in the previous subsections that weighted imputation yields better estimates of the missing values. Specifically, $\texttt{wNNSel}_{\texttt{dum}}$ performs better than $\texttt{wNNSel}_{\texttt{cat}}$ in the case of the multi-categorical data, while for binary data both methods perform very similar. In this

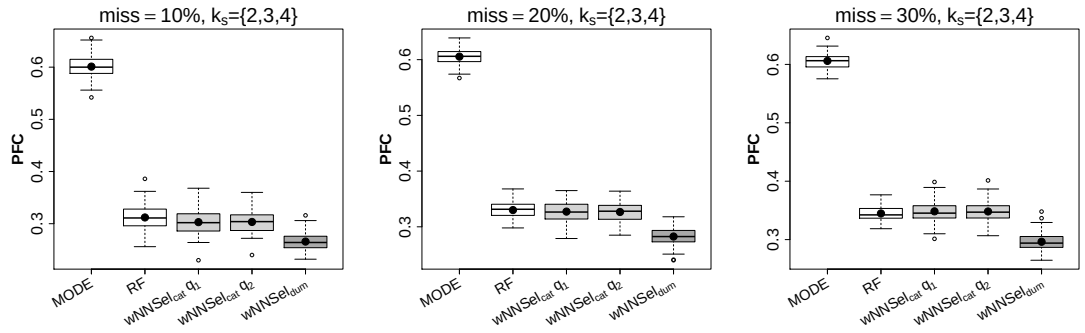


Figure 5.5.: Simulation study for mixed data: Boxplots of PFCs for MCAR missing pattern with binary and multi-categories in the data, $S = 200$ samples were drawn from multivariate normal distribution using autoregressive correlation structures to form the categories. Solid circles within boxes show mean values.

Table 5.3.: Comparison of imputation methods using binary and multi-categorical simulated data

miss	MODE	RF	$\texttt{wNNSel}_{\texttt{cat}}$				$\texttt{wNNSel}_{\texttt{dum}}$
			Gauss.q1	Gauss.q2	Tri.q1	Tri.q2	
10%	0.6011	0.3120	0.3030	0.3034	0.3270	0.3219	**0.2658**
20%	0.6054	0.3301	0.3273	0.3266	0.3518	0.3464	**0.2826**
30%	0.6060	0.3448	0.3484	0.3482	0.3701	0.3648	**0.2963**

section we examine the performance of these methods when the data contains a mixture of binary and multi-categorical variables

We use $k_s = \{2, 3, 4\}$ for $S = 200$ samples of size $n = 100,\ p = 50$ drawn from a multivariate normal distribution using autoregressive correlation structure. One third of the variables selected at random are converted to binary and the rest to $k_s = 3, 4$ categories. Then miss=10%, 20% and 30% of the total values are randomly deleted to create missing values. The rest procedure is the same as in previous subsections. The boxplots of resulting PFCs are shown in Figure 5.5. For mixed data, the smallest imputation errors are obtained by the $\texttt{wNNSel}_{\texttt{dum}}$ procedure. It is interesting to see that the random forest method performs as well as the $\texttt{wNNSel}_{\texttt{cat}}$ method.

The detailed results, using triangular kernel function also, are given in Table 5.3. It is obvious that estimates using the mode yields the worst results as in the previous simulations. The random forest method provides imputation estimates that are closer to $\texttt{wNNSel}_{\texttt{cat}}$. In a comparison of $\texttt{wNNSel}_{\texttt{cat}}$ and random forest methods, $\texttt{wNNSel}_{\texttt{cat}}$ shows slightly better results, except for 30% missing values where the smallest average PFC=0.3484 is obtained by random forest. Overall, the $\texttt{wNNSel}_{\texttt{dum}}$ gives the smallest PFCs in all the simulation settings considered here.

5.6. Applications

The results of simulation studies show that the suggested weighted nearest neighbors imputation methods ($\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$) perform better than other competitors. In this section we apply the imputation methods to real data sets. We use three different data sets with binary, multi-categorical and mixed variables.

SPECT heart data (Binary only)

The dataset describes Single Proton Emission Computed Tomography (SPECT) images. Each of the 267 patients is classified into two categories: normal and abnormal based on $p = 22$ binary feature patterns. Kurgan et al. (2001) discuss this processed data set summarizing about 3000 2D SPECT images.

DNA Promoter gene sequence (Multi-categorical)

The data for promoter instances was used by Harley and Reynolds (1987) and for non-promoters by Towell et al. (1990). The total data set contains sequences of $p = 57$ base pairs from $n = 106$ candidates/samples. Each of the 57 variables can be grouped into one of the four DNA nucleotides; adenine, thymine, guanine or cytosine. The response variable is promoter or non-promoter instances.

Lymphography data (Binary and Multi-categorical)

The data were obtained from n=148 patients suffering from cancer in the lymphatic of the immune system. For each patient, $p = 18$ different properties were recorded on a nominal scale. Nine variables out of 18 are binary and the rest have more than two classes. Based on this information, the patients were classified into one of the four categories; normal, metastases, malign lymph or fibrosis.

In each data set, $miss = 10\%, 20\%, 30\%$ values are randomly deleted and imputation is carried out using mode, random forest, $\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ methods. The imputation error is computed in terms of PFC. The results of 30 independent runs are shown in Figure 5.6.

For the $\texttt{wNNSel}_{\texttt{cat}}$ method, we use the Gaussian and triangular kernel function each for the value $q = 1, 2$ (shown as `Gauss.q1`, `Gauss.q2`, `Tri.q1`, and `Tri.q2` in Figure 5.6) as we intended to explore the behavior of the kernel function and the value of q on the real data sets also. It is seen from the figure that the Gaussian kernel yields smaller PFCs as compared to PFCs obtained by using the triangular kernel for DNA and Lymphography data, while both kernels produce similar results for SPECT data. The value of q does not

affect the results and produces similar PFCs. These findings confirm the simulation results obtained in the previous section.

The strongest difference between the performance of $\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ is seen for the DNA promoter data. For the SPECT heart data which contains only binary variables, neither the kernel function nor the value of q have significant impact on the values of PFCs. The PFCs obtained by $\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ are also nearly similar. These results are consistent with the previous findings from simulation studies on binary data.

The random forest method also performs well for the Lymphography data and produces PFCs smaller than some of the $\texttt{wNNSel}_{\texttt{cat}}$ methods (`Tri.q1`, and `Tri.q2`), although the smallest PFCs are obtained by $\texttt{wNNSel}_{\texttt{dum}}$. Overall, $\texttt{wNNSel}_{\texttt{dum}}$ perform better than $\texttt{wNNSel}_{\texttt{cat}}$ method for multi-categorical data, whereas both methods perform equally well in the case of binary data.

Table 5.4.: Comparison of imputation methods using real data

				Gaussian		Triangular		
Data		MODE	RF	$q=1$	$q=2$	$q=1$	$q=2$	$\texttt{wNNSel}_{\texttt{dum}}$
SPECT	10%	0.3070	0.1802	**0.1632**	0.1641	0.1633	0.1639	0.1662
	20%	0.3124	0.1851	0.1807	0.1810	**0.1806**	0.1809	0.1818
	30%	0.3127	0.2026	0.2011	0.2012	0.2004	**0.1999**	0.2011
DNA	10%	0.6966	0.6803	0.5900	0.5875	0.6350	0.6297	**0.4586**
	20%	0.6962	0.6827	0.6168	0.6136	0.6415	0.6419	**0.4818**
	30%	0.6955	0.6908	0.6335	0.6315	0.6458	0.6455	**0.5062**
Lymphography	10%	0.3915	0.3135	0.2922	0.2934	0.3331	0.3338	**0.2813**
	20%	0.3881	0.3304	0.3152	0.3174	0.3373	0.3411	**0.2965**
	30%	0.3896	0.3511	0.3305	0.3314	0.3398	0.3405	**0.3114**

5.7. Concluding Remarks

We proposed a weighted distance metric based on kernel function to impute missing multi-categorical data. The method uses a distance function, called d_{SelCat}, that utilizes information from other covariates by taking information on association into account. To estimate the tuning parameters, a cross validation algorithm is suggested, which automatically selects the best possible values producing the smallest imputation errors. The procedure does not require a specified value of the number of nearest neighbors (k) and provides as accurate results as the best existing methods. Simulation results show that L_1 and L_2 metrics yield similar results. Moreover, the Gaussian kernel provided smaller imputation errors than the triangular kernel.

To our surprise the simple method, which uses dummy variables and the classical correlation coefficient, showed the best performance. For binary data, both procedures $\texttt{wNNSel}_{\texttt{cat}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ yield similar results, whereas, for multi-categorical data $\texttt{wNNSel}_{\texttt{dum}}$ yields smaller imputation errors. The $\texttt{wNNSel}_{\texttt{dum}}$ method outperforms in simulations as well as in real data application all competitors.

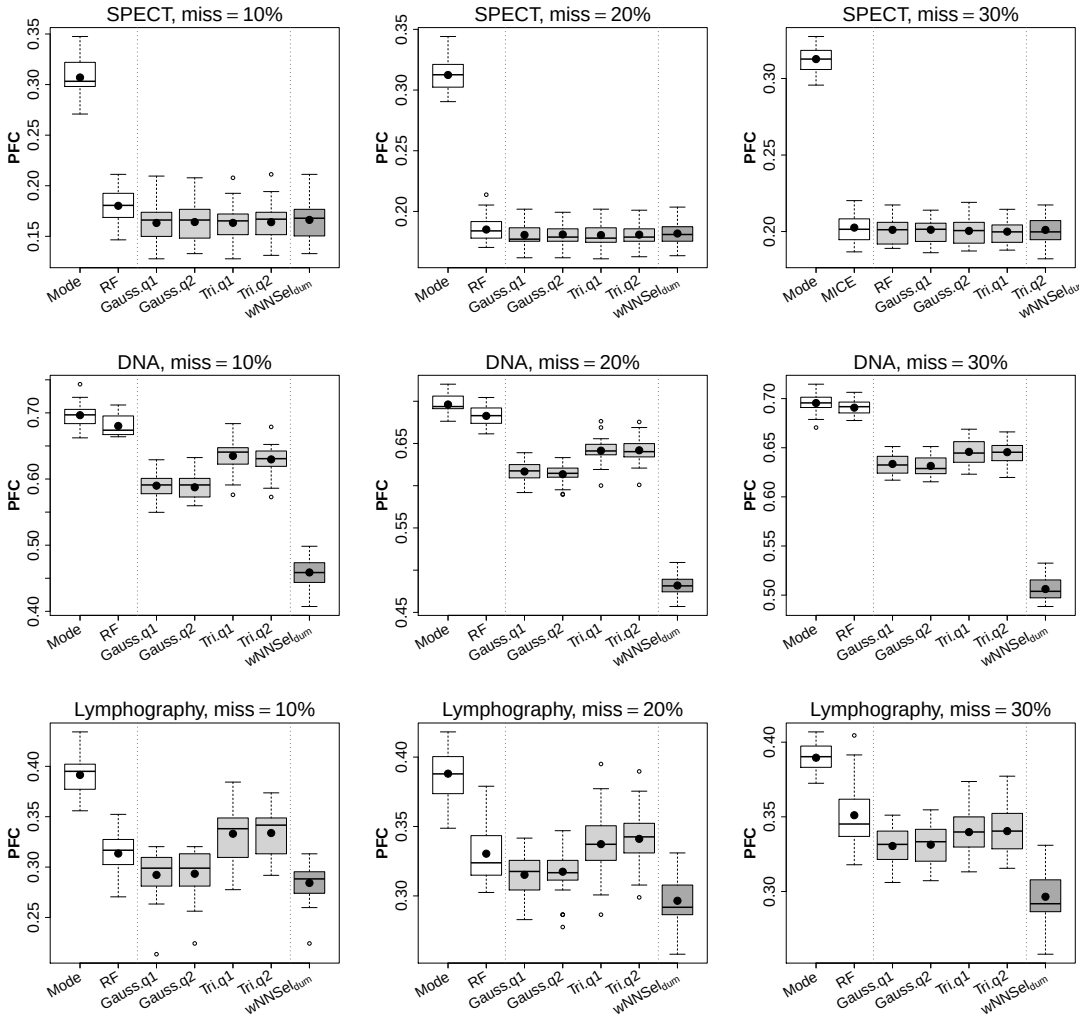


Figure 5.6.: Real data: Boxplots of PFCs obtained by different imputation methods. The SPECT data (upper row), DNA promoter gene sequence data (middle row) and Lymphography data (lower row) with 10%, 20% and 30% missing values is shown. Grey boxes show the proposed $\texttt{wNNSel}_{\texttt{cat}}$ and dark grey show $\texttt{wNNSel}_{\texttt{dum}}$ method. Solid circles within boxes show mean values.

6. Imputation Methods for High-Dimensional Mixed-Type Datasets by Nearest Neighbors

Abstract

In modern industrial and biomedical research, the data often contains a large number of variables of mixed data types (continuous, multi-categorical or binary) but the information on some variables is missing. Imputation is a common solution when the downstream analyses require a complete data matrix. A number of imputation methods are available that work under specific distributional assumptions. We propose an improvement over the popular nonparametric nearest neighbor imputation method which requires no particular assumptions. The proposed method makes practical and effective use of the information on association among the variables. In particular we propose a weighted version of the L_q distance for mixed-type data, which uses the information from a subset of important variables only. The performance of the proposed method is investigated using a variety of simulated and real data from different areas of application. The results show that the proposed methods yield a smaller imputation error and better performance when compared to other approaches. It is also shown that the proposed imputation method works efficiently even when the number of samples is smaller than the number of variables.

6.1. Introduction

Missing values are a common phenomenon in practical research. A number of methods is available that can be used to fill the missing values in metrically scaled data, such as k nearest neighbors imputation (Troyanskaya et al., 2001) and its various adaptations that rely on the basic concept to impute values by building averages over qualifying neighbors, see, for example, Ouyang et al. (2004), Kim et al. (2004), Sehgal et al. (2005), and Scheel et al. (2005). Other variants include the local least squares (LLSimpute) method of Kim et al. (2005), the sequential local least squares (SLLSimpute) method of Zhang et al. (2008), and the iterative LLS (ILLSimpute) of Cai et al. (2006). Other imputation methods require the specification of a joint distribution for the data, for example, Schafer (1997) assumes the

multivariate normal distribution for the data. Imputation methods that use distributional assumptions can be very restrictive in practice.

The missing values in categorical data can be imputed with non-parametric methods such as hot-deck (Rubin, 1987) and k-nearest neighbors (Schwender, 2012) or with model-based parametric methods. The log-linear model is an appropriate choice for categorical data (Schafer, 1997). The simulation studies of Ezzati-Rice et al. (1995) and Schafer (1997) showed that it provides an attractive solution for missing categorical data problems. However, its use is restricted to cases with a small number of attributes (Erosheva et al., 2002) since model selection and fitting becomes very challenging for larger dimensions. Some existing methods to impute attributes are based on the mode or weighted mode of the k nearest neighbors (Liao et al., 2014).

These approaches are limited to handle only one type of data that is continuous or categorical. The literature on the imputation of mixed data is scarce. By mixed data, we mean that some of the variables are continuous, and others are binary or multi-categorical. To deal with mixed data is rather demanding, because it requires to pay attention to the relationships among covariates measured on different scales. One solution is to construct the indicator matrix of dummy variables from the categorical covariates and treating them as continuous along with metric covariates. But the assumptions for continuous variables do not hold for dummy variables (Audigier et al., 2016).

A maximum likelihood based approach was presented by Little and Schluchter (1985). It combines the multivariate normal model for metric and multinomial model for categorical data. This idea was further extended by Little and Rubin (1987), see also, Li (1988), Rubin and Schafer (1990). Schafer (1997) proposed to combine the multivariate normal model and log-linear models for mixed type data but this approach has the advantages and limitations of such models.

The imputation by chained equations (Van Buuren and Oudshoorn, 1999; Van Buuren, 2007) combines different regression models for the metric and categorical variables in a more refined way. See also, Raghunathan et al., 2001. Stekhoven and Bühlmann (2012) have recently proposed an imputation method based on random forest by Breiman (2001). This approach has increasing popularity as it has many attractive features including handling of mixed type data. It has been shown to provide good imputation results regardless of the number of variables, their types, and relationship among variables.

One has to deal with a combination of continuous and nominal variables in many real world applications, therefore the methods to impute mixed data become more important. Since multiple imputation techniques fail for high-dimensional missing data, the nonparametric single imputation methods are gaining more popularity. We propose an improved version of the popular nonparametric nearest neighbors method which uses information only on potentially relevant neighbors. More specifically, we introduce a distance function that is appropriate for mixed data. It is an extension of Tutz and Ramzan (2015) and uses information on association among variables. A particular advantage of the proposed method is that

it simultaneously takes into account the similarities between samples and the relationships between covariates.

The paper is organized as follows: Section 6.2 includes the proposed weighted distance that uses information on correlation/interaction among covariates. The imputation procedure based on weighted nearest neighbors is also given in this section. Section 6.3 describes the benchmark methods and the measures used for evaluation of the performance of different imputation methods. Using simulated data, properties of the proposed method are investigated and performance is compared with existing methods in Section 6.4. The imputation method is applied on some real data sets in Section 6.5. Finally, Section 6.6 concludes some concluding remarks.

6.2. Distances for Mixed-Type Data

Nearest neighbors imputation requires distance/similarity measures that define the degree of nearness of observations. The performance of nearest neighbors imputation depends crucially on the distance used to find these neighbors. In this section, we propose a weighted distance measure for mixed data for the imputation of missing values.

6.2.1. Available Methods

A straightforward approach to handle mixed types of variables is to split the variables into types and confine the analysis to the dominant type (Anderberg, 1973). The judgment of *dominant* type may depend on several factors such as the number of variables in each type, background theory, some variable may be of interest to the analyst etc. It is not a recommended practice since it discards important and relevant information. Moreover, one may be tempted to ignore the differences among different types of variables and use a distance function which is suitable for one type but inappropriate for other types of the variables.

Another simple way to deal with the mixture type of variables is to convert one type of covariate to another, while retaining the maximum possible information, and, then use a distance measure which is suitable for the selected type of covariates. Anderberg (1973, p. 94) argued over the natural question which type should be chosen as the single type for the analysis. For instance, metric variables can be converted into binary variables using a fixed level. Consequently, many observations would be more similar to each other, resulting in the reduction of the influence of metric variables. Alternatively, the nominal variables can be converted into binary variables (see, for example, (Frank and Todeschini, 1994, p. 92)). In specific cases, for example if a continuous variables is converted to binary or ordinal scale, the conversion results in a loss of important information contained in the data (Tarsitano and Falcone, 2011).

Some measures for computing the distance between the observations of mixed type data, have been proposed in the literature. When the data contains a mixture of variable types, the distance between observations i and j is calculated as $d_{ij} = w_1 d^1_{ij} + w_2 d^2_{ij} + w_3 d^3_{ij}$, where d^1, d^2, d^3 are the distances calculated separately for the binary, multi-categorical and continuous variables, respectively. The w_1, w_2, w_3 are weighting coefficients in the aggregate distance measure. An important issue with this type of distance is how to determine the weights. One may use an equal or proportional weight for each type of variables or a priori judgment could also be used. The Gower's distance (Gower, 1971) is a weighted average of three different measure of dissimilarity where the weights are the proportions of each type of variable. It is a widely used distance measure based on a general similarity measure and can be used for different types of variables.

6.2.2. Weighted Distance

Let $\boldsymbol{\mathcal{R}} = (R_{is})$ be the $n \times (p+m)$ data matrix with p continuous and m categorical covariates defined by $\boldsymbol{\mathcal{R}} = (\boldsymbol{X}, \boldsymbol{Z})$, where $\boldsymbol{X} = (x_{ig})$, $g = 1, \cdots, p$, with x_{ig} denoting the i^{th} observation of the g^{th} continuous covariate, and $\boldsymbol{Z} = (z_{il})$, $l = 1, \cdots, m$, with z_{il} denoting the i^{th} observation of the l^{th} categorical covariate. Let $\boldsymbol{O} = (o_{is})$ be the corresponding $n \times (p+m)$ matrix with $o_{is} = 1$ if R_{is} is observed, otherwise $o_{is} = 0$.

The categorical observations z_{il} in the data matrix, can take values $c_l \in \{1, \ldots, k_l\}, l = 1, \ldots, m$, where k_l is the number of categories that the l^{th} attribute can take. It is to note that the value of $k_l = 2$ shows that the l^{th} variable is binary whereas $k_l > 2$ shows a multi-categorical variable. Then the i^{th} row of the data matrix $\boldsymbol{\mathcal{R}}$ can be written as

$$\boldsymbol{R}_i^T = (\boldsymbol{x_i}^T, \mathbf{z_i}^T)$$

with $\boldsymbol{x_i}^T = (x_{i1}, \cdots, x_{ip})$ and $\mathbf{z_i}^T = (z_{i1}, \cdots, z_{im})$.

For the computation of distances, the categorical variables are transformed into binary variables. Thus the observation z_{il} becomes a vector, $\mathbf{z}_{il}^T = (z_{il1}, \ldots, z_{ilk_s})$ with $z_{ilr} = 1$ if $Z_{il} = r$. The dummy vectors $\mathbf{z}_{il}^T$ for a nominal variable with four categories can be written as

category	z_{il1}	z_{il2}	z_{il3}	z_{il4}
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Thus, the i^th row of the transformed matrix $\boldsymbol{\mathcal{R}}^D$ has the form

$$[(x_{i1}, \ldots, x_{ip}), (\mathbf{z}_{i1}^T, \cdots, \mathbf{z}_{im}^T)]^T$$

with dummy vectors $\mathbf{z}_{il}$, $l = 1, \cdots, m$.

Let R_{is} be a missing entry in the data matrix $\boldsymbol{\mathcal{R}}$, that is $O_{is} = 0$. Then the distance between the i-th and the j-th rows specific for a missing value in the sth covariate is defined by

$$d(\boldsymbol{R}_i, \boldsymbol{R}_j) = \left(\gamma_1 \sum_{g=1}^{p} |x_{ig} - x_{jg}|^q I_{(o_{ig}=1)} I_{(o_{jg}=1)} + \gamma_2 \sum_{l=1}^{m} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|^q I_{(o_{il}=1)} I_{(o_{jl}=1)} \right)^{1/q}, \quad (6.1)$$

where $I_{(.)}$ is the indicator function and γ_1 and γ_2 are weights satisfying $\gamma_1 + \gamma_2 = 1$.

6.2.3. Weighted Distance With Selection of Variables

The distance in the equation (6.1) uses all the predictors/covariates. In high-dimensional settings, imputation suffers from curse of dimensionality. A more convenient way is to use the *relevant* covariates only, which is done by introducing an additional weight function in the distance formula.

$$\begin{aligned} d(\boldsymbol{R}_i, \boldsymbol{R}_j) \quad = \quad & \left(\gamma_1 \sum_{g=1}^{p} |x_{ig} - x_{jg}|^q I_{(o_{ig}=1)} I_{(o_{jg}=1)} . C(\delta_{sg}) + \right. \\ & \left. \gamma_2 \sum_{l=1}^{m} \sum_{c=1}^{k_l} |z_{ilc} - z_{jlc}|^q I_{(o_{il}=1)} I_{(o_{jl}=1)} . C(\delta_{sl}) \right)^{1/q}, \end{aligned} \quad (6.2)$$

where $C(.)$ is a convex function defined on the interval $[-1, 1]$ which transform the correlation into weights. Thus the covariates having higher correlation with the sth variable will get higher weight thus contribute more to the computation of the distances. We use $C(\delta_{sl}) = |\rho_{sl}|^{\omega}$, where ω is a tuning parameter, and δ_{sl} is a measure of association between covariates s and l.

Selection of weights γ_1 and γ_2

The weights γ_1 and γ_2 can be determined on the basis of a *prior* judgment. But if the investigator has no knowledge about which type of covariates are important or to be prioritized, the assessment is not be feasible. A simpler approach is to use an equal weighting for all data types (Chiodi, 1990), that is, to use $\gamma_1 = \gamma_2 = 0.5$. Romesburg (2004) used weights that are proportional to the number of variable in each type. The proportional weighting would be the right choice if all the variables are supposed to be equally important, irrespective of the scale on which they are measured.

We propose to consider $\gamma_1,\ \gamma_2$, as tuning parameters and select their value by cross validation. The procedure for cross validation is given in the next section.

6.2.4. Weighted Imputation by Nearest Neighbors

Using information on all nearest neighbors does not necessarily provide better imputation results and for larger data it would also consume more time. Alternatively, a weighted nearest neighbors method based on kernel function is proposed. More specifically, the weights we are using are defined by

$$w(\boldsymbol{R}_i, \boldsymbol{R}_j) = \frac{k(d(\boldsymbol{R}_i, \boldsymbol{R}_j)/\lambda)}{\sum_{l=1}^{k} k(d(\boldsymbol{R}_i, \boldsymbol{R}_j)/\lambda)}, \tag{6.3}$$

where $K(.)$ is a kernel function and λ is a tuning parameter. We use Gaussian kernels as in the initial simulations it yielded smaller imputation errors. The tuning parameter λ is chosen by cross validation.

Let a value is missing in the i^{th} row of the data matrix $\boldsymbol{\mathcal{R}}$. There are two possibilities: (i) metric covariate has the missing value (ii) a categorical covariate has the missing value. One finds the k nearest neighbor observation vectors $\boldsymbol{R}_k$ based on the distances

$$\boldsymbol{R}^D_{(1)}, \ldots, \boldsymbol{R}^D_{(k)} \quad \text{with} \quad d(\boldsymbol{R}_i, \boldsymbol{R}_{(1)}) \leq \cdots \leq d(\boldsymbol{R}_i, \boldsymbol{R}_{(k)})$$

where the distances are computed using equation (6.2). The computation of the imputed value depends on the type of variable and computed differently for metric and categorical variable.

Imputing Categorical Missing Value

Let z_{is} be the missing value in the i^{th} row and s^{th} categorical variable. For the imputation of the values z_{is}, we use the weighted estimator

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\boldsymbol{R}_i, \boldsymbol{R}_{(j)}) z_{(j)sc}, \tag{6.4}$$

where $c = 1, \ldots, k_s$. The weighted imputation estimate of a categorical missing value z_{is} is

$$\hat{z}_{is} = \arg\max_{c=1}^{k_s} \hat{\pi}_{isc}, \tag{6.5}$$

i.e. the value of $c \in \{1, \ldots, k_s\}$ with highest value of $\hat{\pi}$.

Imputing Continuous Missing Value

Let x_{is} be the missing value in the i^{th} row and s^{th} continuous variable. The weighted imputation estimate of continuous missing value, x_{is}, is defined by

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\boldsymbol{R}_i, \boldsymbol{R}_{(k)}) x_{(j)s} \tag{6.6}$$

The proposed $\texttt{wNNSel}_{\texttt{mix}}$ procedure is described in the following.

Procedure $\texttt{wNNSel}_{\texttt{mix}}$**:**

- Locate a missing value in the data matrix $\boldsymbol{\mathcal{R}}$
- Rank samples (rows) of the matrix $\boldsymbol{\mathcal{R}}^D$ based on d_{MixSel}.
- Compute weights using equation (6.3).
- if missing value belongs to categorical variable
  - compute probabilities by using equation (6.4)
  - estimate the missing value using equation (6.5).
- if missing value belongs to continuous variable
  - Compute the missing value using equation (6.6).
- Seek the next missing value in $\boldsymbol{\mathcal{R}}$ and repeat these steps until all missing values in $\boldsymbol{\mathcal{R}}$ have been imputed.

6.2.5. Choice of Tuning Parameters by Cross-Validation

The imputation procedure $\texttt{wNNSel}_{\texttt{mix}}$ requires pre-specified values of the tuning parameters γ_1, ω and λ. In this section we present a cross validation algorithm that automatically choses those values for which the imputation error is minimum.

To estimate the tuning parameters, we set some available values $(o_{is}) = 1$ in the data matrix as missing $(o_{is}) = 0$. The advantage of this step is that these values are used to estimate the tuning parameters. The procedure of cross validation to find optimal tuning parameters is given in the following as algorithm 6.1.

Here, $\psi_{(\gamma_1,\omega,\lambda)}$ is the measure of imputation error for specific values of the tuning parameters γ_1, ω and λ. We use the mean squared imputation error (MSIE) to obtain the optimal values

Algorithm 6.1 Cross-validation for `wNNSelmix`

Require: $\boldsymbol{\mathcal{R}}$ an $n \times (p+m)$ matrix, number of validation sets t, range of suitable values for tuning parameters $\mathcal{G}_1$, $\mathcal{W}$ and $\mathcal{L}$

$\boldsymbol{\mathcal{R}}^{\text{cv}} \leftarrow$ initial imputation using unweighted 5-nearest neighbors
for $t = 1, \ldots, \mathcal{T}$ **do**
  $\boldsymbol{\mathcal{R}}^{\text{cv}}_{\text{miss,t}} \leftarrow$ artificially introduce missing values to $\boldsymbol{\mathcal{R}}^{cv}$
  for $\gamma_1 \in \mathcal{G}_1$ **do**
    for $\omega \in \mathcal{W}$ **do**
      for $\lambda \in \mathcal{L}$ **do**
        $\boldsymbol{\mathcal{R}}^{\text{cv}}_{\texttt{wNNSel}_{\texttt{mix}},t} \leftarrow$ imputation of $\boldsymbol{\mathcal{R}}^{\text{cv}}_{\text{miss,t}}$ using $\texttt{wNNSel}_{\texttt{mix}}$ procedure
        $\psi_{(\gamma_1,\omega,\lambda),t} \leftarrow$ imputation error of $\texttt{wNNSel}_{\texttt{mix}}$ procedure using γ_1, ω & λ
      end for
    end for
  end for
  Determine $(\gamma_1, \omega, \lambda)_{\text{best}} \leftarrow \text{argmin } \frac{1}{T} \sum_{t=1}^{T} \psi_{(\gamma_1,\omega,\lambda),t}$
end for
$\boldsymbol{\mathcal{R}}^{\text{imp}} \leftarrow \texttt{wNNSel}_{\texttt{mix}}$ imputation of $\boldsymbol{\mathcal{R}}$ using $(\gamma_1, \omega, \lambda)_{\text{best}}$

with $\gamma_1 \in \mathcal{G}_1 = (0, 1]$, $\omega \in \mathcal{W} = \{2, 4, 6\}$, $\lambda \in \mathcal{L} = (0, 1]$ and a split into five validation sets.

6.2.6. Measuring Association Among Mixed Variables

The association among covariates typically depends on their measurement scales. As the proposed method requires the information of the association among covariates, it is crucial to separately describe the association measures. We use already established statistical measures to compute the strength of relationship among different types of variables. Table 6.1 the association measures for each variable type are listed.

Table 6.1.: Correlation measures for different types of covariates

Type	Binary	Multi-categorical	Continuous
Binary	Phi	—	—
Multi-categorical	Cramer's V	Cramer's V	—
Continuous	Point Biserial	Point Biserial extension	Pearson

Phi coefficient (ϕ)

For nominal variables with only two categories, a simple measure of association is the ϕ-coefficient defined by

$$\phi_{sl} = \sqrt{\frac{\chi^2_{sl}}{n}}.$$

where n is the number of subjects and χ^2_{sl} is the chi-square statistic for the 2×2 contingency table of the two binary variables.

Cramer's V

If the covariates have different number of categories ($k_s \neq k_l$), Cramer's V is an attractive option Cramér (1946). It is defined by

$$\text{Cramer's V} = \sqrt{\frac{\chi^2_{sl}/n}{min(k_s - 1, k_l - 1)}},$$

where $n = k_s \times k_l$ is the total number of cells in the contingency table.

Point Biserial Correlation

Point biserial correlation between a continuous variable x_g and a binary variable Z_s is defined as

$$\text{Point Biserial Corr } = \frac{\bar{x_g}_1 - \bar{x_g}_0}{S_{x_g}/\sqrt{p_{Z_s} \times (1 - p_{Z_s})}}$$

where $\bar{x_g}_1$ and $\bar{x_g}_0$ show the means of x_g given $Z_s = 1$ and 0 respectively. S_{x_g} is the standard deviation of x_g and p_{Z_s} is the proportion of samples/rows with $Z_s = 1$.

The point biserial correlation can be generalized, for the case when the variable Z_s has more than two categories. We implemented using the function `biserial.cor` in the `R` package `ltm` Rizopoulos (2006).

6.2.7. A Pearson Correlation Strategy

As an alternative we consider a strategy that uses the dummy variables directly. The principle consists in converting the categorical into dummy variables. Starting from the matrix of dummies $\boldsymbol{\mathcal{R}}^D$ we use the Pearson correlation coefficient between dummy variables

as association measure. The weighting scheme remains the same. The imputation is again determined by

$$\hat{\pi}_{isc} = \sum_{j=1}^{k} w(\boldsymbol{R}_i, \boldsymbol{R}_{(j)}) z_{(j)sc},$$

Although $\hat{\pi}_{isc}^T = (\hat{\pi}_{is1}, \ldots, \hat{\pi}_{isk_s})$ might not be a vector of probabilities, simple standardization by setting

$$\tilde{\pi}_{isc} = \hat{\pi}_{isc} / \sum_{r=1}^{k_s} \hat{\pi}_{isr}$$

yields a vector of probabilities that can be used to determine the mode. The method can be seen as an adaptation of the weighting method proposed by Tutz and Ramzan (2015). To apply this method, only small modifications are required. One is that the now the missing value of a variable would refer to a vector of dummies. For this method we use the Gaussian kernel and the Euclidean distance throughout our evaluations (Faisal and Tutz, 2017c). We denote the method by $\texttt{wNNSel}_{\texttt{dum}}$.

6.3. Existing Approaches for Comparison

In this section, we briefly describe some existing approaches to impute missing values in mixed type data and performance measures used for comparison of these methods.

kNN Imputation

This method was originally proposed by Troyanskaya et al. (2001) for the imputation of missing values. In this approach, an imputed value is obtained by taking the average of the values of k candidate samples, called neighbors, chosen based on a distance measure. For this method to perform, one needs to select the suitable value of k, and a distance measure. For mixed type data, Gower's distance (Gower, 1971) is used. We use the `R` package `VIM` for implementation of this method (Templ et al., 2016).

Random Forests

Random forests were introduced by Breiman (2001) as an extension of classification and regression trees (CART). The approach uses bootstrap samples of the data to build various trees. It can accommodate not only mixed types of predictors with nonlinear and complex relationships, but also deal with high-dimensional data ($p \gg n$). A random subset of the variables is used as candidates at each split. To build the trees, bootstrap aggregation (bagging) as well as random variable selection can be used. The recently introduced `missForest` procedure (Stekhoven and Bühlmann, 2012) uses the versatile and flexible random forests model to obtain an estimate of a missing value. It is an iterative procedure that uses initial

imputations using mean imputation or another imputation method, and then improves the imputed data matrix on successive iterations. A random forest model is developed for each predictor with a missing value by using the rest of the predictors, and this model is used to estimate the missing value of that predictor. The imputed data matrix is updated, and the difference between previous and new imputation is assessed at the end of each iteration. The whole process is repeated until a specific criterion is met. We use the `R` package `missForest` (Stekhoven and Bühlmann, 2012) for this method.

6.3.1. Performance Measures

To compare the performance of imputation procedure, we compute the imputation error separately for categorical and continuous variables. For the categorical variables, the proportion of falsely imputed categories is computed, whereas for the continuous variables the mean of squared imputation errors is used as performance measure.

$$\text{MSIE}_{cont} = \frac{1}{\sum I_{o_{is}=0}} \sum_{x_{is}:o_{is}=0} (x_{is} - \hat{x}_{is})^2,$$

$$\text{PFC} = \frac{1}{\sum I_{o_{is}=0}} \sum_{z_{is}:o_{is}=0} I(z_{is} \neq \hat{z}_{is}),$$

where $I(.)$ is an indicator function, x_{is} and z_{is} are the true values in the complete data matrix, $\hat{x}_{is}$ and $\hat{z}_{is}$ are the corresponding imputed values. These two measures are combined to get the final estimate of the imputation error.

$$\text{MSIE} = \text{MSIE}_{\text{cont}} + \text{PFC}$$

A higher value of MSIE would indicate a higher discrepancy among imputed and true data. Moreover, when MSIE equals zero, the imputation is perfect. An alternative option is to compute mean absolute imputation error (MAIE), by taking absolute values instead of square in the above formula. In our evaluations, we computed both MSIE and MAIE as criteria for comparison of different imputation methods.

6.4. Simulation Studies

In this section, we perform simulation studies to investigate different properties of the proposed imputation method using a variety of simulation settings.

First, we conduct a small simulation study to investigate which kernel function, value of q and the convex function perform better. We generated $S = 200$ samples of size $n = 100$ for $p = 30$ predictors drawn from a multivariate normal distribution with $N(\mathbf{0}, \mathbf{\Sigma})$. The correlation matrix $\mathbf{\Sigma}$ has an autoregressive type of order 1 with $\rho = 0.9$. Some

randomly selected variables are converted to nominal scale. We construct categories from the continuous data by setting cut points. In each sample, $miss = 10\%, 20\%, 30\%$ of the total values were replaced by missing values completely at random (MCAR). We investigate the performance of different kernel functions, values of q (q =1,2) and two convex functions (linear and power). The results (not given here) showed that the Gaussian kernel for $q = 2$ using power function provides smallest imputation error. The findings are consistent to those of Tutz and Ramzan (2015) and Faisal and Tutz (2017c). We had to perform this simulation because the structure of the data is now different (categorical and binary variables are also included). We will use these findings to compare the performance of $\texttt{wNNSel}_{\texttt{mix}}$ with other existing methods.

Relationship between Different Types of Variables

The main objective of the proposed distance formula (6.3) is to take into account the relationships between categorical and continuous covariates. We believe that considering both types of covariates during imputation will improve the results.

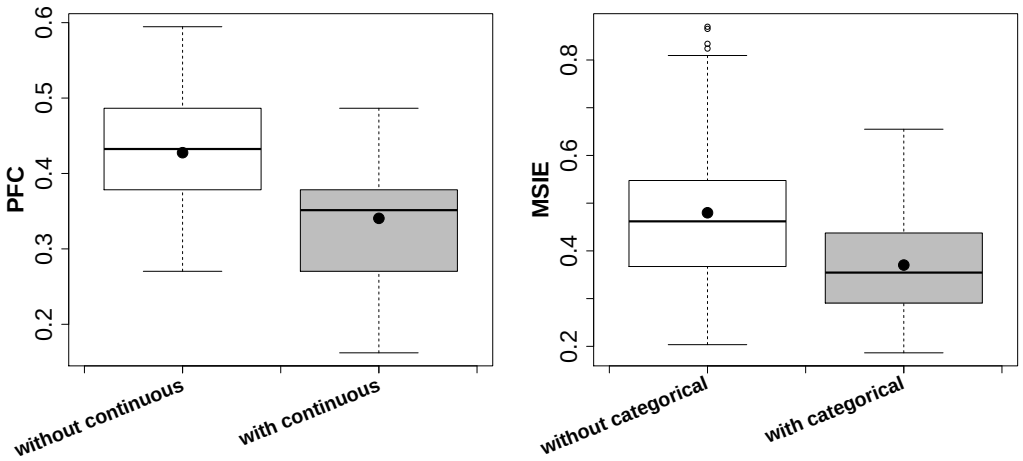
We conduct a simulation study to investigate whether a variable contributes to imputing the missing values of a variable with different scale of measurement. We consider $p = 30$ covariates, half of which are continuous and half are multi-categorical having $\{3, 4\}$ categories. In setting (a), we delete some values completely at random in categorical variables only. We impute these missing categorical values without using information from continuous variables (Figure 6.1 left panel: white box) and then using information of both types of covariates (Figure 6.1 left panel: grey box). The same process is repeated for continuous variables and the results are shown in Figure 6.1 right panel. It is clear from the Figure that taking into account both types of variables improves the imputation results.

(a) Imputing categorical variables
   - Only categorical variables contribute to the imputation
   - Both, categorical and continuous, variables contribute to the imputation

(b) Imputing continuous variables
   - Only continuous variables contribute to the imputation
   - Both, categorical and continuous, variables contribute to the imputation

Effect of the Number of Covariates

The imputation of missing values becomes very complex as the number of covariates increases. The following subsection includes some simulations to investigate the effect on imputation results when the number of covariates increases. We set $p = 15, 30, 45, 60$ variables for $n = 50$ samples. In each sample 66% of the variables are converted to categorical variables and the rest are taken as continuous.



(a) Imputing categorical variables (b) Imputing continuous variables

Figure 6.1.: Boxplots of average imputation errors for imputing categorical (left) and continuous (roght) missing values. Solid circles within boxes show mean values.

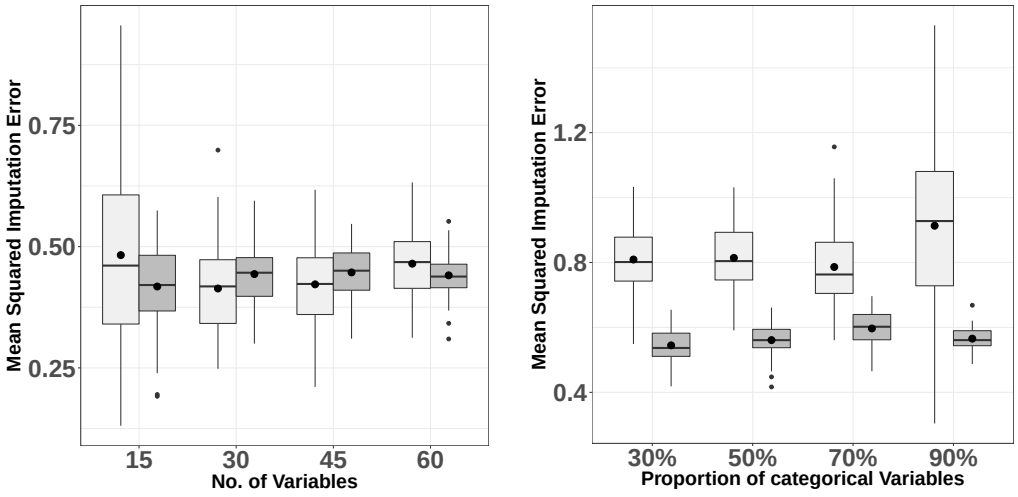
Figure 6.2(a) shows the imputation errors for continuous (light grey boxes) and for categorical (dark grey boxes) variables separately. It is clear that the performance does not deteriorates with increasing number of covariates. The imputation errors for continuous and categorical variables remains stable even when the number of predictors are higher than the number of samples.

Effect of the Proportion of Categorical Covariates

The data in practice may include different number of categorical and metric variables. For example, survey data typically contain on a large number of categorical variables with only a few covariates on continuous scale. In the following, we investigate whether the proportion of the number of categorical variables has any effect on the imputation results. We set the proportion of categorical variables as $30, 50, 70, 90\%$ of the total variables. Figure 6.2(b) compares the distributions of imputation errors for continuous (light grey) and categorical (dark grey) variables. It is noted from the figure, whatever is the proportion of categorical variables, the proposed weighted neighbors method performs well.

Effect of Correlation Among Covariates

The proposed weighted nearest neighbors imputation method imputes a missing value by using the information from other covariates. This method should perform better when the covariates are highly associated. For this purpose, we use two different structures of correlation, namely blockwise and AR(1) type correlation. We select $n = 50$ samples for $p = 45$ covariates from a multivariate normal distribution with $N(\mathbf{0}, \mathbf{\Sigma})$. For the AR(1)



(a) (b)

Figure 6.2.: Boxplots of average MSIE obtained by $\texttt{wNNSel}_{\texttt{mix}}$ method for different number of predictors. Light grey denotes continuous and dark grey for categorical variables. Solid circles within boxes show mean values.

correlation structure, we set $\rho = 0.2, 0.5, 0.8$ and for blockwise, we set $\rho_w = 0.2, 0.5, 0.8$ and $\rho_b = 0.1$. We set 30 out of 45 covariates as categorical.

The imputation results for the random forest (RF), $\texttt{wNNSel}_{\texttt{mix}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ methods are given in Figure 6.3. The results obtained from AR(1) correlation structure are shown in the left panel of Figure 6.3 and blockwise correlation structure are shown in the right panel. Clearly, the proposed methods yield smaller imputation error than the random forests method. In particular, the $\texttt{wNNSel}_{\texttt{dum}}$ method yields promising results in the data settings. It is also seen from the Figure that when the correlation among variables increases, the quality of imputation also increase as expected. Thus the relationship (strength of association) among variables plays an important role in improving the imputation results and the $\texttt{wNNSel}_{\texttt{mix}}$ method nicely captures this information for imputing the missing values.

6.5. Real Data Applications

In this section, we use some real data sets to investigate the performance of imputation methods. We use a variety of data comprising number of samples, number of predictors, proportion of metric to nominal predictors, number of categories for the categorical variables. Moreover, these data sets are taken from different areas of application. We chose only complete data so that the true imputation errors would be possible to compute. If the dataset has natural missing values, we discard the incomplete rows. The missing values are then introduced to each data completely at random with a specific level. The data is imputed using traditional k-nearest neighbors (shown as VIM), the random forest method

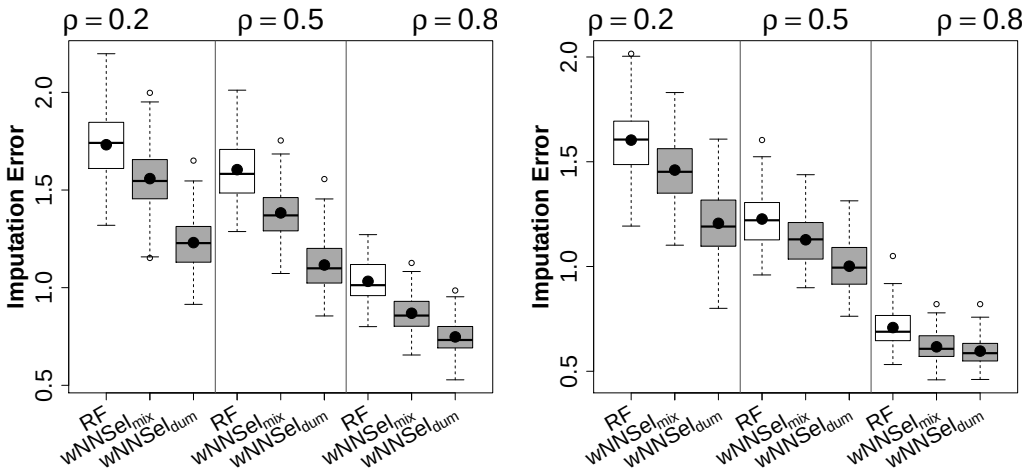


Figure 6.3.: Boxplots of average MSIEs for low ($\rho = 0.2$), medium ($\rho = 0.5$) and high ($\rho = 0.8$) correlation using autoregressive (left) and block-wise (right) structure of correlation matrix. White boxplots correspond to the imputation error obtain by random forest (RF) method and gray boxplots to the imputation error for proposed methods ($\texttt{wNNSel}_{\texttt{mix}}$ and $\texttt{wNNSel}_{\texttt{dum}}$). Solid circles within boxes show mean values.

(shown as `RF`) and the two proposed methods $\texttt{wNNSel}_{\texttt{mix}}$ and $\texttt{wNNSel}_{\texttt{dum}}$. We use a missing percentage of 10%, 20% and 30% in our simulations. Each configuration is repeated $S = 200$ times for three percentages of missing data.

In the following, we briefly describe the datasets used in our evaluations.

Table 6.2.: Data sets

Data set	Total Samples	used samples	Total variables	binary	Nominal	Continuous
Cars	93	82	24	2	4	18
GBSG2	686	100	10	1	2	7
Hepatitis	155	155	19	13	0	6
BMI	152	152	6	2	0	4
Automobile	205	155	24	3	6	15

Cars Data

This data contains the sales characteristics of cars in USA in the year 1993. Cars were selected at random from among 1993 passenger car models that were listed in both the consumer reports issue and the PACE buying guide. Pickup trucks and sport/utility vehicles were eliminated due to incomplete information in the consumer reports source. The data has a total of 24 covariates, among them two are binary, 4 are multi-categorical with 3 or 6 categories and remaining are considered as continuous. Further detail on the data can be found in (Lock, 1993).

German Breast Cancer Study Group 2: GBSG2

This data is taken from the `R` package `ipred`. The complete data contains information on 686 women having breast cancer for $p = 10$ covariates. we select a random sample of $n = 100$ women. Out of 10 predictors, seven are continuous.

Hepatitis

The data contains information of $p = 19$ variables obtained from a sample of 155 patients of the disease hepatitis. Among 19 predictors, 13 are binary and the remaining 6 are continuous in nature. The response variable is binary. The original data contains natural missing values, but we removed the cases with incomplete information.

Body Mass Index: BMI

This data contains body mass index of French children aged 3 to 4 (De Micheaux et al. (2011)). A total of 6 variables concerning morphology and the characteristics of their kindergarten were measured from $n = 152$ children. Two out of six variables are binary and the remaining four are continuous in nature.

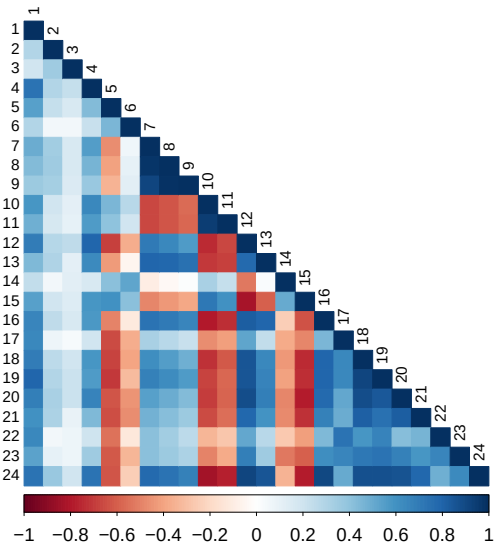
Automobile data

This data set consists of three types of entities: (a) the specification of an auto in terms of various characteristics, (b) its assigned insurance risk rating, (c) its normalized losses in use as compared to other cars. Full data contains 205 samples but due to missing values exclusion only $n = 155$ samples contain complete information for 24 covariates.

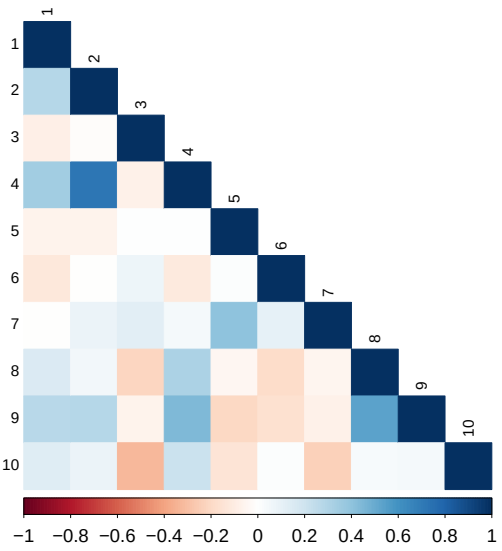
Results

The correlation among covariates in all the data sets are shown in Figure 6.4. The color follows a gradient according to the strength of the correlation. The color varies from red (negative correlation) to blue (positive correlation). The stronger the strength of the relationship, the darker is the color and vice versa. It is clear from the plots that the covariates in the cars and automobile data are strongly associated. For cars data, the correlation measure between some variables is very high, sometimes even closer to ± 1. Whereas the covariates, in BMI data have a week association and a moderate association can be seen among the covariates of GBSG2 and Hepatitis data.

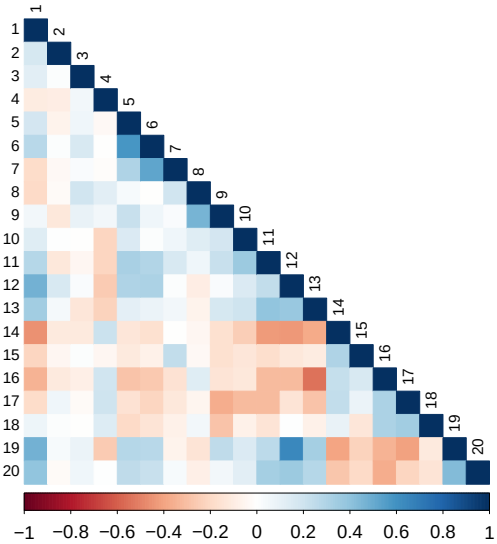
Now we randomly delete 10%, 20% and 30% of the total values in each the data sets. The imputation is performed using the traditional k-nearest neighbors (shown as `VIM`), the random forest method (shown as `RF`) and the two proposed methods $\texttt{wNNSel}_{\texttt{mix}}$ and



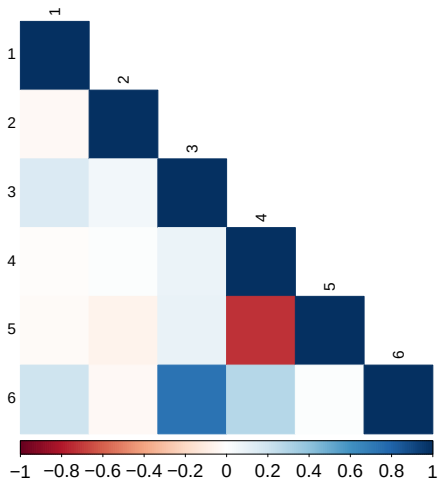
Cars



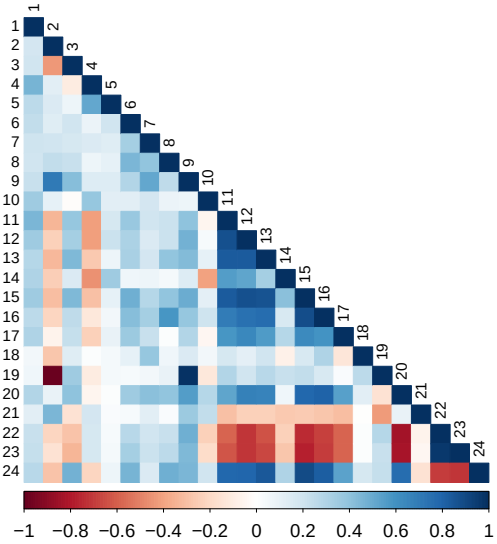
GBSG2



Hepatitis



BMI



Automobile

Figure 6.4: Correlation matrices of real data sets

$\texttt{wNNSel}_{\texttt{dum}}$. The MSIEs and MAIEs are computed in each setting. The whole process is repeated $S = 200$ times for each of the three missing percentages. The result for MSIEs obtained are plotted in Figure 6.5 for all the data sets.

For cars data, `VIM` provides highest MSIEs as it does not uses the information of association among the covariates. Moreover, the greater variability can be seen for this method from the figure. The `RF` and proposed methods seem to use the correlation among covariates for imputation and provide better imputation results than the `VIM` method. If we examine the `RF` method only, it yields better imputation results for 10% and 20% missing data, but suddenly the variability goes much higher for 30% missing data. In contrast, the $\texttt{wNNSel}_{\texttt{mix}}$ and $\texttt{wNNSel}_{\texttt{dum}}$ seem to perform well even for 30% missing data. Similarly, for the Hepatitis data, the `VIM` method provides poor results whereas other three imputation methods provide results that are smaller than that of `VIM` but closer to each other.

Overall when the relationship among covariates is stronger (e.g., cars, hepatitis), the proposed method makes an efficient use of this information to produce quality estimates for missing values. The random forests method also provide better results in this case but only when the missing values are smaller. Its performance deteriorates as the missing percentage increase. As for automobile data (table 6.3), `RF` yields smallest error for 10% missing, but for 30% missing data, the minimum is achieved by the $\texttt{wNNSel}_{\texttt{dum}}$ method.

In general, one would expect to obtain higher imputation errors with the increase in the percentage of missing data in a variable or when the number of incomplete variables increase or both. The reason is that the potentially useful information for the imputation of missing values is reduced. Indeed, it seems to confirmed by the Figure 6.6 which shows the average results over 200 repetitions obtained by different imputation methods. Each row shows a data set and within the row, the total error is plotted on the left, error for continuous variables is plotted in the middle and for nominal on the right. These results can be explained in different ways. First, the imputation quality depends on the percentage of missing values.

The quality of imputation deteriorates quickly with increasing missing values for BMI data, whereas it remains relatively stable for GBSG2 and automobile data. Overall, the method `VIM` always provides poor imputation compared other methods which seems to be similar to those observed in the simulations. The best imputation is obtained by the $\texttt{wNNSel}_{\texttt{dum}}$, regardless of the percentage of missing data. The difference in $\texttt{wNNSel}_{\texttt{dum}}$ and $\texttt{wNNSel}_{\texttt{mix}}$ methods is not so high except for hepatitis data, where it higher for 10% missing values but both methods get closer when the percentage of missing values increases. In contrast, for BMI data all methods, except `VIM`, show similar results.

We investigate how well each imputation method imputes the missing values in continuous and categorical variables. The detailed results for MSIEs and MAIEs of all the data sets are given in table 6.3 and 6.4 respectively. The smallest values are shown in boldface. The left panel of the table shows total error, the middle panel shows error for continuous variables only and the left panel for nominal variables. It can be seen from the table 6.3, that the nominal variables are best imputed by the $\texttt{wNNSel}_{\texttt{dum}}$ method for all the data

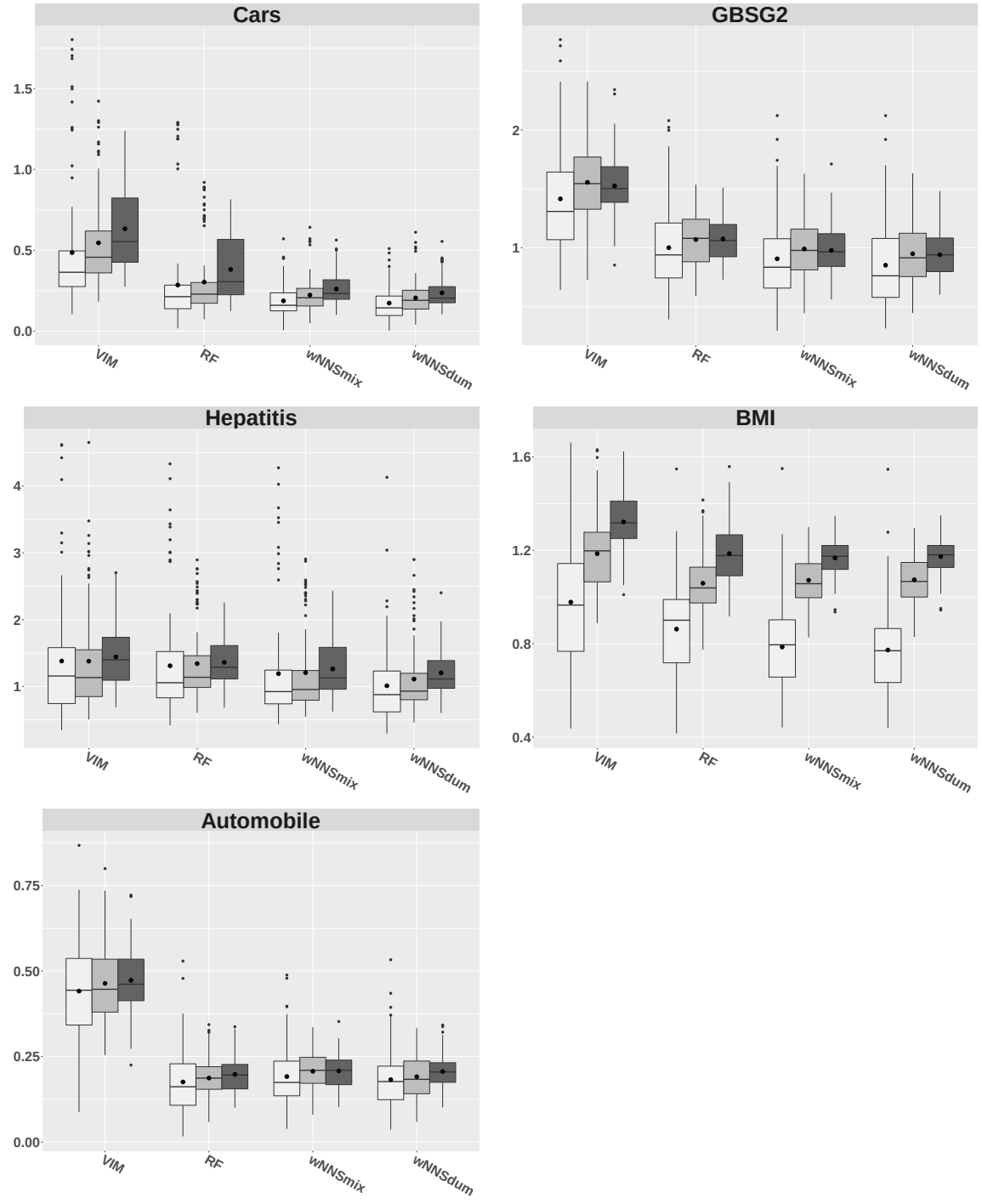


Figure 6.5.: Comparison of total MSIEs (mean squared imputation errors) using 10% (white), 20% (light gray) and 30% (dark gray) missing values, using k-nearest neighbors (`VIM`), random forest method (`RF`), and two proposed methods $\texttt{wNNSel}_{\texttt{mix}}$ and $\texttt{wNNSel}_{\texttt{dum}}$. Solid circles within boxes show mean values.

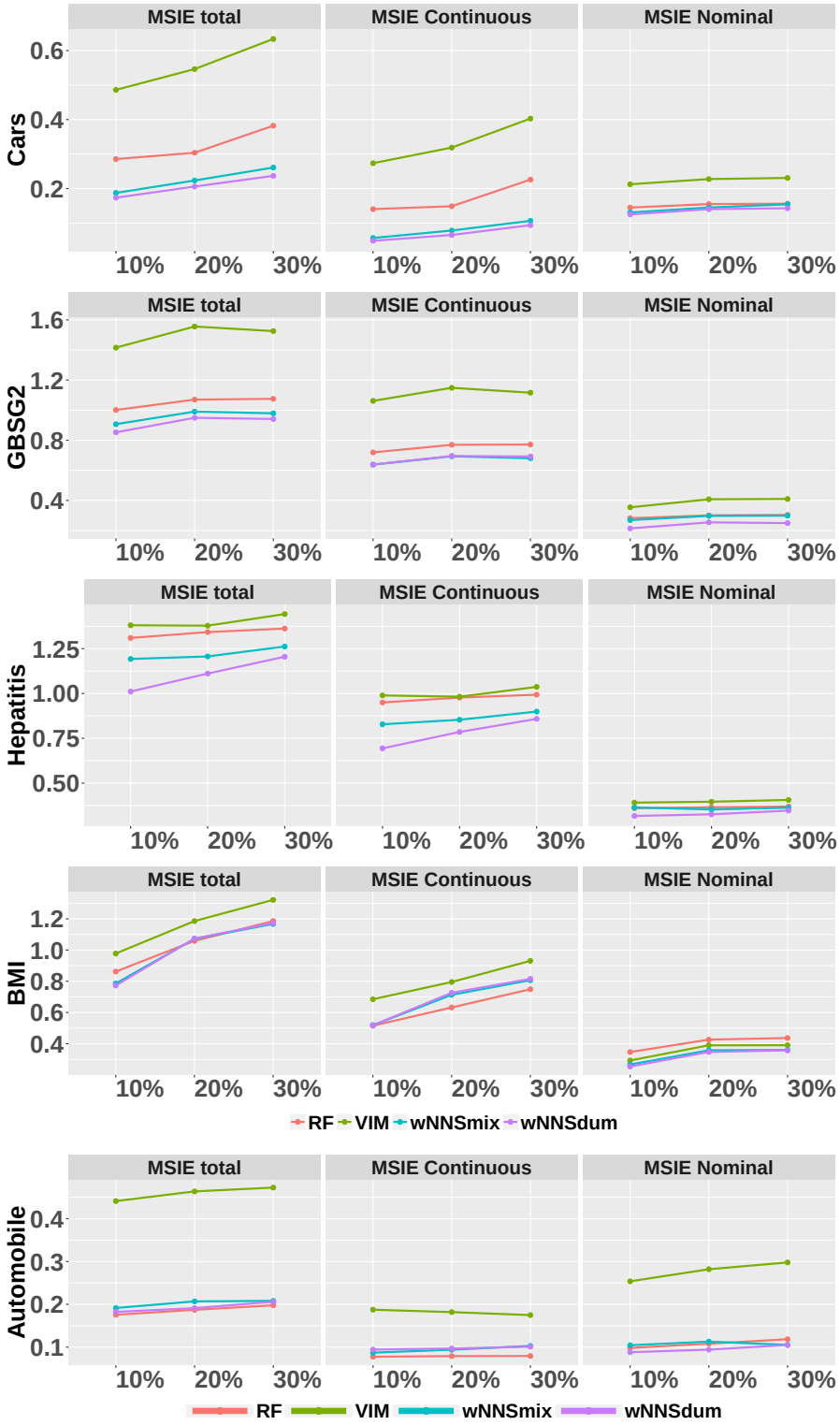


Figure 6.6.: Comparison of average imputation error obtained by different imputation methods using 10%, 20% and 30% missing data.. The left panel shows total error, the middle panel shows error for continuous variables only and the left panel for nominal variables.

Table 6.3.: Average MSIEs for different imputation methods.

Data	missing	Total MSIE				Continuous				Nominal			
		VIM	RF	wNNSel$_{mix}$	wNNSel$_{dum}$	VIM	RF	wNNSel$_{mix}$	wNNSel$_{dum}$	VIM	RF	wNNSel$_{mix}$	wNNSel$_{dum}$
Cars	10%	0.4860	0.2854	0.1874	**0.1735**	0.2735	0.1404	0.0567	**0.0485**	0.2125	0.1450	0.1306	**0.1250**
	20%	0.5462	0.3038	0.2234	**0.2058**	0.3187	0.1488	0.0784	**0.0655**	0.2275	0.1550	0.1450	**0.1403**
	30%	0.6335	0.3821	0.2609	**0.2368**	0.4027	0.2259	0.1065	**0.0938**	0.2308	0.1562	0.1544	**0.1430**
GBSG2	10%	1.4156	1.0012	0.9067	**0.8524**	1.0616	0.7192	**0.6377**	0.6384	0.3540	0.2820	0.2690	**0.2140**
	20%	1.5560	1.0700	0.9903	**0.9492**	1.1485	0.7695	**0.6933**	0.6952	0.4075	0.3005	0.2970	**0.2540**
	30%	1.5257	1.0753	0.9789	**0.9411**	1.1160	0.7713	**0.6803**	0.6921	0.4097	0.3040	0.2987	**0.2490**
Hepatitis	10%	1.3805	1.3098	1.1926	**1.0108**	0.9892	0.9498	0.8282	**0.6940**	0.3912	0.3600	0.3644	**0.3169**
	20%	1.3782	1.3425	1.2066	**1.1111**	0.9820	0.9772	0.8534	**0.7852**	0.3962	0.3653	0.3531	**0.3259**
	30%	1.4429	1.3622	1.2619	**1.2050**	1.0369	0.9932	0.8990	**0.8583**	0.4060	0.3690	0.3629	**0.3467**
BMI	10%	0.9780	0.8623	0.7859	**0.7728**	0.6852	**0.5154**	0.5189	0.5185	0.2928	0.3469	0.2670	**0.2543**
	20%	1.1857	**1.0588**	1.0717	1.0740	0.7955	**0.6323**	0.7135	0.7264	0.3902	0.4264	0.3581	**0.3475**
	30%	1.3215	1.1857	**1.1672**	1.1731	0.9311	**0.7491**	0.8069	0.8156	0.3904	0.4367	0.3603	**0.3575**
Automobile	10%	0.4412	**0.1756**	0.1915	0.1824	0.1874	**0.0775**	0.0871	0.0942	0.2537	0.0981	0.1044	**0.0881**
	20%	0.4637	**0.1872**	0.2069	0.1910	0.1818	**0.0791**	0.0940	0.0968	0.2819	0.1081	0.1129	**0.0942**
	30%	0.4727	**0.1978**	0.2080	0.2060	0.1749	**0.0793**	0.1028	0.1010	0.2978	0.1185	0.1052	**0.1050**

sets. For continuous variables, the wNNSel$_{dum}$ method provides smallest error in cars and hepatitis data and wNNSel$_{mix}$ method in GBSG2 data. For the other two data sets (BMI and automobile) the *RF* and the proposed methods provide nearly similar results (see Figure 6.6). For cars and Hepatitis data, the smallest errors for categorical as well as continuous variables by the wNNSel$_{dum}$ method. Except for automobile data, all the results are similar using MSIEs and MAIEs, the smallest error is obtained by *RF* using MSIEs as criterion and by the wNNSel$_{dum}$ method using MAIEs.

6.6. Concluding Remarks

Imputation of mixed type data is very challenging specially when the number of predictors is large. Only a few methods for the imputation of high-dimensional mixed data are available in literature. The imputation methods proposed in this paper are based on weighted nearest neighbors. We develop a distance formula for mixed data which takes into account the relationship among different data types as well as the association among variables. Thus the wNNSel$_{mix}$ imputation method uses the information on similarities among samples and association among covariates (continuous and categorical) simultaneously. The wNNSel$_{mix}$ method requires the values of tuning parameters, an algorithm for the selection of tuning parameters via cross-validation is also provided. Although time-consuming, cross-validation procedure choses the most plausible values for the particular data at hand.

Based on a simple idea, the wNNSel$_{dum}$ method outperformed in our simulations as well as real data evaluations surprisingly. Another advantage of the wNNSel$_{dum}$ is the computing speed since it requires less tuning parameters than that of wNNSel$_{mix}$ for imputation of missing values. The proposed method provides good imputation estimates for the missing

Table 6.4.: Average MAIEs for different imputation methods.

Data	missing	Total MSIE				Continuous				Nominal			
		VIM	RF	wNNSel$_{mix}$	wNNSel$_{dum}$	VIM	RF	wNNSel$_{mix}$	wNNSel$_{dum}$	VIM	RF	wNNSel$_{mix}$	wNNSel$_{dum}$
Cars	10%	0.5475	0.3328	0.2906	**0.2563**	0.3350	0.1878	0.1600	**0.1313**	0.2125	0.1450	0.1306	**0.1250**
	20%	0.5946	0.3556	0.3245	**0.2876**	0.3671	0.2006	0.1795	**0.1473**	0.2275	0.1550	0.1450	**0.1403**
	30%	0.6249	0.3931	0.3452	**0.3049**	0.3941	0.2369	0.1908	**0.1619**	0.2308	0.1562	0.1544	**0.1430**
GBSG2	10%	1.1411	0.9084	0.8541	**0.8026**	0.7871	0.6264	**0.5851**	0.5886	0.3540	0.2820	0.2690	**0.2140**
	20%	1.2229	0.9432	0.9050	**0.8649**	0.8154	0.6427	**0.6080**	0.6109	0.4075	0.3005	0.2970	**0.2540**
	30%	1.2142	0.9538	0.9043	**0.8626**	0.8045	0.6498	**0.6056**	0.6136	0.4097	0.3040	0.2987	**0.2490**
Hepatitis	10%	1.0593	1.0645	1.0220	**0.9152**	0.6681	0.7045	0.6577	**0.5983**	0.3912	0.3600	0.3644	**0.3169**
	20%	1.0475	1.0825	1.0228	**0.9694**	0.6513	0.7172	0.6697	**0.6435**	0.3962	0.3653	0.3531	**0.3259**
	30%	1.0862	1.0940	1.0534	**1.0228**	0.6802	0.7250	0.6905	**0.6761**	0.4060	0.3690	0.3629	**0.3467**
BMI	10%	0.8552	0.8842	0.8282	**0.8132**	0.5625	**0.5373**	0.5612	0.5589	0.2928	0.3469	0.2670	**0.2543**
	20%	**0.9960**	1.0116	1.0485	1.0448	0.6058	**0.5852**	0.6904	0.6972	0.3902	0.4264	0.3581	**0.3475**
	30%	**1.0570**	1.0662	1.1023	1.1036	0.6666	**0.6296**	0.7420	0.7462	0.3904	0.4367	0.3603	**0.3575**
Automobile	10%	0.4966	**0.2570**	0.2951	0.2572	0.2429	**0.1589**	0.1908	0.1690	0.2537	0.0981	0.1044	**0.0881**
	20%	0.5277	0.2742	0.3127	**0.2702**	0.2457	**0.1661**	0.1998	0.1760	0.2819	0.1081	0.1129	**0.0942**
	30%	0.5466	0.2924	0.3167	**0.2896**	0.2487	**0.1739**	0.2115	0.1846	0.2978	0.1185	0.1052	**0.1050**

values in the categorical as well as in the metric variables. The quality of imputation deteriorates as the missing percentage increases for all the methods, however, this deterioration mainly depends on the structure of the data specially the relationship among covariates.

Although the proposed approaches for the imputation of missing values in mixed data, improve traditional NN-based and the popular random forests imputation methods, they share the drawbacks of all single imputation methods. These include the fact that the single imputation do not account the uncertainty associated with imputation of missing values. If a statistical method is applied on the data, the variability of the estimators will be underestimated. A solution to this problem is to perform multiple imputation (Little and Rubin, 1987 , Little and Rubin, 2002); this will be a topic of future research. An advantage of the methods is that they also can be used in high-dimensional settings in which the number of covariates is higher than the number of samples. The proposed imputation algorithms could be a first step in a multiple imputation for mixed type data. When the relationship among covariates is stronger (e.g., cars, automobile), the proposed method makes an efficient use of this information to produce quality estimates for missing values.

7. Bootstrap Inference for Weighted Nearest Neighbors Imputation

Abstract

An attractive approach to fill in substitute values for missing data is imputation. A number of methods are available in literature that can be used for the imputation of the missing data. However it is not advisable to treat the imputed data just as the complete data. To apply the existing methods for analyzing the data, for example, to estimate the variance and/or statistical inference will probably produce invalid results because these methods do not account for the uncertainty of imputations. In this article we present analytic techniques for inference from a dataset in which missing values have been replaced by nearest neighbors imputation method. A simple and easy to use bootstrap algorithm that combines the nearest neighbors imputation with bootstrap resampling estimation to obtain valid bootstrap inference in a linear regression model is suggested. More specifically, imputing the bootstrap samples in the exact same way as original data was imputed produces correct bootstrap estimates. Simulation results show the performance of our approach in different data structures.

7.1. Introduction

Imputation is not the ultimate goal of any analysis. Missing values are generally not ignorable and hence are often imputed using some imputation technique. Many techniques have been proposed for the imputation of missing values, which provide valid estimates under certain conditions. It is not compulsory that an estimate with smaller imputation error also performs better in downstream analysis like regression analysis etc. Therefore, it is customary to compare the performance of imputed data in further analysis.

Under some basic assumptions, multiple imputation provides valid inferences and therefore remains the standard approach (Rubin, 1987). In the case of large samples, this approach gives nice properties under some assumptions including normality. However, when these assumption are not met, the confidence intervals and inferences based on them should be use carefully (Nielsen, 2003). In single imputation we obtain one estimate for each missing value, and when we take the asymptotics of this estimate like in GLM, it is misleading

because we are treating the imputed values as if they have been complete. They are not reflecting the uncertainty in the variance estimates of the regression coefficients.

A well known method for estimating the missing values in a data matrix is the nearest neighbors imputation (Hastie et al., 1999). Information from the candidate samples is used to estimate the missing entry (Troyanskaya et al., 2001). This method has been shown to be useful particularly for correlated data structures, see for example, Nguyen et al. (2004). Nearest neighbor methods may lead to invalid inferences after imputation despite providing smaller imputation errors. Therefore it becomes crucial to investigate about their performance of estimated parameters or models based on imputed data sets.

Bootstrap is a popular and computer intensive method which is often used for estimating the sampling distribution of estimators. Efron (1979) introduced this concept for independently identically distributed sample, and Rao and Wu (1988) provided the extensions for complex designs. They are valid for complete data only. When the data has been imputed to fill the missing values, the usual bootstrap methods should be used carefully as they may provide invalid results (Shao and Sitter, 1996). More specifically, if the estimated missing values are also treated like the true values, then the variance of the imputation estimate is underestimated because the expansion in variance due to imputation has been ignored. Alternatively, to get asymptotically valid variances and estimators of the population parameters the bootstrap sample should be imputed in the similar way as the original data (Shao and Sitter, 1996). A fractional bootstrap resampling method was proposed by Bello (1994) to obtain a *representative* sample relative to the proportion of missing values in the original data.

The theoretical studies by Rancourt et al. (1994) and Rancourt (1999) consist of the properties of NN estimators under a linear regression model. Under a general nonparametric setting, Chen and Shao (2000) developed the conditions under which the NN estimators, such as sample mean and quantiles, are consistent estimators and their bias is small relative to the standard error. However, these methods were developed in parametric models, since NN is nonparametric, a nonparametric variance estimation method is preferred as well. The jackknife variance estimators are described in Chen and Shao (2001), interval estimation in Shao and Wang (2008), and a consistent nonparametric variance estimator in Shao (2009).

Recently, Tutz and Ramzan (2015) proposed a weighted nearest neighbor method for the imputation of missing values. It utilizes a weighting scheme for selecting only those variables that actually contribute to the computation of the distances. Although this imputation method has been shown to provide smaller imputation errors when compared to other methods, inference based on imputation estimates has not yet been investigated. In this paper, We investigate the performance of the inferences based on the imputed data by weighted nearest neighbors including selection of predictors (`wNNSel`). In particular, we use bootstrap sampling to estimate the parameters and their confidence intervals. The coverage probability of the confidence intervals is quantified for several sample sizes and confidence levels. The paper compares coverage probabilities of the confidence intervals of regression coefficients to see if the true parameters are covered or not.

Section 7.2 briefly describes the weighted nearest neighbor approach to impute missing data and Section 7.3 describes the methods to calculate confidence intervals for the population parameters in general, and specific for the regression coefficients. The proposed procedure of bootstrap sampling to reach parameter estimates is described in Section 7.4. The coverage probability of confidence intervals of these estimates is also presented here. In Section 7.5 we present a simulation design and results. Finally, Section 7.6 presents some comments and concluding remarks.

7.2. Nearest Neighbors Imputation of Missing Values

In this section we briefly describe the nearest neighbor method to impute missing values.

Let $\mathbf{X} = (x_{is})$ be an $n \times p$ data matrix consisting of n observations on p covariates. Let $\mathbf{O} = (o_{is})$ denote the corresponding $n \times p$ matrix of dummies with entries

$$o_{is} = \begin{cases} 1 & \text{if } x_{is}\text{is not mising} \\ 0 & \text{for missing value.} \end{cases}$$

The observation vector $\mathbf{x}_i$ is the i-th row in the data matrix. To compute the distance between two observations $\mathbf{x}_i$ and $\mathbf{x}_j$, we use the L_q-metric defined by the following

$$d_q(\mathbf{x}_i, \mathbf{x}_j) = \left(\frac{1}{m_{ij}} \sum_{s=1}^{p} |x_{is} - x_{js}|^q I(o_{is} = 1) I(o_{js} = 1) \right)^{1/q},$$

An improved version of the above formula proposed by Tutz and Ramzan (2015) is defined by

$$d_{q,C}(\mathbf{x}_i, \mathbf{x}_j) = \left(\frac{1}{m_{ij}} \sum_{l=1}^{p} |x_{il} - x_{jl}|^q I(o_{il} = 1) I(o_{jl} = 1) C(r_{sl}) \right)^{1/q}, \tag{7.1}$$

where r_{sl} is the empirical correlation between covariates s, l and $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the correlations into weights. This equation uses selected covariates to compute the distance between observations. The covariates, who have a higher correlation with s^{th} covariate, will get a higher weight and thus contribute more to the calculation of distances.

A linear function in absolute correlation is $C(r_{sl}) = |r_{sl}|/(1-c) - c/(1-c)$ if $|r_{sl}| > c$, $C(r_{sl}) = 0$ if $|r_{sl}| \leq c$. Thus if $|r_{sl}| \leq c$ the covariate s has no contribution to the distance. Another smoother function is $C(r_{sl}) = |r_{sl}|^m$. The power m in the function $C(r_{sl}) = |r_{sl}|^m$ and c in linear function are tuning parameters and chosen by cross validation.

Based on the distances computed by equation (7.1), the observations are arranged in ascending order is $\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(k)}$ with $d(\mathbf{x}_i, \mathbf{x}_{(1)}) \leq \cdots \leq d(\mathbf{x}_i, \mathbf{x}_{(k)})$, where $\mathbf{x}_{(j)}^T = (x_{(j)1}, \ldots, x_{(j)p})$ denotes the jth nearest neighboring observation.

The imputed value of x_{is} is defined by

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\mathbf{x}_i, \mathbf{x}_{(j)}) x_{(j)s} \tag{7.2}$$

with weights

$$w(\mathbf{x}_i, \mathbf{x}_{(j)}) = \frac{K(\frac{d(\mathbf{x}_i, \mathbf{x}_{(j)})}{\lambda})}{\sum_{h=1}^{k} K(\frac{d(\mathbf{x}_i, \mathbf{x}_{(h)})}{\lambda})}. \tag{7.3}$$

where $K(.)$ is a kernel function and λ is a tuning parameter. The results of simulation studies of Tutz and Ramzan (2015) showed that Gaussian kernel with $q = 2$ provides a smaller imputation error under different simulation settings. Moreover the power function performs slightly better than the linear function. We take advantage of these findings and use Gaussian kernel, L_2-metric ($q = 2$) and power function for imputation of missing data in the next sections. The corresponding method is denoted by `wNNSel`.

7.3. Bootstrap Sampling and Missing Data

In practice, statistical inference, such as computing an approximate confidence interval for a population parameter, is needed. Bootstrap sampling is a general process of taking samples of finite size with or without replacement from the data at hand. The concept of using bootstrap in missing data problems is not recent (see for example, Efron (1994) and Schomaker and Heumann (2016)).

Ordinary Confidence Interval

Suppose p variables on n samples are observed and $\boldsymbol{X}$ is the observed data matrix. We use subscripts i and s to index the subjects and variables respectively with $i = 1, \cdots, n$ and $s = 1, \cdots, p$. Let $\boldsymbol{X} = (x_{is})$ denote a $n \times p$ matrix with independently and identically distributed rows, $\boldsymbol{x_i}^T = (x_{i1}, \cdots, x_{ip})$.

Let $\boldsymbol{X}$ the $n \times p$ data matrix with n samples on p predictors. Consider a linear regression model

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

where $\boldsymbol{y}^T = (y_1, \cdots, y_n)$ is the vector of responses, $\boldsymbol{\beta}^T = (\beta_0, \cdots, \beta_p)$ is the $(p+1)$-dimensional parameter vector, and $\boldsymbol{\epsilon}^T = (\epsilon_1, \cdots, \epsilon_n)$ is the vector of errors. The coefficients estimates are obtained by

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T\boldsymbol{X})^{-1} \cdot \boldsymbol{X}^T\boldsymbol{y}$$

The $(1-\alpha)\%$ confidence interval for the population parameter β is defined as

$$\hat{\beta} \pm t_{\alpha/2} S.E(\hat{\beta}),$$

where $\hat{\beta}$ is the point estimate, $S.E(\hat{\beta})$ is its standard error and $t_{\alpha/2}$ is the $100(\alpha/2)\%$ percentile of the t-distribution. If the distribution is normal, then the percentiles from the normal distribution, instead of the t-distribution, are used in the computation of the confidence interval.

7.4. Proposed Bootstrap Imputation Procedure

In this section we describe the procedure of drawing bootstrap samples from the data matrix to get estimates.

let $\boldsymbol{X}$ the $n \times p$ data matrix with n samples on p predictors and $\boldsymbol{y}$ an $n \times 1$ vector of responses. Let $\mathcal{D}$ be the $n \times (p+1)$ data set defined by $\mathcal{D} = (\boldsymbol{y}, \boldsymbol{X})$. Let θ be the parameter for which the confidence interval is to be constructed. Practically, θ can be sample mean $(\bar{X})$ or variance, regression coefficient β, correlation coefficient (r) or coefficient of determination (R^2). In this study, we focus only on the linear regression model and construct the confidence interval for the regression coefficients. We want to estimate $\theta = (\theta_1, \cdots, \theta_t)^T, t \geq 1$, where θ is a regression function of $\boldsymbol{y}$ on $\boldsymbol{X}$.

Furthermore, if the matrix $\boldsymbol{X}$ contains some missing values we denote it by $\boldsymbol{X}^{\text{mis}}$ and the corresponding data set as $\mathcal{D}^{\text{mis}}$. The procedure to get bootstrapped point estimates is defined as follows:

1. Impute the incomplete data $\mathcal{D}^{mis}$ using `wNNSel` method and obtain the optimal values of the tuning parameters (λ, m) values.
2. Draw B bootstrap samples of n rows with replacement from the missing data $\mathcal{D}^{mis}$ to get bootstrapped datasets $\mathcal{D}_b^{\text{mis},*}, b = 1, \cdots, B$.
3. Impute each $\mathcal{D}_b^{\text{mis},*}$ with `wNNSel` method using optimal tuning parameters (λ, m) obtained in step 1 to get $\mathcal{D}_b^{\text{imp},*}, b = 1, \cdots, B$.
4. Based on $\mathcal{D}_b^{\text{imp},*}, b = 1, \cdots, B$, Obtain B point estimates $\hat{\theta}_b^{\text{imp},*}$.

It is obvious that the bootstrap samples $\mathcal{D}_b^{\text{mis},*}$ may not contain exactly the same number of missing values as $\mathcal{D}^{mis}$. Using the imputed data $\mathcal{D}^{\text{imp}}$ (from step 1), we calculate $\hat{\theta}^{\text{imp}}$ the estimate of the parameter θ. (only for normal interval)

7.4.1. Bootstrap confidence interval

The bootstrap confidence intervals can be constructed in several ways. One approach is the normal theory interval which assumes that the statistic $\hat{\theta}$ is normally distributed, or, in sufficiently large samples, approximately normally distributed:

$$\hat{\theta}^{\text{imp}} \pm z_{\alpha/2}\hat{SE}(\hat{\theta}^{\text{imp}})$$

where $\hat{SE}(\hat{\theta}^{\text{imp}}) = \sqrt{\hat{\text{var}}(\hat{\theta}^{\text{imp}})}$ is the bootstrap estimate of the standard error of $\hat{\theta}$ based on $[\frac{1}{B}\sum_{b=1}^{B}(\hat{\theta}_b^{\text{imp},*} - \hat{\theta}_b^{\text{imp},*})^2]^{1/2}$ and $z_{\alpha/2}$ is the $\alpha/2$ quantile of the standard normal distribution.

7.4.2. Bootstrap Percentile Interval

Since a large number of samples is drawn in bootstrap sampling procedure, a simple natural choice to find confidence intervals is to make use of the quantiles. It has been shown that the intervals based on quantiles have correct asymptotic coverage (Efron, 1987). We take an advantage of this approach, and use the empirical quantiles of the $\hat{\theta}_b^*$ to form a confidence interval for θ.

Consider an ordered set of point estimates $\Theta_B^{\text{imp},*} = \{\hat{\theta}_{(b)}^{\text{imp},*}; 1, \cdots, B\}$, where $\hat{\theta}_{(1)}^{\text{imp},*} < \hat{\theta}_{(2)}^{\text{imp},*} < \cdots < \hat{\theta}_{(B)}^{\text{imp},*}$. Thus one obtains α level percentile confidence interval is given by

$$\hat{\theta}_{lower} < \theta < \hat{\theta}_{upper} = \hat{\theta}_{\alpha/2}^{\text{imp},*} < \theta < \hat{\theta}_{1-\alpha/2}^{\text{imp},*},$$

where $\hat{\theta}_{\alpha/2}^{\text{imp},*}$ is the $\alpha/2$-percent quantile of the ordered bootstrap set of estimates $\Theta_B^{\text{imp},*}$.

In contrast to the standard asymptotic confidence intervals, the bootstrap percentile intervals are generally asymmetric around the underlying parameter estimates. This is an attractive feature for situations when the true sampling distribution may not be symmetric.

7.4.3. Coverage Probability

The coverage probability (π) for a bootstrap confidence interval is the proportion of times the confidence interval $[\hat{\theta}_{lower}, \hat{\theta}_{upper}]$ contains the true value of the parameter θ.

$$\pi = \frac{1}{B}\sum_{b=1}^{B} I_{([\hat{\theta}_{lower}^*, \hat{\theta}_{upper}^*] \ni \theta)}$$

where $I(a)$ is an indicator function that takes the value 1 if the confidence interval covers the parameter.

$$I(\theta) = \begin{cases} 1 & \text{if } [\hat{\theta}^*_{lower}, \hat{\theta}^*_{upper}] \ni \theta \\ 0 & \text{Oherwise.} \end{cases}$$

The complete procedure for calculating the coverage probability of confidence interval obtained by bootstrapping is described in algorithm 7.1.

Algorithm 7.1 Coverage Probability of Bootstrap Confidence intervals

Require: $\mathbf{X}$ an $n \times p$ matrix, validation sets t, $\mathbf{y}_{t,n\times 1}$ $t = 1, \cdots, \mathcal{T}$ to form $\mathcal{D}_t^{mis} = (\boldsymbol{y}, \boldsymbol{X})$,

1: $\mathbf{X}^{\text{imp}} \leftarrow$ initial imputation using `wNNSel` to find (λ_{opt}, m_{opt})

2: **for** $t = 1, \ldots, \mathcal{T}$ **do**

3: $\mathcal{D}_{t,b}^{\text{mis},*}$: Draw b bootstrap samples from $\mathcal{D}_t^{mis}$

4: **for** $b \in B$ **do**

5: $\mathcal{D}_{t,b}^{\text{imp},*}$: imputation of $\mathcal{D}_{t,b}^{\text{mis},*}$ using (λ_{opt}, m_{opt}) via `wNNSel` method

6: $\hat{\beta}^*_{t,b}$: bootstrap estimates of the regression coefficients from $\mathcal{D}_{t,b}^{\text{imp},*}$

7: $[\hat{\beta}^{\text{imp},*}_{\alpha/2}, \hat{\beta}^{\text{imp},*}_{1-\alpha/2}]_{t,b}$: bootstrap confidence interval using α-percent quantiles of the ordered bootstrap estimates.

8: $\pi^*_t = \frac{1}{B} \sum_{b=1}^{B} I_{([\hat{\beta}^{\text{imp},*}_{\alpha/2}, \hat{\beta}^{\text{imp},*}_{1-\alpha/2}]_{t,b} \ni \beta)}$

9: **end for**

10: $\pi^* = \frac{1}{t} \sum_{t=1}^{\mathcal{T}} \pi^*_t$

11: **end for**

7.5. Simulation Studies

To investigate the performance of the proposed method we conduct simulation study under different settings.

7.5.1. Simulation Study 1

In our first study, we simulate a data matrix $\boldsymbol{X}_{n\times p}$ from a multivariate normal distribution with mean $\boldsymbol{\mu} = \mathbf{0}$ and covariance matrix $\boldsymbol{\Sigma}$ of autoregressive type with $\rho = 0.9$. We use

linear regression model $\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$, to generate the outcome or response. The parameter vector $\boldsymbol{\beta}_{(p+1)\times 1}$ is generated using $\boldsymbol{\beta}_{(p+1)\times 1} = [\beta_0 = 0.5, \boldsymbol{\beta}_{p\times 1} \sim unif(-1, 1)\ ]$.

Table 7.1.: Estimated coverage rates, standard errors, and widths of confidence intervals for $\boldsymbol{\beta}$

n	α	**method**	**Average**	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
			Coverage probabilities (%) of confidence intervals for $\hat{\boldsymbol{\beta}}$											
50	5%	`boot`	96	94	97	95	97	97	97	97	97	97	97	98
		`noboot`	95	93	95	94	96	95	94	97	96	97	92	94
	10%	`boot`	92	87	92	90	91	93	93	93	95	96	89	92
		`noboot`	90	85	88	90	90	92	90	94	91	93	86	89
100	5%	`boot`	95	97	95	96	95	94	94	99	94	93	94	95
		`noboot`	95	97	96	95	96	96	96	98	94	92	94	95
	10%	`boot`	90	92	91	92	91	90	89	92	89	88	89	89
		`noboot`	90	92	92	91	90	91	89	95	89	88	90	90
			SE($\hat{\boldsymbol{\beta}}$)											
50	5%	`boot`	0.47	0.16	0.42	0.57	0.50	0.47	0.55	0.76	0.50	0.45	0.45	0.34
		`noboot`	0.46	0.16	0.42	0.57	0.50	0.47	0.56	0.79	0.50	0.44	0.39	0.31
	10%	`boot`	0.47	0.16	0.42	0.57	0.50	0.47	0.55	0.76	0.50	0.45	0.45	0.34
		`noboot`	0.46	0.16	0.42	0.57	0.50	0.47	0.56	0.79	0.50	0.44	0.39	0.31
100	5%	`boot`	0.31	0.11	0.25	0.35	0.37	0.35	0.33	0.38	0.38	0.34	0.35	0.25
		`noboot`	0.31	0.11	0.25	0.36	0.37	0.35	0.33	0.38	0.38	0.34	0.35	0.25
	10%	`boot`	0.31	0.11	0.25	0.35	0.37	0.35	0.33	0.38	0.38	0.34	0.35	0.25
		`noboot`	0.31	0.11	0.25	0.36	0.37	0.35	0.33	0.38	0.38	0.34	0.35	0.25
			Median of widths of confidence interval for $\boldsymbol{\beta}$											
50	5%	`boot`	2.06	0.67	1.85	2.43	2.19	2.03	2.38	3.29	2.21	2.00	2.11	1.52
		`noboot`	1.87	0.63	1.70	2.28	2.00	1.88	2.27	3.17	2.01	1.76	1.59	1.24
	10%	`boot`	1.69	0.56	1.52	2.00	1.79	1.68	1.97	2.73	1.82	1.61	1.73	1.23
		`noboot`	1.55	0.52	1.42	1.90	1.67	1.56	1.89	2.64	1.67	1.47	1.33	1.03
100	5%	`boot`	1.25	0.45	1.01	1.42	1.45	1.36	1.28	1.50	1.47	1.42	1.37	1.00
		`noboot`	1.25	0.44	1.00	1.41	1.45	1.38	1.29	1.50	1.50	1.36	1.40	1.00
	10%	`boot`	1.04	0.38	0.84	1.19	1.22	1.13	1.08	1.25	1.23	1.18	1.15	0.83
		`noboot`	1.04	0.37	0.83	1.18	1.21	1.15	1.08	1.26	1.25	1.14	1.17	0.83

We select samples of size $n = 50,\ 100$ with $p = 10$ predictors. In the data matrix $\boldsymbol{X}$, we delete 10% of the values as missing completely at random. For each simulated sample, the distribution of $\hat{\boldsymbol{\beta}}$ is estimated based on $B = 500$ bootstrap samples. The percentile method is then used to construct the confidence intervals of the regression coefficients. We use $\alpha = 0.05, 0.10$ in our simulations. We calculate the bootstrap percentile intervals using the bootstrap sample estimates. For comparison of the confidence intervals, we compute the coverage percentages and their median widths.

The empirical coverage probabilities of the estimated regression coefficients using the bootstrap method (shown as `boot`) and single imputation (that is without bootstrap method

, shown as `noboot`) are summarized in Table 7.1. The first column in table indicates the mean coverage of the $\boldsymbol{\beta}$, and the corresponding columns show the coverage probabilities for each individual regression coefficient. For $n = 50$ with α=5%, using `noboot` to estimate the standard errors in the linear regression model yields an average estimated coverage probability of 95%. If we look on the coverage probabilities of individual βs, it is seen that β_9 gets the smallest coverage (92%) and the highest coverage probability is achieved by β_6 and β_8 (97%). Bootstrapping yields average estimated coverage probability of 96%. It is seen from the individual probabilities, that most of the βs attain estimated coverage of more than 95% with an exception of β_0 having a coverage of 94%. When $\alpha = 10\%$, the average estimated coverage probability is 90% if using no bootstrapping and 92% with bootstrap. For a larger sample with $n = 100$, using `noboot` and with bootstrap both methods yield an average estimated coverage probability of 95% for $\alpha = 5\%$, and 90% for $\alpha = 10\%$.

The estimated standard errors of $\boldsymbol{\beta}$ and the median width of their confidence intervals are given in the middle and lower panel of Table 7.1 respectively. When $n = 100$, the standard errors of $\boldsymbol{\beta}$ using bootstrap procedure are almost identical to the mean estimated standard errors under no bootstrap estimation (see Table 7.1 middle panel). It is also clear that the increment in the sample size decreases the standard error of $\boldsymbol{\beta}$ (for $n = 50$ we get $\hat{SE}(\boldsymbol{\beta})$ = 1.46, whereas for $n = 100$ we get $\hat{SE}(\boldsymbol{\beta}) = 0.31$).

It is noticeable that for $n = 100$, the width of confidence interval are almost identical for both methods using bootstrap and no bootstrap (Table 7.1 lower panel). On the other hand, for $n = 50$, using bootstrap yields wider confidence interval with average width of 2.06 as compared to 1.87 when no bootstrap is used for α=5%, and an average width of 1.69 (`boot`) verses 1.55 (`noboot`) for α=10%. Overall, imputing the bootstrap data using the same method as the original data lead to better results with coverage probabilities very close to the nominal level.

7.5.2. Simulation Study 2

In the following, we conduct another simulation study to investigate the bootstrap performance of the method when the number of predictors increases, number of missing values increases, and the missing values occur under different mechanisms. More specifically, we set the number of predictors 15, 30, 45, the missing values come from MCAR and MAR mechanism, and the missing percentage varies between 10% to 30%. We use the following data settings:

- Setting 1: $n = 50$, $p = 15, 30$
- Setting 2: $n = 100$, $p = 15, 45$
- $\alpha = 5\%, 10\%$
- $miss = 10\%, 20\%, 30\%$ for both settings
- Mechanism : MCAR, MAR

The coverage probabilities of 95% confidence interval for $\hat{\boldsymbol{\beta}}$ using $n = 100$ and $p = 15$ are plotted in Figure 7.1 respectively. The horizontal red line indicates the nominal level. It is clear from the figure that `noboot` method provides lower coverage than `boot` method. This difference gets higher when the missing data is MAR. The detailed results for $n = 50, 100$ using 95% confidence level are given in Table 7.2. Similar results are obtained for 90% confidence level (see Table B.1 in Appendix B). The low coverage of `noboot` method is due to its narrow intervals than that of `boot` (see Tables B.2 and B.3 in Appendix B).

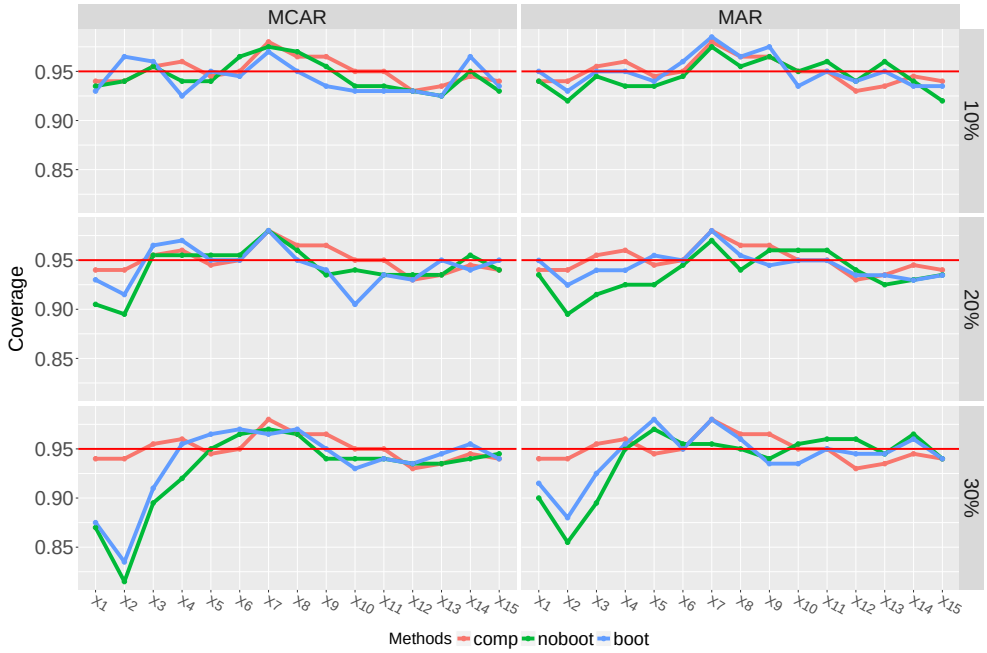


Figure 7.1.: Coverage for 95% CI of individual $\boldsymbol{\beta}$'s using $n = 100, p = 15$. The horizontal red lines show the nominal level of confidence.

To see the effect of increasing the number of predictors, we look at Figure 7.2 that shows the coverage probabilities for 10, 20 and 30% missing data under MCAR and MAR missing mechanisms. The upper panel shows coverage $n = 100$, $p = 15$ and lower panel shows for $p = 45$. It can be seen that the bootstrapping (shown as `boot`) provides stable coverage for all missing percentages, whereas the coverage rates of single imputation, i.e. without bootstrap (shown as `noboot`) are slightly lower than the nominal rates.

For $n = 100$ and $p = 45$, the coverage probabilities for `boot` method are greater than the nominal level. It is due the higher standard errors of $\hat{\boldsymbol{\beta}}$ which result in wider confidence intervals, and as a result the coverage rate rises than its nominal values. The standard errors of $\hat{\boldsymbol{\beta}}$ are higher for bootstrap as can be seen in Figure 7.3. The upper panel of Figure 7.3 shows the SE($\hat{\boldsymbol{\beta}}$) for $n = 50$ and lower panel for $n = 100$. When the sample size is $n = 50$, the SE($\hat{\boldsymbol{\beta}}$) for `boot` method is increases drastically for $p = 30$. Whereas for $n = 100$, an increase in the number of predictors from $p = 15$ to $p = 45$ does not inflate the standard errors. Overall the `boot` method always provides higher values of SE($\hat{\boldsymbol{\beta}}$).

Table 7.2.: Coverage rate for 95% confidence intervals for $\hat{\boldsymbol{\beta}}$

Mechanism	miss	method	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
			$n = 50$														
MCAR	10	comp	0.92	0.91	0.95	0.95	0.90	0.94	0.94	0.95	0.95	0.96	0.98	0.96	0.95	0.92	0.97
		noboot	0.93	0.93	0.94	0.95	0.92	0.94	0.94	0.95	0.95	0.96	0.96	0.96	0.94	0.93	0.95
		boot	0.95	0.95	0.97	0.97	0.94	0.96	0.96	0.97	0.97	0.98	0.98	0.96	0.96	0.96	0.97
	20	comp	0.92	0.91	0.95	0.95	0.90	0.94	0.94	0.95	0.95	0.96	0.98	0.96	0.95	0.92	0.97
		noboot	0.93	0.92	0.95	0.95	0.92	0.92	0.96	0.96	0.95	0.96	0.96	0.95	0.92	0.92	0.96
		boot	0.96	0.96	0.99	0.99	0.96	0.97	0.96	0.97	0.97	0.98	0.98	0.97	0.96	0.96	0.97
	30	comp	0.92	0.91	0.95	0.95	0.90	0.94	0.94	0.95	0.95	0.96	0.98	0.96	0.95	0.92	0.97
		noboot	0.92	0.88	0.90	0.94	0.94	0.95	0.93	0.96	0.96	0.97	0.96	0.96	0.93	0.91	0.97
		boot	0.97	0.95	0.97	0.98	0.98	0.97	0.97	0.98	0.97	0.98	0.98	0.97	0.95	0.96	0.98
MAR	10	comp	0.92	0.91	0.95	0.95	0.90	0.94	0.94	0.95	0.95	0.96	0.98	0.96	0.95	0.92	0.97
		noboot	0.92	0.90	0.95	0.94	0.92	0.96	0.94	0.96	0.95	0.96	0.96	0.95	0.93	0.92	0.95
		boot	0.96	0.96	0.97	0.97	0.97	0.97	0.96	0.98	0.97	0.98	0.98	0.98	0.95	0.96	0.98
	20	comp	0.92	0.91	0.95	0.95	0.90	0.94	0.94	0.95	0.95	0.96	0.98	0.96	0.95	0.92	0.97
		noboot	0.94	0.88	0.95	0.94	0.92	0.93	0.94	0.95	0.96	0.98	0.95	0.94	0.94	0.92	0.96
		boot	0.97	0.95	1.00	0.97	0.96	0.97	0.97	0.99	0.99	0.99	0.97	0.97	0.95	0.95	0.98
	30	comp	0.92	0.91	0.95	0.95	0.90	0.94	0.94	0.95	0.95	0.96	0.98	0.96	0.95	0.92	0.97
		noboot	0.90	0.88	0.90	0.97	0.94	0.94	0.94	0.95	0.95	0.97	0.94	0.92	0.94	0.92	0.95
		boot	0.98	0.95	0.97	0.99	0.99	0.98	0.96	0.98	0.97	0.98	0.97	0.98	0.97	0.95	0.97
			$n = 100$														
MCAR	10	comp	0.94	0.94	0.95	0.96	0.94	0.95	0.98	0.96	0.96	0.95	0.95	0.93	0.94	0.94	0.94
		noboot	0.94	0.94	0.95	0.94	0.94	0.96	0.97	0.97	0.95	0.94	0.94	0.93	0.92	0.95	0.93
		boot	0.93	0.96	0.96	0.92	0.95	0.94	0.97	0.95	0.94	0.93	0.93	0.93	0.92	0.96	0.94
	20	comp	0.94	0.94	0.95	0.96	0.94	0.95	0.98	0.96	0.96	0.95	0.95	0.93	0.94	0.94	0.94
		noboot	0.90	0.90	0.95	0.95	0.95	0.95	0.98	0.96	0.94	0.94	0.94	0.94	0.94	0.95	0.94
		boot	0.93	0.92	0.96	0.97	0.95	0.95	0.98	0.95	0.94	0.90	0.94	0.93	0.95	0.94	0.95
	30	comp	0.94	0.94	0.95	0.96	0.94	0.95	0.98	0.96	0.96	0.95	0.95	0.93	0.94	0.94	0.94
		noboot	0.87	0.81	0.90	0.92	0.95	0.96	0.97	0.96	0.94	0.94	0.94	0.94	0.94	0.94	0.94
		boot	0.88	0.83	0.91	0.95	0.96	0.97	0.96	0.97	0.95	0.93	0.94	0.94	0.94	0.95	0.94
MAR	10	comp	0.94	0.94	0.95	0.96	0.94	0.95	0.98	0.96	0.96	0.95	0.95	0.93	0.94	0.94	0.94
		noboot	0.94	0.92	0.94	0.94	0.94	0.94	0.97	0.95	0.96	0.95	0.96	0.94	0.96	0.94	0.92
		boot	0.95	0.93	0.95	0.95	0.94	0.96	0.98	0.96	0.97	0.94	0.95	0.94	0.95	0.94	0.94
	20	comp	0.94	0.94	0.95	0.96	0.94	0.95	0.98	0.96	0.96	0.95	0.95	0.93	0.94	0.94	0.94
		noboot	0.94	0.90	0.92	0.92	0.92	0.94	0.97	0.94	0.96	0.96	0.96	0.94	0.92	0.93	0.94
		boot	0.95	0.92	0.94	0.94	0.95	0.95	0.98	0.95	0.94	0.95	0.95	0.93	0.93	0.93	0.93
	30	comp	0.94	0.94	0.95	0.96	0.94	0.95	0.98	0.96	0.96	0.95	0.95	0.93	0.94	0.94	0.94
		noboot	0.90	0.85	0.90	0.95	0.97	0.95	0.95	0.95	0.94	0.95	0.96	0.96	0.94	0.96	0.94
		boot	0.92	0.88	0.92	0.95	0.98	0.95	0.98	0.96	0.94	0.94	0.95	0.94	0.94	0.96	0.94

Figure B.1 in appendix B shows 95% confidence interval plots of estimated regression coefficients for $n = 50, 100$ and $p = 15$ with 10% and 30% missing observations. The upper two panels are for MCAR and the lower two panels are for MAR mechanisms respectively. To construct the CIs, likelihood estimates were used. In each plot, the black horizontal line represents the true value of the regression coefficient, the gray lines are for the samples (out of 200) for which CI covered the true parameter value, and the red lines are for the samples for which CI did not cover the true parameter value.

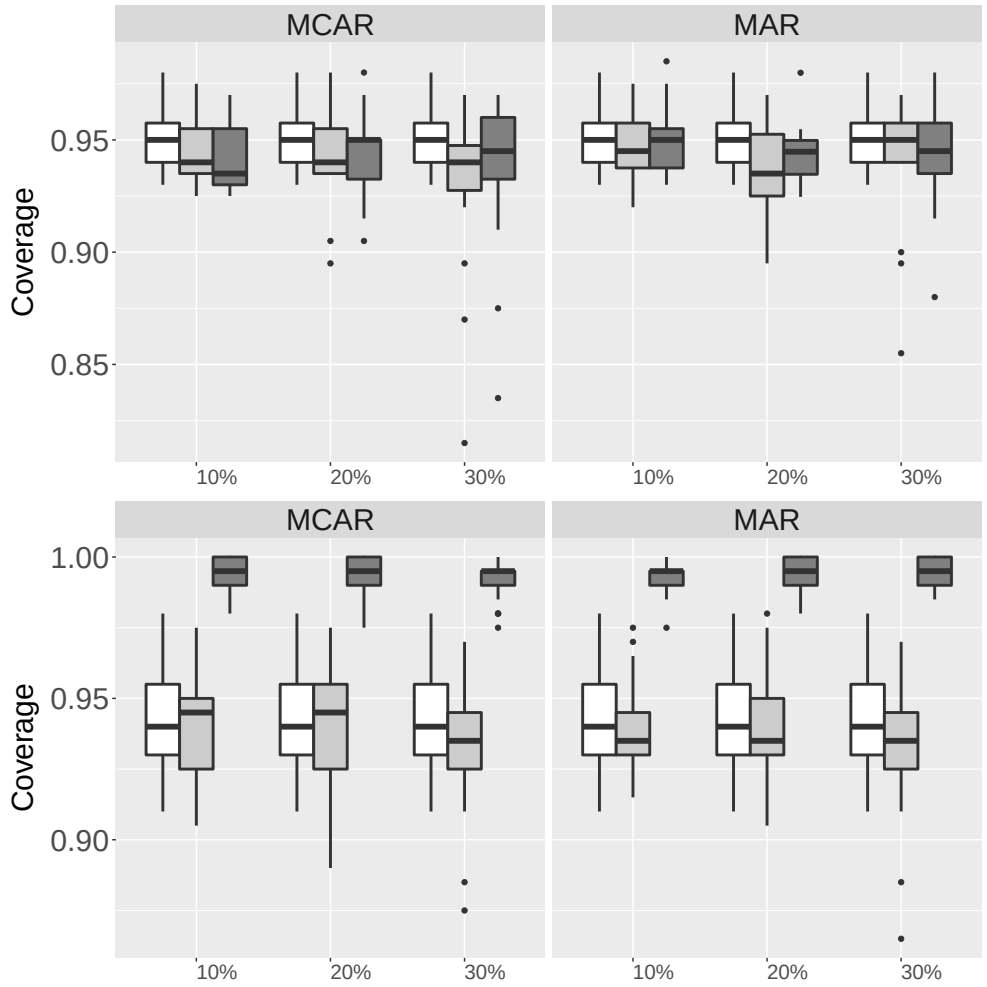


Figure 7.2.: Coverage of confidence intervals for 95% confidence level using $n = 100$, $p = 15$ (upper) and $p = 45$ (lower).

The average results for the coverage probabilities using 90% and 95% level of confidence and the standard errors of $\hat{\boldsymbol{\beta}}$ are given in Table 7.3. As expected, the overall standard errors for $n = 100$ are smaller than that of $n = 50$. The results show that `noboot` sometimes has too low coverage, for example, the coverage for $n = 50, p = 30$, 30% MAR missing data, is 0.86 (90% confidence level) and 0.92 (95% confidence level). Whereas the bootstrap method attains too high, for example, for the larger number of predictors $p = 45$ the coverage is near 1. The reason is the length of the intervals and the standard errors of the estimated coefficients. The intervals for the bootstrap method are longer than `noboot`.

7.6. Concluding Remarks

Treating the imputed data just like complete may produce invalid inferences. We proposed an algorithm that combines the bootstrap with the weighted nearest neighbors imputation

Table 7.3.: Average coverage for 90% and 95% confidence intervals and standard errors of the estimated coefficients $\hat{\boldsymbol{\beta}}$

n	p	Mechanism	miss	90%			95%			SE($\hat{\boldsymbol{\beta}}$)		
				comp	noboot	boot	comp	noboot	boot	comp	noboot	boot
50	15	MCAR	10%	0.89	0.89	0.93	0.95	0.94	0.97	0.52	0.53	0.54
			20%	0.89	0.88	0.93	0.95	0.94	0.97	0.52	0.54	0.55
			30%	0.89	0.88	0.94	0.95	0.94	0.97	0.52	0.55	0.57
		MAR	10%	0.89	0.88	0.94	0.95	0.94	0.97	0.52	0.53	0.54
			20%	0.89	0.88	0.94	0.95	0.94	0.97	0.52	0.53	0.55
			30%	0.89	0.88	0.94	0.95	0.94	0.98	0.52	0.54	0.55
	30	MCAR	10%	0.88	0.87	1.00	0.93	0.93	1.00	0.70	0.72	0.91
			20%	0.88	0.88	0.99	0.93	0.93	1.00	0.70	0.75	0.97
			30%	0.88	0.87	1.00	0.93	0.92	1.00	0.70	0.78	1.00
		MAR	10%	0.88	0.88	1.00	0.93	0.93	1.00	0.70	0.71	0.88
			20%	0.88	0.88	0.99	0.93	0.93	0.99	0.70	0.72	0.89
			30%	0.88	0.86	1.00	0.93	0.92	1.00	0.70	0.73	0.95
100	15	MCAR	10%	0.90	0.89	0.90	0.95	0.95	0.94	0.33	0.34	0.34
			20%	0.90	0.89	0.90	0.95	0.94	0.94	0.33	0.34	0.35
			30%	0.90	0.87	0.88	0.95	0.93	0.94	0.33	0.35	0.35
		MAR	10%	0.90	0.89	0.90	0.95	0.95	0.95	0.33	0.34	0.34
			20%	0.90	0.89	0.90	0.95	0.94	0.94	0.33	0.34	0.35
			30%	0.90	0.88	0.89	0.95	0.94	0.94	0.33	0.34	0.35
	45	MCAR	10%	0.89	0.89	0.98	0.94	0.94	0.99	0.42	0.43	0.44
			20%	0.89	0.88	0.98	0.94	0.94	0.99	0.42	0.44	0.45
			30%	0.89	0.88	0.98	0.94	0.94	0.99	0.42	0.44	0.45
		MAR	10%	0.89	0.89	0.98	0.94	0.94	0.99	0.42	0.43	0.44
			20%	0.89	0.88	0.98	0.94	0.94	0.99	0.42	0.44	0.45
			30%	0.89	0.88	0.98	0.94	0.94	0.99	0.42	0.44	0.45

to reach valid statistical inference. Simulation results show that imputing the bootstrap samples in the exact same way as original data was imputed produces correct bootstrap estimates. The estimated values for the missings which have not only smaller standard errors, but also provide better inference, when one variable is a response variable and the rest of the variables are predictors in a linear regression model. The single imputation provides lower coverage than the nominal level as expected. The suggested bootstrap resampling estimation to obtain valid bootstrap inference in a linear regression model provides promising results with the desired nominal coverage.

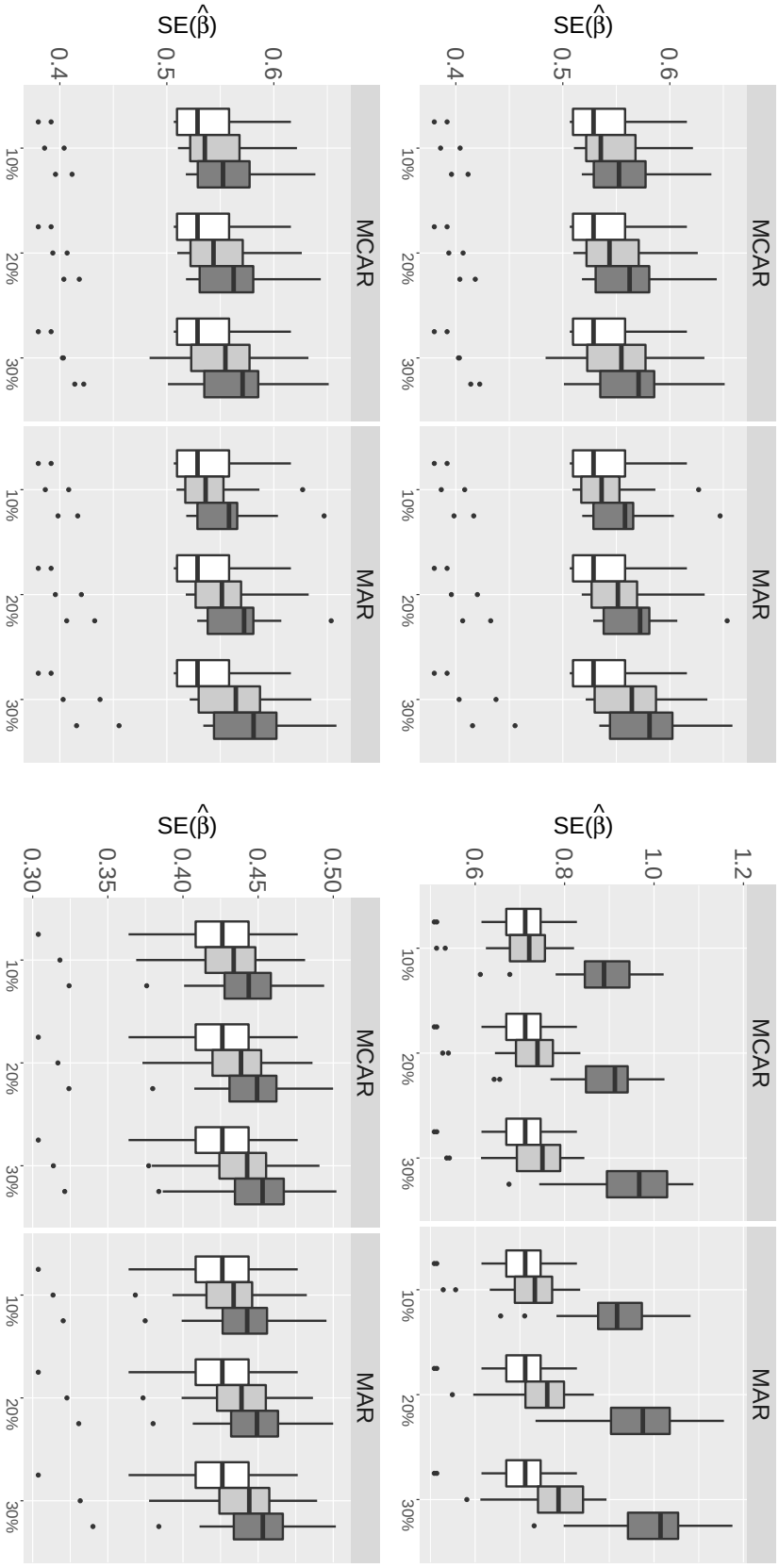


Figure 7.3.: Standard errors of $\hat{\boldsymbol{\beta}}$ for $n = 50$ $p = 15$ (upper left), $n = 50$ $p = 45$ (upper right), $n = 100$ $p = 15$ (lower left), and $n = 100$ $p = 45$ (lower right).

8. Missing Values in Classification: Improved Imputation Methods for High-Dimensional Settings

Abstract

Missing values are an ubiquitous problem in classification. Although some methods in machine learning are tolerant to the presence of missing values, many statistical methods require a complete data matrix and so do classification methods. A number of methods have been suggested to impute missing values and often they provide better results in terms of imputation errors. We investigate the performance of imputation methods, in particular improved nearest neighbors method, in classification. Some classification methods are robust to the presence of missing values and hence their accuracy is not affected much. Other methods are influenced by imputed values. We compare the performance of different imputation methods on the classification accuracy of the dataset. In particular, we consider missing data ranging from 10% to 60% in the training data, impute them by using different imputation techniques and compute the performance of different classifiers on the test data. Our experiments show that the weighted nearest neighbors and the random forest imputation methods perform better in terms of classification accuracy in various datasets considered in our evaluations.

8.1. Introduction

In real world applications, data sets often come with missing values. Even after careful experimentation, it is sometimes not possible to get a complete data without the curse of missing values. Statistical modeling and inferences may suffer greatly due to presence of such missing values. The inappropriate handling of missing values in the analysis may introduce bias and can not only provide misleading inferences but can also limit the generalizability of the research findings (Wang and Wang, 2010). A number of approaches have been suggested in the literature to deal with them. A simple and easy approach is to ignore those samples and/or covariates that are not complete. However, this method is practical only when the data contain a very small proportion of missing values and when the analysis of the complete observations will not lead to serious bias during the inference (Rubin,

1987). On the other hand, if a significant number of cases contain missing values, it may be beneficial to impute the missing data before proceeding. Some machine learning tools are tolerant to missing values, however, a notable number of statistical analyses require complete data. Thus the imputation approaches aimed at fill in missing data are worthy of attention.

In the particular case of classification, incomplete data in either the training set or test set or in both sets affect the prediction accuracy of learned classifiers. A naive imputation method can affect the performance of a classifier constructed by using that data set as a training sample. An imputed value is always a guess at the actual value. The imputation variation is natural due to the uncertainty about the actual value. Imputation can lead to an underestimation or overestimation of the uncertainty of the estimate. If the uncertainty is underestimated, the classifier trained with the imputed data set will over-fit the training data and produce invalid outputs. On the other hand, if the uncertainty is overestimated, the classifier trained with the imputed data set will under-fit the training data and provide poor prediction (Gheyas and Smith, 2010).

The literature relating to the classification with missing data usually focuses on analyzing and comparing a particular imputation method with others for a specific proportion of missing values under a particular missing mechanism (MCAR: Missing Completely at Random, MAR: Missing at Random, or MNAR: Missing Not at Random). Grzymala-Busse and Hu (2001) compared different imputation approaches to examine their impact on classification. Their results which are based on a limited study with real-life data sets containing missing values, showed that the imputation discussed by Grzymala-Busse (1991) can be a promising candidate for the best-performance in the classification stage. Batista and Monard (2003) compared k-nearest neighbors imputation with mean imputation, and internal handling of missing values in CN2 (Clark and Niblett, 1989a) and C4.5 (Quinlan, 1993) algorithms, for supervised learning. Their comparison showed that the results with 10-nearest neighbors are very good, even for training sets having a large amount of missing data. Further, their findings lead to the conclusion that the imputation of a missing attribute that is highly correlated with some others attributes in the data is useless, or even harmful. Acuña and Rodriguez (2004) evaluated the effect of different imputation methods on the misclassification error rate. They considered deletion of missing cases, mean and median imputation and kNN imputation for two classifiers i.e., Linear Discriminant Analysis (LDA) that is a parametric classifier and the nonparametric kNN classifier. Both classifiers performed similarly when there was a small proportion of missing values, regardless of the selection of any missing data treatment approach.

Hruschka et al. (2007) examined the prediction and classification performance of their proposed imputation methods using a simulation study with four real datasets. Although, their approach performed better than the classical imputation methods in two datasets, the better imputation results did not implied in better classification results. Farhangfar et al. (2008) compared different classifiers for classification with missing values in discrete data, but could not find any universally best imputation method that can improve the misclassification error in all situations. However, according to their findings the C4.5 and

Naive-Bayes classifiers were resistant to missing data, i.e., they can also produce accurate classification in the presence of missing data, while the other classifiers essentially required the complete data. Luengo et al. (2012) used a variety of imputation techniques, and different classification methods by splitting them into three categories. Their findings proved the importance of imputation over case deletion. Furthermore, none of the imputation method could universally outperform the others regarding the accuracy of classification (see also, García-Laencina et al., 2010). Gheyas and Smith (2010) mentioned that the single imputation methods are able to show better prediction capabilities than multiple imputation ones for a wide range of data sets due to the lower overfitting responses of the former ones.

For classification or prediction with missing data, instead of imputing the missing data, an alternative is to construct models that employ only those features that are known for a particular test case making imputation unnecessary. This reduced-model approach uses only a subset of the features that are available for the training data. This approach potentially employs a different model for each test instance. Such reduced-model approach was treated to some extent by Schuurmans and Greiner (1997), and presented as *lazy* classification tree induction by Friedman et al. (1996). This approach provides accurate results, but expensive in terms of computation and storage (Saar-Tsechansky and Provost, 2007).

In the classification context, features may be missing at induction time, in the historical training data, or at prediction time, in to-be-predicted test cases. In this paper, we handle missing values at induction time, and compare different imputation methods to study their effect on the classification performance. The classification performance of different classifiers is compared using misclassification error and Brier score. We use a variety of real-life data sets with missing values to reach some conclusion. Our focus is on single imputation techniques, namely, mean, k nearest neighbors, random forests and weighted nearest neighbors imputation. The main objective of this paper is to assess whether the classification accuracy is affected with randomly generated missing values comparing to the same classifier trained and tested using the complete data set. While most of the previous studies use the misclassification error only to compare the performance of difference classifiers, we use other performance measures for evaluation of the classification methods. In particular, we compute the Brier score, which is a strict measure than classification error.

The organization of the rest of the paper is as follows: After a brief overview of missing data treatment methods in Section 8.2, Section 8.3 describes the methods of imputation that are used for the comparison purpose. In Section 8.4 the performance measure to compare these classifiers are given. Section 8.5 presents the simulation based on different real data sets to compare the efficiency of imputation for different classifiers. In Section 8.6 some concluding remarks are given.

8.2. Missing Data Treatment for Classification

Approaches to handle the missing data can be categorized into the following four categories:

- *Ad-hoc methods:* These methods mainly focus on ignoring the cases with missing values and dealing with the complete cases left. Examples are list-wise deletion (or case deletion), pairwise deletion etc. The complete data obtained by this approach can be classified by any classification method, the samples that contain missing values are excluded in the classification process (García-Laencina et al., 2010). This approach does not work for smaller samples.

- *Single imputation:* The missing values are retrieved by replacing each missing value by a *plausible* value. The imputed values are assumed to be the real values that would have been observed at the time of data collection. The method does not address the uncertainty of the imputation process. Some examples are: hot-deck, mean substitution, regression imputation, etc.

- *Multiple imputation:* It addresses the uncertainty factor inherent in the imputations by providing more than one imputed value for a missing value (Rubin, 1987). As a result we have more than one imputed data sets. Multiple imputation (MI) formally comprises three stages: imputation, analysis, and combining results of the analysis. In the imputation stage, M independent imputed values are obtained corresponding to each missing value to get M complete imputed data sets. In the analysis stage, each of the M imputed data sets is analyzed using standard statistical techniques for complete data. In the third stage, M sets of the desired estimates, obtained from the imputed data sets, are combined into one set of parameter estimates.

- *Machine learning approach:* In recent years, machine learning algorithms that work without imputation, were introduced. In particular, they build classifiers that have the ability to classify directly missing data, without using imputation techniques. For example, C4.5 (Quinlan, 2014), CART (De'ath and Fabricius, 2000), CN2 (Clark and Niblett, 1989b) algorithms can handle the missing values in both test and training sets.

An appropriate imputation of missing values provides a concrete basis for further statistical analysis and drawing valid statistical inferences. A poor imputation strategy can create more problems than it solves, and can lead to wrong predictions and misclassification. According to Rubin (1987), missing data imputation is challenging because of the possible biases due to the reason that samples with missing values are often systematically different from the samples without missing values.

8.3. Imputation Methods for Missing Values

In this section, we briefly describe some existing approaches to impute missing values and performance measures used for comparison of these methods.

8.3.1. Mean imputation

The simplest approach consists of replacing the missing values with zeros or mean/average values (Alizadeh et al., 2000). This approach is more popular to impute missing values in gene expression or similar data sets. A distinct disadvantage of this method is that the information on the correlation among predictors/genes is ignored. The estimated parameters and their standard errors are biased due to the reason that variance of the variable being imputed reduces since it shifts the possible extreme values towards the center of the distribution. It has been demonstrated that imputation methods perform better than both the simple deletion of whole observations and the replacement of the missing data by zeros or average values, see, for example, Troyanskaya et al. (2001).

8.3.2. NN Imputation

This method was originally proposed by Troyanskaya et al. (2001) for the imputation of missing values. An imputed value is obtained by taking the average of the values of k candidate samples, called neighbors, which are chosen by a distance measures. For this method to perform, one needs to select a suitable value of k, and a distance measure. We use two functions to evaluate the performance of this method, (i) the function `impute` from the Bioconductor package `impute` (Hastie et al., 2013); (ii) the function `kNN` from the R package `VIM` (Templ et al., 2016). In our experiments, we used cross-validation for selecting the value of k that yielded smallest imputation error.

8.3.3. Random Forests

Random forests were introduced by Breiman (2001) as an extension of classification and regression trees (CART). The approach uses bootstrap samples of the data to build various trees. It can accommodate not only mixed types of predictors with nonlinear and complex relationships, but also deal with high-dimensional data ($p \gg n$). A random subset of the variables is used as candidates at each split. To build the trees, bootstrap aggregation (bagging) as well as random variable selection can be used. The recently introduced `missForest` procedure (Stekhoven and Bühlmann, 2012) uses the versatile and flexible random forests model to obtain an estimate of a missing value. It is an iterative procedure that uses initial imputations using mean imputation or another imputation method, and then improves the imputed data matrix on successive iterations. A random forest model is developed for each predictor with a missing value by using the rest of the predictors, and this model is used to estimate the missing value of that predictor. The imputed data matrix is updated, and the difference between previous and new imputation is assessed at the end of each iteration. The whole process is repeated until a specific criterion is met. We use the `R` package `missForest` (Stekhoven and Bühlmann, 2012) for this method.

8.3.4. wNN Imputation with Selected Features (`wNNSel`)

Let $\mathbf{X} = (x_{is})$ be an $n \times p$ data matrix consisting of n observations on p covariates. Let $\mathbf{O} = (o_{is})$ denote the corresponding $n \times p$ matrix of dummies with entries

$$o_{is} = \begin{cases} 1 & \text{if } x_{is}\text{is not mising} \\ 0 & \text{for missing value.} \end{cases}$$

The observation vector $\mathbf{x}_i$ is the ith-row in the data matrix. To compute the distance between two observations $\mathbf{x}_i$ and $\mathbf{x}_j$, we use the L_q-metric defined by the following

$$d_q(\mathbf{x}_i, \mathbf{x}_j) = \left(\frac{1}{m_{ij}} \sum_{s=1}^{p} |x_{is} - x_{js}|^q I(o_{is} = 1) I(o_{js} = 1) \right)^{1/q},$$

An improved version of the above formula proposed by Tutz and Ramzan (2015) is defined by

$$d_{q,C}(\mathbf{x}_i, \mathbf{x}_j) = \left(\frac{1}{m_{ij}} \sum_{l=1}^{p} |x_{il} - x_{jl}|^q I(o_{il} = 1) I(o_{jl} = 1) C(r_{sl}) \right)^{1/q}, \tag{8.1}$$

where r_{sl} is the empirical correlation between covariates s, l and $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the correlations into weights. This equation uses selected covariates to compute the distance between observations. The covariates, who have a higher correlation with s^{th} covariate, will get a higher weight and thus contribute more to the calculation of distances.

A linear function in absolute correlation is $C(r_{sl}) = |r_{sl}|/(1-c) - c/(1-c)$ if $|r_{sl}| > c$, $C(r_{sl}) = 0$ if $|r_{sl}| \leq c$. Thus if $|r_{sl}| \leq c$ the covariate s has no contribution to the distance. Another smoother function is $C(r_{sl}) = |r_{sl}|^m$. The power m in the function $C(r_{sl}) = |r_{sl}|^m$ and c in linear function are tuning parameters and chosen by cross validation.

Based on the distances computed by equation (8.1), the observations are arranged in ascending order is $\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(k)}$ with $d(\mathbf{x}_i, \mathbf{x}_{(1)}) \leq \cdots \leq d(\mathbf{x}_i, \mathbf{x}_{(k)})$, where $\mathbf{x}_{(j)}^T = (x_{(j)1}, \ldots, x_{(j)p})$ denotes the jth nearest neighboring observation.

The imputed value of x_{is} is defined by

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\mathbf{x}_i, \mathbf{x}_{(j)}) x_{(j)s} \tag{8.2}$$

with weights

$$w(\mathbf{x}_i, \mathbf{x}_{(j)}) = \frac{k(\frac{d(\mathbf{x}_i, \mathbf{x}_{(j)})}{\lambda})}{\sum_{h=1}^{k} k(\frac{d(\mathbf{x}_i, \mathbf{x}_{(h)})}{\lambda})}. \tag{8.3}$$

where $k(.)$ is a kernel function and λ is a tuning parameter. The results of simulation studies of Tutz and Ramzan (2015) showed that Gaussian kernel with $q = 2$ provides a smaller imputation error under different simulation settings. Moreover the power function performs slightly better than the linear function. We take advantage of these findings and use Gaussian kernel, L_2-metric ($q = 2$) and power function for imputation of missing data in the next sections.

8.4. Performance Measures

We evaluate the performance of the considered imputation methods in two ways: how well an imputation method predicts the missing values and how well the imputed data performs as classifiers? We use some accuracy measures from the **?**.

Imputation Accuracy

The accuracy of the imputation is evaluated by computing the error between the true (known) values and the imputed values from the considered methods. More specifically, we compute the mean squared imputation error by

$$\text{MSIE} = \frac{1}{n^{miss}} \sum_{x_{is}:o_{is}=0} (x_{is} - \hat{x}_{is})^2,$$

where x_{is} is the true(known) value in the complete data, $\hat{x}_{is}$ is the corresponding imputed value, and n^{miss} is the number of missing values.

Misclassification Error

The misclassification error (MCE) is defined as the rate of incorrectly classified responses. More specifically, let $Y_i \in \{0, 1\}$ denote the classes. Then the misclassification error (MCE) is defined by

$$\text{MCE} = \frac{1}{n_T} \sum_{i=1}^{n_T} I(Y_i \neq \hat{Y}_i),$$

where n_T is the number of observations in the test set, Y_i is the true value and $\hat{Y}_i$ is the corresponding predicted value. It is to note that we use the test set only to compute the error (MCE) in our experiments. For a good classifier, the MCE will be close to 0.

Brier Score

The Brier score is defined by

$$\text{BS} = \frac{1}{n_T}\sum_{i=1}^{n_T}(Y_i - \hat{P}_i)^2,$$

where n_T is the number of observations in the test set, Y_i is the true value and $\hat{P}_i$ is the estimated probability for class 1. It is similar to the mean squared error in regression. A value close to 0 shows better performance.

8.5. Experimental Evaluations

We focus only on the imputation methods and compare their efficiency to classify the data correctly. For the purpose of comparison, we compute the MCE (misclassification error) using the complete data.

8.5.1. Datasets

We use six benchmark datasets selected from the UCI Machine learning repository (Lichman, 2013). Table 8.1 describes the characteristics of these datasets. All of these date have two classes except for Glass data. We consider only two classes of the Glass data in our study. The selected datasets are complete without any missing values, we artificially create missing values under MCAR mechanism.

Table 8.1.: Description of the datasets

Data	Abbriviation	n	p
SPECTF heart data	SPECT	80	45
Wisconsin Prognostic Breast Cancer	WPBC	198	34
Parkisons Disease data	Parkinson	195	23
FRI data	FRI	100	25
Glass	Glass	146	10
Sonar	Sonar	208	60

8.5.2. Study Design

To evaluate the performance, the missing values are generated in the training data only. We split the data into training (80%) and test(20%) sample. A certain percentage of values in the training data are deleted artificially and then imputed using an imputation method. The classifier is trained on the training data and the classification accuracy is

measured on the test data. The missing values are introduced in complete datasets in the following six amounts: 10%, 20%, 30%, 40%, 50%, and 60%. The missing values

Table 8.2.: Classification methods used in the study

Abbriviation	Classifier	R package
rpart	Decision Tree	rpart
SVM	Supprt vector machine	e1071
NB	Naive Bayes	e1071
LDA	Linear discriminant analysis	MASS
sparseLDA	Sparse Linear discriminant analysis	sparseLDA
multinom	Multinomial log-linear models via neural networks	nnet
rForest	Random forests	randomForest
glmnet	GLM with Regularization	glmnet
RDA	Regularized Discriminant Analysis	klaR
ctree	Conditional Trees	party
kNN	`k` nearest neighbors	kknn
NNet	Neural networks	nnet
C5.0	C5.0 decision trees and rule-based models	C50

in each data are imputed using mean imputation, k nearest neighbors imputation (`BIO`, `VIM`), random forest imputation (`RF`) and weighted nearest neigbors imputation (`wNNSel`) methods. The performance of difference classifiers is computed for each imputed data. The whole procedure is repeated 50 times and the average results are reported. The classifier that are used in our study are summarized in Table 8.2

8.5.3. Results

We analyze different imputation approaches to find an overall best imputation method. For this purpose we compute ranks for these imputation methods based on the performance. The best value in each case gets a rank of 1 and the second best rank 2, etc. Then we take an average of these ranks over all the classifiers considered in the study. The results for differing percentages of missing data based on MCE and Brier score are given in Table 8.3 and Table 8.4 respectively. The *best* value in each case is shown in boldface. The last column of the table shows an average over all the percentages, to get an overall best imputation method.

The results for FRI data show that the `wNNSel` imputation provides the best ranking for all performance criteria (Table 8.3). Surprisingly, the second best method is the `MEAN` imputation for this data set with an average rank of 2.81 (Table 8.3) and Brier score 2.60 (Table 8.4). Whereas the `RF` imputation yields the worst results, with highest MCE and Brier score (see Table 8.3 and Table 8.4).

The average misclassification errors for different classifiers obtained from FRI data are shown in Figure 8.1. For each classifier the error is obtained using the complete data and the imputed data by five imputation methods. It can be seen in the figure, that the average

Table 8.3.: MCE: average ranks over all the classifiers

Data	Methods	10%	20%	30%	40%	50%	60%	Average
FRI	MEAN	2.73	3.20	2.77	2.60	2.67	2.90	2.81
	BIO	3.50	3.57	3.57	**2.43**	3.33	3.13	3.26
	VIM	**2.60**	3.07	3.37	3.33	3.50	3.17	3.17
	RF	3.27	2.57	3.50	4.17	3.53	3.73	3.46
	wNNSel	2.90	**2.60**	**1.80**	2.47	**1.97**	**2.07**	**2.30**
WPBC	MEAN	3.67	3.97	4.13	2.73	3.23	3.80	3.59
	BIO	3.13	**2.43**	2.50	2.83	3.17	2.87	2.82
	VIM	**2.73**	3.07	3.70	4.00	3.70	3.00	3.37
	RF	**2.73**	2.90	**2.33**	2.83	2.83	3.00	2.77
	wNNSel	**2.73**	2.63	**2.33**	**2.60**	**2.07**	**2.33**	**2.45**
SPECT	MEAN	3.53	2.73	3.10	**2.63**	3.27	3.00	3.04
	BIO	3.07	3.20	**2.63**	3.13	3.03	3.53	3.10
	VIM	3.03	3.43	3.50	3.53	3.10	3.60	3.37
	RF	**2.63**	3.47	2.83	2.97	2.83	**2.30**	2.84
	wNNSel	2.73	**2.17**	2.93	2.73	**2.77**	2.57	**2.65**
Parkinson	MEAN	4.23	4.33	4.27	4.10	4.10	3.90	4.16
	BIO	2.83	2.53	2.40	2.70	2.63	2.37	2.58
	VIM	3.13	3.67	3.23	3.87	3.53	3.63	3.51
	RF	**2.17**	2.43	2.67	2.43	**2.37**	2.63	2.45
	wNNSel	2.63	**2.03**	**2.43**	**1.90**	**2.37**	2.47	**2.31**
Glass	MEAN	4.47	4.63	4.67	4.30	4.87	4.80	4.62
	BIO	3.00	2.67	2.80	2.37	2.33	1.97	2.52
	VIM	2.97	3.77	3.90	4.20	4.13	4.20	3.86
	RF	2.47	**1.10**	**1.57**	**1.27**	**1.33**	**1.57**	**1.55**
	wNNSel	**2.10**	2.83	2.07	2.87	2.33	2.47	**2.44**
Sonar	MEAN	2.80	3.50	3.57	3.30	3.47	3.53	3.36
	BIO	3.33	**2.43**	2.60	2.90	2.60	**2.30**	**2.69**
	VIM	3.13	2.83	**2.00**	2.90	3.63	3.27	2.96
	RF	**2.80**	3.23	3.27	**2.83**	3.13	3.30	3.09
	wNNSel	2.93	3.00	3.57	3.07	**2.17**	2.60	**2.89**

MCE increases with the increasing percentage of missing values for all the classifiers. For 10% missing data, the results of imputed data and complete data are very close to each other. All the classifiers suffer from the amount of missing values. For example, the performance of SVM classifier quickly deteriorates when the missing values increase. In contrast, kNN classifier is stable.

For breast cancer data (WPBC), the `wNNSel` yields the best results according to MCE (Table 8.3). When the data contains 10% missing values, the `VIM`, `RF`, and `wNNSel` imputation methods obtain a same value of 2.73 using misclassification error (Table 8.3). For higher missing percentages, `wNNSel` method shows better performance. Overall, the `wNNSel` method obtains the best and the `RF` obtains the second best rank using MCE. Whereas based on Brier score, the `BIO` gets the best rank and `wNNSel` and `RF` methods

are the second and third best methods, but the difference among these three methods is very small. The misclassification errors for WPBC data are shown in Figure 8.2. It is seen

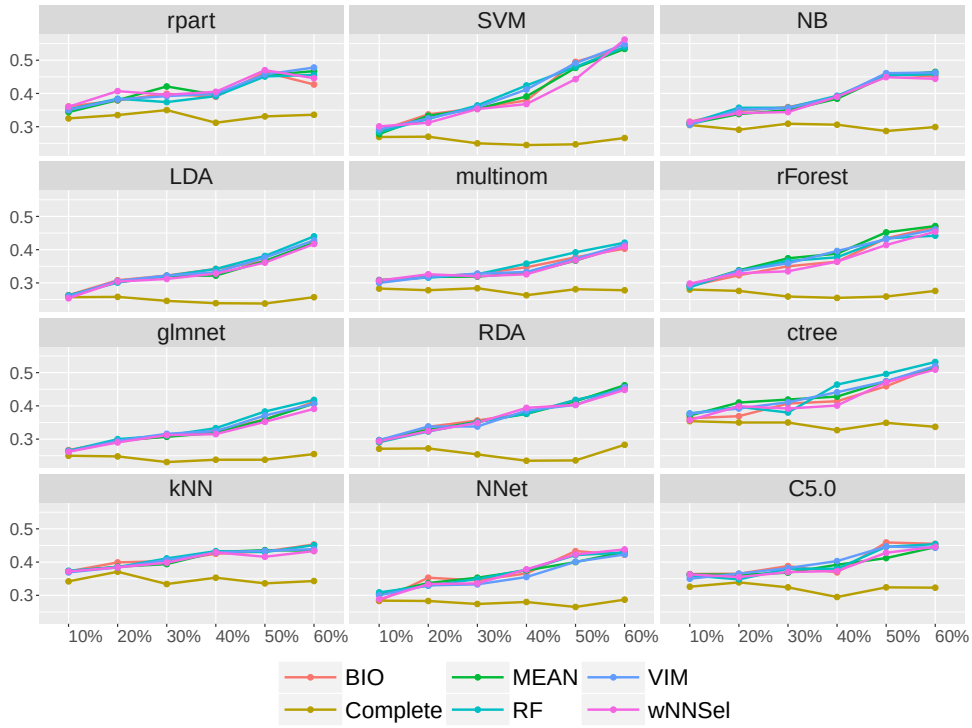


Figure 8.1.: Average misclassification error for FRI data.

that SVM classifier provides the smallest error rate and hence a better performance than other methods. It is also to note that the performance of SVM method is stable upto 30% missing missing data. The random forest (rForest) classifier also perform better for this data, whereas NB and kNN show a poor performance.

In contrast, the NB classifier performs well for SPECT data. Figure C.1 (in Appendix C) shows the performance of each individual classifier for the SPECT heart data. It can be seen that SVM, NB and rForest show better performance than other methods of classification. LDA performs very poor in this dataset. It is important to note that imputation improves the classification performance for some classifiers, for example, the error rate for LDA, multinom (for 20% to 60% missing), NNet (for 30% to 60% missing) and glmnet (for 30% to 60% missing) methods using imputation is smaller than obtained from complete data. The results for the SPECT heart data, given in Table 8.3, show that the `wNNSel` imputation perform best based on MCE. The difference between the `wNNSel` and `RF` methods in average MCE rankings is quite small (Table 8.3). They are well separated from other three imputation methods. The Brier score for these two imputation methods is same that is 2.69 (Table 8.4).

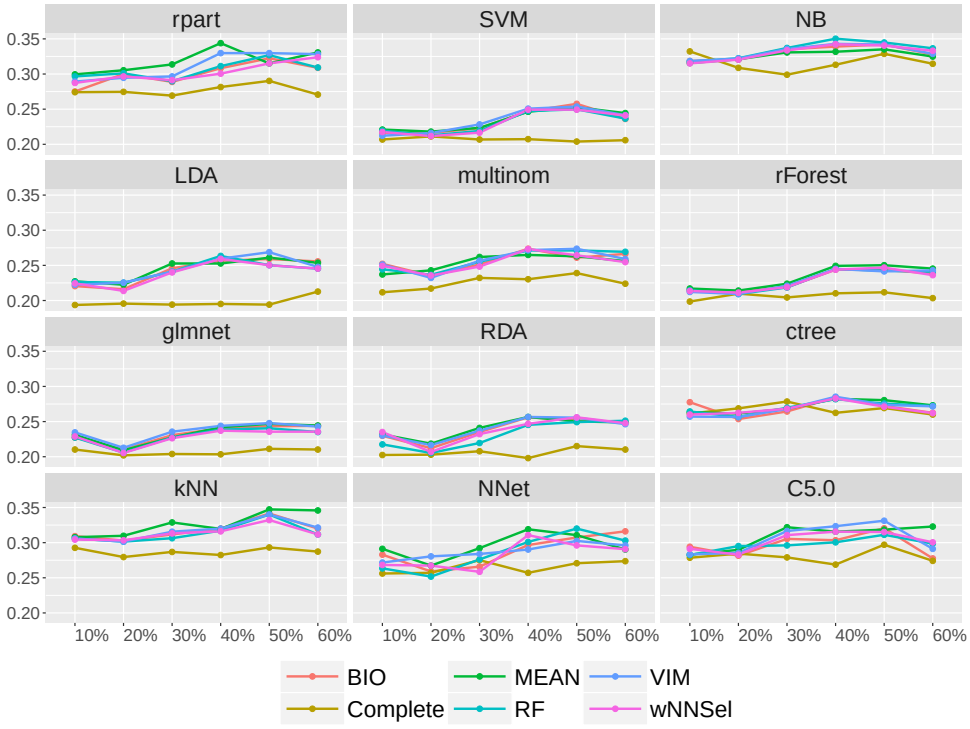


Figure 8.2.: WPBC data: average misclassification error

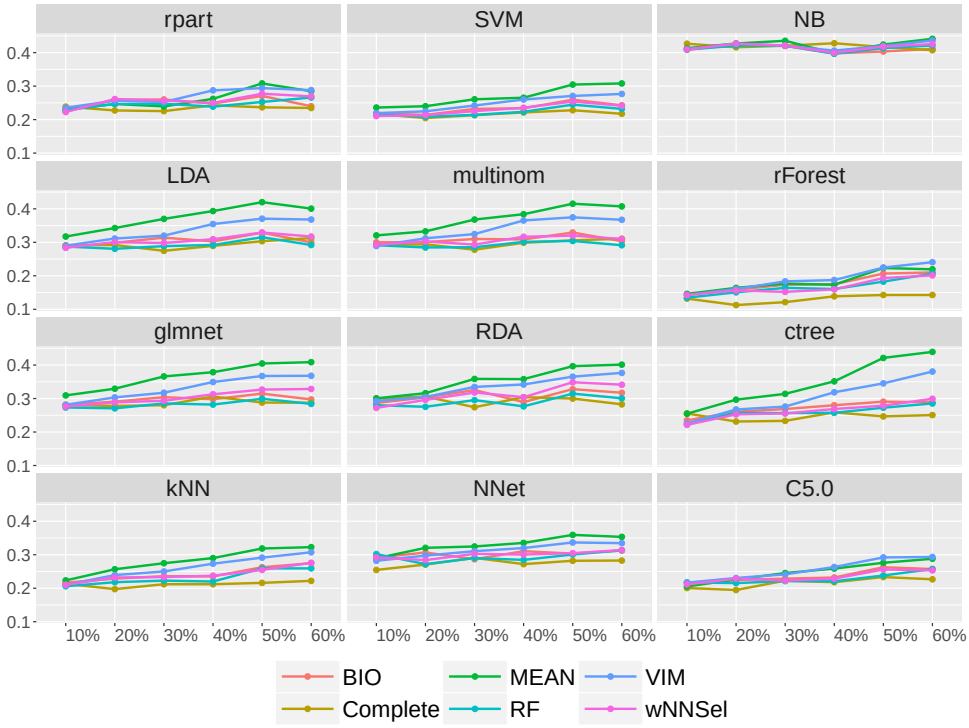


Figure 8.3.: Glass data: average misclassification error

Figure 8.3, for the Glass data, shows that the imputation results in the increase of MCE with increasing percentage of missing values. In particular, `MEAN` imputation perform very poor for all the classifiers. The best method. The performance of NB classifier is poor as compared to other methods of classification. The random forest (rForest), c5.0 and kNN yield better classification of the Glass data. The `RF` imputation provides the best ranking and the `wNNSel` imputation gets the second best ranking using MCE (Table 8.3) and Brier score (Table 8.4).

Table 8.4.: Brier score: average ranks over all the classifiers

Data	Methods	10%	20%	30%	40%	50%	60%	Average
FRI	MEAN	2.87	3.07	2.47	2.60	2.27	2.33	2.60
	BIO	2.93	3.13	3.93	2.80	3.33	3.80	3.32
	VIM	3.13	3.03	3.47	3.63	3.07	3.07	3.23
	RF	3.47	**2.73**	3.40	4.13	4.07	3.93	3.62
	wNNSel	**2.60**	3.03	**1.73**	**1.83**	**2.27**	**1.87**	**2.22**
WPBC	MEAN	3.07	4.33	4.07	3.07	3.60	4.00	3.69
	BIO	3.07	**2.20**	2.80	**2.77**	2.80	2.87	2.75
	VIM	**2.37**	3.37	3.27	3.10	3.90	3.13	3.19
	RF	3.30	2.43	2.53	3.00	2.90	2.77	2.82
	wNNSel	3.20	2.67	**2.33**	3.07	**1.80**	**2.23**	**2.55**
SPECT	MEAN	3.80	2.90	3.00	2.67	3.13	3.00	3.08
	BIO	2.93	3.30	**2.50**	3.60	3.47	3.57	3.23
	VIM	**2.67**	3.00	3.67	3.63	3.20	3.67	3.31
	RF	2.73	3.00	2.70	2.80	**2.33**	2.57	**2.69**
	wNNSel	2.87	**2.80**	3.13	**2.30**	2.87	**2.20**	**2.69**
Parkinson	MEAN	3.93	4.13	4.27	3.97	4.20	3.60	4.02
	BIO	2.57	2.80	2.67	2.87	2.67	**2.33**	2.65
	VIM	3.20	3.40	3.33	3.43	3.27	3.33	3.33
	RF	**2.30**	2.40	2.60	2.87	2.53	2.93	2.61
	wNNSel	3.00	**2.27**	**2.13**	**1.87**	**2.33**	2.80	**2.40**
Glass	MEAN	4.33	4.27	4.47	4.20	4.67	4.40	4.39
	BIO	3.27	2.67	3.07	2.33	2.47	1.73	2.59
	VIM	2.67	3.80	3.40	3.87	3.93	4.13	3.63
	RF	2.53	**1.27**	**1.73**	**1.47**	**1.67**	**2.13**	**1.80**
	wNNSel	**2.20**	3.00	2.33	3.13	2.27	2.60	**2.59**
Sonar	MEAN	2.93	3.83	3.80	3.20	3.53	3.87	3.53
	BIO	3.33	**2.50**	**2.60**	2.67	**2.53**	2.47	**2.68**
	VIM	3.07	2.60	**2.60**	3.47	3.47	3.23	3.07
	RF	**2.47**	2.80	2.93	**2.47**	2.93	3.10	2.78
	wNNSel	3.20	3.27	3.07	3.20	**2.53**	**2.33**	2.93

Similarly, for Parkinson data the `wNNSel` yields the best and the `RF` imputation yields second best results according to both performance measures. For the Sonar data, the `BIO` method provides the best rank and the `wNNSel` imputation gets the second best ranking (see Table

8.3 and Table 8.4). The misclassification errors for the Parkinson and Sonar data are shown in the Figure C.2 and Figure C.3 respectively in Appendix C.

8.6. Concluding Remarks

Missing data is often a problem in many statistical analyses like classification. Although some machine learning algorithms can internally handle the missing data, many statistical techniques require a complete data. Many imputation techniques are available in literature that fill the missing values. In this paper, we investigate the impact of imputing missing data imputation on the subsequently performed classification. To this end, we performed an experimental study on six data sets using five imputation methods. We used a variety of different classifiers to investigate the effect of imputation on the classification accuracy. The classification accuracy was measured using classification error and Brier score. We also considered imputations for six different amounts of missing data ranging from 10% to 60% for each dataset and compared the results obtained from classification of imputed data using different methods.

Our study shows that the impact of imputation varies between data sets and different classifiers. Overall the `wNNSel` imputation provided best results in four out of six datasets considered and second best for the remaining two datasets. The `RF` imputation provided the second best classification results in our experiments. The `MEAN` imputation was seen to yield poor performance in all datasets considered. Based on the above experiments, it is seen that imputation improves the classification accuracy and in particular, beneficial for the higher amounts of missing data. When the amount of missing data is small, there is not much difference among the imputation methods. It is hard to find a single classifier that perform best for all the datasets.

9. Multiple Imputation Using Nearest Neighbor Methods

Abstract

Missing values are a major problem in all areas of quantitative research. As the complete case analysis discards useful information, estimation and inference may suffer strongly. Multiple imputation has been shown to be a useful strategy to handle missing data problems and account for the uncertainty of imputation. We present multiple imputation methods based on nearest neighbors. The distances are computed using the information of correlation among the target and candidate predictors. Thus only the relevant predictors contribute to the computation of distances. The method successfully imputes missing values also in high-dimensional settings, in which existing software tend to fail. Using a variety of simulated data with MCAR and MAR missing patterns, the proposed algorithm is compared to existing methods. The performance is evaluated by using mean squared imputation errors and inference results obtained for the imputed data. Various measures are used to compare methods, including mean squared errors of estimated regression coefficients, their standard errors, confidence intervals and their coverage probabilities. The simulation results, for both cases $n < p$ and $n > p$, show that the sequential imputation using weighted nearest neighbors can be successfully applied to a wide range of data settings and outperforms or is close to the best when compared to existing methods.

9.1. Introduction

Missing values are a challenging issue for the researchers in various fields. Handling the missing values in an inadequate manner may lead to biased results and inference based on them are, in turn, often misleading. The standard statistical techniques for analyzing data require complete cases without any missing observation. The deletion of the cases with missing information to obtain complete data causes the loss of important information. One way to deal with missing data is to impute them. Imputation is the process of replacing the missing values by substitutes. Many methods for the imputation of missing values have been developed. They can be divided into two branches: *single imputation* and *multiple imputation*. In single imputation, each missing value is replaced by one value. In statistical analysis with completed data, the imputed values are treated as real values, as if they had

been actually observed. Examples of single imputation methods are mean substitution, regression imputation, hot-deck imputation (distance function approach and the pattern matching approach), last observation carried forward, nearest neighbor imputation and maximum likelihood method. Single imputation does not account for the uncertainty due to the imputation process. The point estimates obtained by single imputation can be inconsistent if the missing data mechanism goes beyond MCAR (Cranmer and Gill, 2013).

Due to the advancement and rapid growth in technology, the collection of high-dimensional data is no longer a tedious task. For *single imputation* in high-dimensional settings several approaches are available. An algorithm to impute missing data in high-dimensional settings, which is based on the popular random forests technique was proposed by Stekhoven and Bühlmann (2012). Random forests avoid overfitting by bootstrap aggregation of multiple regression trees. Moreover, the accuracy of predictions is enhanced by combining the predictions across all the trees (Breiman, 2001). An adaptation of this method and its comparison to parametric imputation methods was given by Shah et al. (2014). The results of their studies showed that the method is more efficient and typically produces confidence interval that are narrower than standard MI approaches. Four different variants of the nearest neighbor imputation method were developed by Liao et al. (2014). But all of these methods do not properly account for the uncertainty of imputation, Deng et al. (2016) regarded them as *improper* in the sense of Rubin (1987).

One of the most popular methods for handling missing values is *multiple imputation* (MI) as propagated by Rubin (1987). It is a technique to generate more than one plausible value for each missing value in data. It can provide valid results even when the missing mechanism is missing at random (MAR) (He and Belin, 2014). Multiple imputation (MI) accounts for the uncertainty due to imputation by explicitly using the estimates obtained for the different imputed datasets. To get an overall inference, estimation results obtained for each of the imputed data set are combined across all the imputed data sets using the well known Rubin's rule (Little and Rubin, 2014, Rubin, 2004). In recent years, MI has emerged as the leading approach for imputing missing data with lots of advancements in methodology and software (Harrel and Zhou, 2007; Horton and Kleinman, 2007).

For high-dimensional data, some specific methods are available. But, in general, these methods suffer from the curse of dimensionality and seem not to work very well. When the number of predictors is less than the sample size, many software packages provide an easy application of MI methods, for example, the `R` packages `mice` (van Buuren and Groothuis-Oudshoorn, 2011) and `Amelia` (Honaker et al., 2011). When the number of predictors is less than but close to the sample size, the existing methods can still be applied to get imputation results but may not perform well (Zhao and Long, 2016). However, when the number of predictors exceeds the sample size, the available software packages typically fail. Some researchers have suggested to use regularized regression methods, which allow for variable selection before building the final imputation model (Long and Johnson, 2015). For a multivariate normal model, Song and Belin (2004) presented a method to extract the common factors in high-dimensional data so that the number of parameters to be estimated is reduced. To impute missing data in high-dimensional data structures,

Zhao and Long (2016) proposed the Bayesian lasso regression for multiple imputation with normally distributed data. The approach is computationally very demanding and becomes infeasible for an increasing number of variables with missing data. Another adaptation of `mice` for high-dimensional setting was given by Deng et al. (2016). But, as mentioned in the paper, the theoretical properties of `mice` are still not well established.

Although MI methods have been very popular and the most widely used approach for imputing missing data, the validity of MI is based on various assumptions. For example, most of the MI methods assume that the missing data follow the MAR mechanism (Little and Rubin (2002)). Another major issue is the correct (or at least close to the true model) specification of the imputation model (Rubin, 1996, Akaike, 1974). Another drawback of conditional MI is that it may not converge to the correct posterior distribution, similar to the Gibbs sampler.

Nearest neighbors (NN) is a popular non-parametric technique that has been widely used for single value imputation (Troyanskaya et al., 2001). Many variants have been proposed in the literature as improvements over the original kNN method (Ouyang et al. (2004), Kim et al. (2004), and Scheel et al. (2005), Johansson and Hakkinen (2006), Bø et al. (2004) and Zhang et al. (2008), etc.). An adaptation of nearest neighbor imputation was given by Verboven et al. (2007) for the imputation of missing values sequentially. They successfully applied the method to gene expression data, and Branden and Verboven (2009) provided a robust version of their method. Tutz and Ramzan (2015) proposed a weighted nearest neighbor approach to impute missing data. It was shown that the method provides better imputation results when compared to existing approaches. The approach is particularly useful when the data suffers from the curse of dimensionality (Faisal and Tutz, 2017c). Since these methods impute only one plausible value for each missing value, they share all the drawbacks of single imputation.

The objective of the paper is to demonstrate that NN methods can also be used for multiple imputation, which has advantages, in particular when the number of covariates is large when compared to the sample size (for example in genetics studies). In particular we propose two methods: (i) a combination of the bootstrap and NN (`miNNboot-miNNboot`) (ii) a sequential NN imputation method (`miNNseq-miNNseq`). We are not aware of any other publications based on multiple imputation using nearest neighbors. The proposed methods are compared with the widely used MI method Multivariate Imputation by Chained Equations (MICE). We report the results of imputation errors, and investigate the performance of inference based on the imputed data in two ways; (i) the accuracy of the estimated coefficients (ii) confidence intervals of the estimated coefficients.

The paper is organized as follows: Section 9.2 describes the general multiple imputation technique and Rubin's method for combining the results. Some methods for reaching inferences after MI are also given in this section. In Section 9.3, the nearest neighbor approach to impute missing values is given in detail. The two proposed algorithms for MI are also given in this section. Section 9.4 presents a comprehensive simulation study to investigate the performance of proposed methods in a variety of low and high-dimensional data set-

tings. The results obtained from real data are given in Section 9.5. Finally, Section 9.6 gives some concluding remarks.

9.2. Multiple Imputation

MI is a preferable choice for data analysts due to its flexibility and its applicability in a wide variety of missing data scenarios. Before proceeding to MI, we briefly sketch the missing mechanism under which missing values can occur. Rubin (1976) categorized the missing mechanisms into three categories: (i) *missing completely at random* (MCAR: missing pattern results from a completely random process and the probability of missingness is the same for all units); (ii) *missing at random* (MAR: the probability that a variable is missing depends only on the available information and not on the observations that are missing in the data, e.g., females are less likely to answer age question (with gender completely observed), or the weight of individuals measured with high blood pressure only); (iii) *missing not at random* (MNAR: the probability of a missing data value depends on the missing data itself, e.g., respondents with high income are less likely to report their income).

The standard implementations of MI usually assume MAR. Although MAR is a non-testable assumption, one hopes to get very close to MAR if enough variables are included in the imputation model. Multiple imputation can also be implemented with an MNAR pattern (Rubin, 1987; Carpenter et al., 2007). However, the estimates based on MI are asymptotically unbiased under MAR and MCAR (White et al., 2011), but potentially biased under MNAR (Schafer and Graham, 2002; Newman, 2009).

Application of multiple imputation involves three stages:

(i) *Imputation*, that is, each missing value of the data is imputed $M \geq 2$ times. This stage results in M complete data sets.

(ii) *Analysis*, that is, each complete imputed dataset is analyzed independently using standard statistical techniques for complete data.

(iii) *Pooling*, that is, the estimates of the M analyses are combined into one set of parameter estimates. When pooling the estimates one accounts for the missing data uncertainty and sampling variation.

Rubin (1987) provided some rules to combine the estimated parameters and their standard errors. In the following subsection, we briefly describe Rubin's rule for combining estimates of the parameters obtained by multiple imputation of missing data.

Rubin's Rule and Inference

In multiple imputation, M imputed datasets are obtained, and each of these imputed datasets is analyzed using the standard statistical techniques for complete data. The estimated parameters and their standard errors obtained by analyzing the M imputed datasets are combined for further inferences. Rubin's rule (Rubin, 1987; Little and Rubin, 2002) is used to obtain the combined estimate.

More specifically, suppose a linear regression model is fitted to each of the imputed datasets separately, yielding the estimates of the regression coefficients denoted by $\hat{\boldsymbol{\beta}}^m = (\hat{\beta}_1^m, \ldots, \hat{\beta}_p^m)$ for $m = 1, 2, \ldots, M$. Let $\hat{\boldsymbol{W}}^m$ be the estimated variance-covariance matrix for the coefficient estimate $\hat{\boldsymbol{\beta}}^m$.

When the data contains no missing values, inference using complete data is based on

$$(\hat{\boldsymbol{\beta}}_{complete} - \boldsymbol{\beta}) \sim \ N(\mathbf{0}, \mathbf{W}).$$

where $\mathbf{W}$ is the variance-covariance matrix for the coefficients $\hat{\boldsymbol{\beta}}_{complete}$ estimated using the complete data.

When the data contains missing values, a combined estimate of $\boldsymbol{\beta}$ using M imputations is obtained by

$$\hat{\boldsymbol{\beta}} = \frac{1}{M} \sum_{m=1}^{M} \hat{\boldsymbol{\beta}}^m.$$

The variance of $\hat{\boldsymbol{\beta}}$ has two components: the average within-imputation variance

$$\hat{\mathbf{W}} = \frac{1}{M} \sum_{m=1}^{M} \hat{\boldsymbol{W}}^m.$$

and the between-imputation variance

$$\hat{\mathbf{B}} = \frac{1}{M-1} \sum_{m=1}^{M} (\hat{\boldsymbol{\beta}}^m - \hat{\boldsymbol{\beta}})(\hat{\boldsymbol{\beta}}^m - \hat{\boldsymbol{\beta}})'.$$

The combined variance associated with $\hat{\boldsymbol{\beta}}$ is given by

$$\hat{\mathbf{V}} = \hat{\mathbf{W}} + \left(\frac{M+1}{M}\right)\hat{\mathbf{B}}.$$

For sufficiently large sample size and M, one can use the normal distribution to obtain inference results for $\hat{\boldsymbol{\beta}}$ (Rubin, 1987) by using

$$(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta}) \sim \quad N(\mathbf{0}, \hat{\mathbf{V}})$$

When the value of M is small, Rubin and Schenker (1986) proposed to use the t-distribution instead of the normal distribution. Then the confidence intervals are based on the Student's t-distribution (Rubin and Schenker, 1986; Rubin, 1987). For large sample size one uses

$$(\hat{\beta}_j - \beta_j)\hat{V}_j^{-1/2} \sim \; t_{df_{Rubin}},$$

where $\hat{V}_j$ is the jth diagonal element of $\hat{\mathbf{V}}$ and the degrees of freedom are given by

$$df_{Rubin} = (M-1)\Big(1 + \frac{M}{M+1} \cdot \frac{\bar{W}_j}{B_j}\Big)^2.$$

The approximation of degrees of freedom uses the between and within imputation variance along with the number of multiple imputations to calculate degrees of freedom. It is based on the assumption that the number of observations in the sample are infinite. For the case of a small number of observations, Lipsitz et al. (2002) proposed an alternative way to approximate the degrees of freedom that also uses the number of observations in the sample. Their simulation studies showed that for smaller samples, the approximation is more exact. In our simulations, we use both approximations; the one proposed by Rubin and Schenker (1986) and the method of Lipsitz et al. (2002).

9.3. Multiple Imputation Based on Nearest Neighbors

In this section, we briefly describe our weighted nearest neighbor approach to impute missing data. We propose to combine this imputation approach with bootstrap and sequential imputation to get multiply imputed data.

Let $\boldsymbol{X} = (x_{is})$ be an $n \times p$ data matrix consisting of n observations on p covariates. Let $\boldsymbol{O} = (o_{is})$ denote the corresponding $n \times p$ matrix of dummies with entries

$$o_{is} = \begin{cases} 1 & \text{if } x_{is} \text{ is not mising} \\ 0 & \text{for missing value.} \end{cases}$$

The observation vector $\boldsymbol{x_i}$ is the ith-row in the data matrix. To compute the distance between two observations $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$, one may use the L_q-metric defined by

$$d_q(\boldsymbol{x_i}, \boldsymbol{x_j}) = \left(\frac{1}{m_{ij}} \sum_{s=1}^{p} |x_{is} - x_{js}|^q I(o_{is} = 1) I(o_{js} = 1) \right)^{1/q},$$

An improved version of the above formula proposed by Tutz and Ramzan (2015) is given by

$$d_{q,C}(\boldsymbol{x_i}, \boldsymbol{x_j}) = \left(\frac{1}{m_{ij}} \sum_{l=1}^{p} |x_{il} - x_{jl}|^q I(o_{il} = 1) I(o_{jl} = 1) C(r_{sl}) \right)^{1/q}, \tag{9.1}$$

where r_{sl} is the empirical correlation between covariates s, l and $C(.)$ is a convex function defined on the interval $[-1, 1]$ that transforms the correlations into weights. This distance measure uses selected covariates to compute the distance between observations. The covariates that have a higher correlation with the s-th covariate get a higher weight and thus contribute more to the calculation of distances.

A linear function in absolute correlation is $C(r_{sl}) = |r_{sl}|/(1-c) - c/(1-c)$ if $|r_{sl}| > c$, $C(r_{sl}) = 0$ if $|r_{sl}| \leq c$. Thus if $|r_{sl}| \leq c$ the covariate s does not contribute to the distance. An alternative smooth function is $C(r_{sl}) = |r_{sl}|^m$. The power m in the function $C(r_{sl}) = |r_{sl}|^m$ and c in linear function are tuning parameters and chosen by cross validation.

Based on the distances computed by equation (9.1), the observations are arranged in ascending order is $\boldsymbol{x_{(1)}}, \ldots, \boldsymbol{x_{(k)}}$ with $d(\boldsymbol{x_i}, \boldsymbol{x_{(1)}}) \leq \cdots \leq d(\boldsymbol{x_i}, \boldsymbol{x_{(k)}})$, where $\boldsymbol{x_{(j)}^T} = (x_{(j)1}, \ldots, x_{(j)p})$ denotes the jth nearest neighboring observation.

The imputed value of x_{is} is defined by

$$\hat{x}_{is} = \sum_{j=1}^{k} w(\boldsymbol{x_i}, \boldsymbol{x_{(j)}}) x_{(j)s} \tag{9.2}$$

with weights

$$w(\boldsymbol{x_i}, \boldsymbol{x_{(j)}}) = \frac{k(\frac{d(\boldsymbol{x}_i, \boldsymbol{x}_{(j)})}{\lambda})}{\sum_{h=1}^{k} k(\frac{d(\boldsymbol{x}_i, \boldsymbol{x}_{(h)})}{\lambda})}. \tag{9.3}$$

where $k(.)$ is a kernel function and λ is a tuning parameter. The results of simulation studies of Tutz and Ramzan (2015) showed that Gaussian kernel with $q = 2$ provides a smaller imputation error under different simulation settings. Moreover, the power function performs slightly better than the linear function. We take advantage of these findings and use Gaussian kernel, L_2-metric ($q = 2$) and the power function for imputation of missing data in the next sections. In the following the method is referred to `wNNSel`.

9.3.1. Sequential Multiple Imputation

Our first approach is based on imputing the missing values in a sequential order. The missing values are imputed one at a time using the `wNNSel` method. After imputing one missing value, it is considered as observed to obtain an updated matrix which is used for the imputation of the next missing value. Thus each imputed value contributes to the imputation of others. For each data set the first value to be imputed is selected at random.

Algorithm `miNNseq`

The following algorithm describes the proposed `miNNseq` procedure for imputing the missing values in the data matrix $\boldsymbol{X}$ sequentially.

1. Impute the incomplete data matrix $\boldsymbol{X}$ using the `wNNSel` method, to get the tuning parameters λ and m.
2. Repeat, to get M multiply imputed data sets
   a) Select *one* missing value, say x_{is}, at random in the data matrix $\boldsymbol{X}$.
   b) Calculate imputation estimate of the missing value, i.e., $\hat{x}_{is}$ using `wNNSel`.
   c) Update $\boldsymbol{X}$ to $\boldsymbol{X}^*$ by replacing x_{is} with $\hat{x}_{is}$
   d) Randomly select the next missing value in $\boldsymbol{X}^*$ and repeat the process until all missing values in the original data matrix $\boldsymbol{X}$ have been imputed.

9.3.2. Multiple Imputation using Bootstrap

Alternatively, we make use of the famous bootstrap algorithm to impute missing values. More specifically, we propose to select M bootstrap samples, with replacement, of the same size n from the original missing data matrix. Then we impute missing values for each of the bootstrap samples using the `wNNSel` method. When using bootstrap the sampling as well as the imputation uncertainty is taken into account.

Algorithm `miNNboot`

An outline of the `miNNboot` method for imputing the missing values in the data matrix $\boldsymbol{X}$ using bootstrap sampling is given by

1. Impute the incomplete data matrix $\boldsymbol{X}$ using `wNNSel` method, to get the tuning parameters λ and m.

2. Draw M bootstrap samples of n rows with replacement from the missing data to get bootstrapped datasets

3. Repeat, to get M multiply imputed data sets

   a) Select a bootstrap sample of size n with replacement from $\boldsymbol{X}$.

   b) Impute all the missing values by `wNNSel` method using tuning parameters obtained in step 1.

It should be noted that the bootstrap samples may not contain exactly the same number of missing values as the original missing dataset.

For applying the `wNNSel` method, one needs to specify the tuning parameters. To this end, one can choose the optimal values of the tuning parameters in the beginning, and use them to impute the missing bootstrap samples. Another choice would be to select the tuning parameters for each respective bootstrap sample. Of course, the later procedure increases the computational burden. We compared both approaches in our initial simulations, and found that the obtained results do not differ significantly. To reduce the computational burden, we proceed with the former approach, that is, we impute the bootstrap samples using the optimal values of the tuning parameters of the original data matrix.

For comparison, we apply the same bootstrap algorithm to the other available methods for single imputation, namely mean, k nearest neighbors and random forests imputation. In our evaluations for the k nearest neighbor imputation, we chose the `impute` function from Bioconductor (Hastie et al., 2013) and the `kNN` function from package `VIM` (Templ et al., 2016). We used the `R` package `missForest` (Stekhoven and Bühlmann, 2012) for the implementation of random forests method. We denote these methods as `MEAN`, `BIO`, `VIM`, and `RF` for presentation in the final tables and figures.

9.3.3. Existing Approaches for Multiple Imputation

Before proceeding to the simulations studies, we give a brief introduction of the available MI methods. The commonly used software packages include `mice`, `VIM`, and `Amelia`. The package `Amelia` has a wide popularity in the multiple imputation literature, and provides good results especially when the data follow a normal distribution. But as the the number of predictors increases, it tends to fail. It is also sensitive to the presence of high correlation among the predictors. In numerous datasets, especially with higher number of predictors, it failed to impute the missing values. As we are also considering high-dimensional data and high correlation among predictors, we skipped this method in the final presentation of our simulation results.

The other two packages (`mice` and `VIM`) use the linear model (LM) or the generalized linear model (GLM) (according to the distributional form of the dependent variable), which requires that $n \geq p$. When fitting an imputation model in the $p > n$ situation, both packages ignore some predictors to achieve $n \geq p$. `mice`, in fact, attempts to reduce the

predictor space before fitting each imputation model, regardless of the values of n and p, by removing some linear dependencies. It excludes the predictors from the model in a similar fashion to Algorithm 2. `VIM`, alternatively, fits the imputation model with the first n predictors for the $p > n$ case. Moeover, it provides imputations that are more optimistic because they are based on the fitted values and no uncertainty is involved, which is the basic concept of MI. As a result, `VIM` sometimes provides low imputation error but leads to poor inference, which is also seen in our simulation study.

9.3.4. Evaluation of Performance

A good imputation algorithm is supposed to provide imputed values as close to the corresponding original values as possible. For the assessment of the performance of imputation methods, the Mean Squared Imputation Error (MSIE) is used, which is defined by

$$\text{MSIE} = \frac{1}{M}\sum_{M}\left(\frac{1}{n^{miss}}\sum_{x_{is}:o_{is}=0}(x_{is}-\hat{x}_{is})^2\right), \tag{9.4}$$

where M is the number of multiple imputations, x_{is} is the true value in the complete data matrix, $\hat{x}_{is}$ is the corresponding imputed value, and n^{miss} is the number of missing values in the data matrix. For all the considered MI algorithms, $M = 5$ imputed data sets were generated.

To see the effect of imputation on the inferences, we fit a linear regression model to the imputed data and compare the regression coefficients using MSE($\hat{\boldsymbol{\beta}}$). The mean squared errors of the estimated regression coefficients ($\hat{\boldsymbol{\beta}}$) is defined by

$$\text{MSE}(\hat{\boldsymbol{\beta}}) = \frac{1}{M}\sum_{M}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})^2,$$

where$M$ is the number of multiply imputed data sets. Moreover, confidence interval are constructed and the coverage probabilities of these confidence intervals are computed to see if the nominal coverage is obtained or not.

9.4. Simulation Studies

This section presents an extensive simulation study to compare the performance of the proposed algorithms `miNNseq` and `miNNboot` with some available methods for multiple imputation. We use two different data scenarios: $n < p$ and $n > p$. In the first subsection we compare the performance in terms ofthe mean squared imputation error (MSIE). Then the comparison is using MSE($\hat{\boldsymbol{\beta}}$) and their coverage probabilities.

9.4.1. Simulation Study ($n > p$)

In the first simulation study, we consider two different sample sizes $n = 50, 100$. For $n = 50$ we set $p = 15,\ 30$, and for $n = 100$ we set $p = 15,\ 45,\ 75$, where n indicates the sample size and p is the number of predictors. In each setting, $S = 200$ datasets are generated from a multivariate normal distribution with $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma}$ follows a specific form. The correlation structure we consider is the exponential or *autoregressive* type. An AR(1) correlation matrix of order one is defined by

$$\mathbf{\Sigma} = (\rho^{|i-j|}),$$

for $i, j = 1, \ldots, p$, where ρ denotes the pairwise correlation between predictors and p the number of variables.

In each case, the response variable $\mathbf{y}$ in the complete data is generated using the linear regression model. $\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \epsilon$, where the error term follows a standard normal distribution i.e., $\epsilon \sim N(0, 1)$. The true values of regression coefficients $\boldsymbol{\beta}$ are chosen from a uniform distribution, $unif(-1, 1)$. In the simulation study, we let $\mathbf{y}$ to be fully observed and generate the missing data in $\mathbf{X}$ only using the MCAR and MAR mechanisms. Let O_{ij} be the missing observation indicator for the ith observation in the jth variable. If $O_{ij} = 0$ denotes a missing x_{ij} and $O_{ij} = 1$ denotes an observed x_{ij}, the values were missed at random using the logit function:

$$\text{logit}[\Pr(O_{ij} = 0 \mid \mathbf{x}_k, \mathbf{y})] = \gamma_0 + \gamma_1 \mathbf{y} + \gamma_k \mathbf{x}_k \qquad \text{for } j \neq k, \tag{9.5}$$

with $\mathbf{x}_k$ randomly selected from the data matrix $\mathbf{X}$. We set $\gamma_1 = \gamma_k = 0.5$ and tune γ_0 to achieve the approximate desired missing percentage in the data matrix. In each sample, 10%, 20% and 30% of the total values were replaced by values. Moreover, We set $M = 5$ multiply imputed datasets for all the methods throughout.

Effect on Imputation Error

Our objective is to compare the ability of imputation algorithms to estimate the missing values themselves. For this purpose we compute the MSIE for the imputed data sets using different MI methods. The average MSIEs obtained from different MI methods are reported in Table 9.1. The boldface values show the smallest MSIE for that particular setting. The methods that use bootstrap sampling algorithm are shown as `MEAN`, `BIO`, `VIM`, `RF` and `miNNboot`. It is obvious that the proposed algorithm `miNNseq`, which uses the sequential imputation method, provides smaller imputation errors compared to the existing approaches. The `miNNseq` algorithm has the smallest values of MSIEs throughout all settings. The second best method is `miNNboot`, which uses NN and bootstrap sampling. It is strongly competitive and dominates `RF` in most scenarios. The other imputation methods, in particular `MEAN`, `BIO`, `VIM`, show poor performance in terms of average MSIEs.

Table 9.1.: Average MSIEs for different multiple imputation methods.

n	p	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
50	15	MCAR	10%	1.0349	0.4569	0.4596	0.3169	0.2599	**0.2013**	0.3386	0.4103
			20%	1.0604	0.5096	0.5436	0.3505	0.3167	**0.2490**	0.4216	0.4819
			30%	1.0553	0.5517	0.6031	0.3787	0.3759	**0.2875**	0.5327	0.5993
		MAR	10%	1.7362	0.7264	0.6434	0.4745	0.4068	**0.3443**	0.8415	0.6441
			20%	1.6611	0.7256	0.6764	0.4951	0.4680	**0.4021**	0.9662	0.7346
			30%	1.5762	0.7415	0.7062	0.5250	0.5197	**0.4379**	1.0597	0.8370
	30	MCAR	10%	1.0714	0.6258	0.6017	0.3673	0.2450	**0.1942**	0.6102	0.7125
			20%	1.0714	0.6802	0.6837	0.3989	0.2967	**0.2057**	0.6935	0.9776
			30%	1.0717	0.7155	0.7465	0.4317	0.3547	**0.2168**	0.8017	1.3494
		MAR	10%	1.3541	0.7525	0.6836	0.4652	0.3450	**0.3055**	0.9590	0.9133
			20%	1.3621	0.8357	0.7780	0.5407	0.4490	**0.3495**	1.0767	1.2648
			30%	1.3803	0.9063	0.8354	0.6203	0.5166	**0.3832**	1.1679	1.7039
100	15	MCAR	10%	1.0188	0.4035	0.4057	0.2483	0.2125	**0.1591**	0.2820	0.3159
			20%	0.9972	0.4297	0.4791	0.2695	0.2649	**0.1769**	0.3064	0.3498
			30%	1.0013	0.4901	0.5543	0.2999	0.3268	**0.2095**	0.3525	0.4079
		MAR	10%	1.9157	0.7280	0.6389	0.4121	0.4170	**0.3005**	0.7273	0.5839
			20%	1.6536	0.7572	0.6950	0.4567	0.4983	**0.3733**	0.8714	0.6312
			30%	1.5644	0.8070	0.7620	0.5080	0.5723	**0.4490**	0.9706	0.6786
	45	MCAR	10%	0.9749	0.5797	0.5858	0.3188	0.2179	**0.1690**	0.3940	0.4693
			20%	1.0027	0.6320	0.6451	0.3490	0.2700	**0.1938**	0.4760	0.6053
			30%	0.9830	0.6790	0.6851	0.3786	0.3322	**0.2804**	0.5681	0.8169
		MAR	10%	1.5199	0.8205	0.7958	0.4790	0.3669	**0.3036**	0.7684	0.6636
			20%	1.4113	0.8780	0.8273	0.5108	0.4317	**0.3731**	0.9117	0.8110
			30%	1.3700	0.9315	0.8636	0.5489	0.4948	**0.4033**	1.0153	1.0120
	75	MCAR	10%	1.0260	0.7033	0.7214	0.3866	0.2325	**0.1991**	0.7334	1.0499
			20%	1.0235	0.7359	0.7497	0.4111	0.2798	**0.2191**	0.7647	1.4550
			30%	1.0337	0.7829	0.8045	0.4465	0.3437	**0.2474**	0.8258	2.3243
		MAR	10%	2.8758	2.0154	1.6016	1.3780	0.8692	**0.6982**	2.3689	1.3301
			20%	2.4308	1.8543	1.4099	1.3189	0.9032	**0.6673**	2.2397	1.7193
			30%	2.3014	1.8863	1.3904	1.3767	0.9991	**0.7608**	2.1654	2.6813

Effect of Imputation on Estimation

The results of MSIEs in the previous subsection show that the proposed method provides smaller imputation errors in all settings considered. However, smaller imputation errors do not always guarantee better inference. In this subsection we investigate the performance of imputation methods with regard to inference. We focus on the problem of estimating mean square errors of the regression coefficients and the confidence interval coverage of the estimated parameters in a linear regression model.

We generated $M = 5$ imputed datasets and considered all multiple imputation methods. Then a linear regression model was fitted on each of the imputed data sets and the results of the $M = 5$ imputations were combined using Rubin's rule (subsection 9.2) to obtain $\hat{\boldsymbol{\beta}}$

Table 9.2.: Average MSE($\hat{\boldsymbol{\beta}}$) for different imputation methods.

n	p	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
50	15	MCAR	10%	0.3563	0.3520	0.3568	0.3661	0.3716	**0.2958**	0.3628	0.3354
			20%	0.3628	0.3605	0.3611	0.3791	0.3870	**0.3075**	0.4886	0.3957
			30%	0.3739	0.3717	0.3781	0.3972	0.4001	**0.3097**	1.2359	0.4467
		MAR	10%	0.3712	0.3652	0.3689	0.3778	0.3857	**0.2934**	0.3409	0.3097
			20%	0.3827	0.3780	0.3699	0.3930	0.3872	**0.3149**	0.4300	0.3722
			30%	0.4178	0.3999	0.3821	0.4163	0.4219	**0.3217**	0.6338	0.4514
	30	MCAR	10%	0.2082	0.2051	0.2048	0.2033	0.2025	**0.2024**	0.2040	0.2032
			20%	0.2114	0.2081	0.2085	0.2055	**0.2040**	0.2042	0.2069	0.2062
			30%	0.2150	0.2111	0.2115	0.2067	**0.2052**	0.2059	0.2078	0.2074
		MAR	10%	0.2089	0.2050	0.2067	0.2032	**0.2026**	0.2049	0.2074	0.2054
			20%	0.2119	0.2080	0.2094	0.2057	**0.2047**	0.2069	0.2097	0.2079
			30%	0.2161	0.2112	0.2123	0.2064	**0.2053**	0.2071	0.2111	0.2111
100	15	MCAR	10%	0.1482	0.1408	0.1414	0.1422	0.1420	**0.1220**	0.1299	0.1267
			20%	0.1661	0.1491	0.1525	0.1496	0.1532	**0.1228**	0.1627	0.1440
			30%	0.1697	0.1607	0.1613	0.1619	0.1618	**0.1323**	0.3373	0.1743
		MAR	10%	0.1502	0.1388	0.1396	0.1422	0.1404	**0.1197**	0.1308	**0.1197**
			20%	0.1579	0.1483	0.1478	0.1497	0.1494	**0.1234**	0.1511	0.1326
			30%	0.1614	0.1504	0.1477	0.1508	0.1491	**0.1272**	0.1863	0.1547
	45	MCAR	10%	0.2906	0.2871	0.2880	0.2880	0.2885	**0.1938**	0.2215	0.2202
			20%	0.3047	0.3023	0.3024	0.3026	0.3002	**0.2023**	0.2975	0.3145
			30%	0.3192	0.3120	0.3132	0.3170	0.3181	**0.2092**	0.6506	0.4465
		MAR	10%	0.3067	0.2962	0.2957	0.2932	0.2911	**0.1986**	0.2264	0.2237
			20%	0.3118	0.3032	0.3020	0.3047	0.3017	**0.2034**	0.2587	0.2540
			30%	0.3235	0.3167	0.3094	0.3149	0.3123	**0.2100**	0.3568	0.3563
	75	MCAR	10%	0.0161	0.0153	0.0153	0.0146	0.0143	**0.0129**	0.0139	0.0146
			20%	0.0196	0.0178	0.0179	0.0160	0.0155	**0.0141**	0.0165	0.0180
			30%	0.0241	0.0215	0.0217	0.0184	0.0177	**0.0154**	0.0193	0.0258
		MAR	10%	0.0371	0.0291	0.0244	0.0253	0.0210	**0.0178**	0.0277	0.0180
			20%	0.0533	0.0411	0.0304	0.0332	0.0275	**0.0226**	0.0428	0.0254
			30%	0.0670	0.0540	0.0377	0.0435	0.0353	**0.0281**	0.0561	0.0423

and their standard errors. The results of MSE($\hat{\boldsymbol{\beta}}$) obtained from the different imputation methods are given in Table 9.2. The smallest values in each setting are shown in boldface. It is obvious that the proposed method `miNNseq` has the smallest MSE($\hat{\boldsymbol{\beta}}$), except for $n = 50,\ p = 30$ where minimal values are achieved by the `miNNboot` method. For only one setting with $n = 100,\ p = 15$ and 10% MAR data, the two methods `MICE` and `miNNboot` yield the smallest value of 0.1197. Thus the proposed methods yield not only smaller MSIEs but also MSE($\hat{\boldsymbol{\beta}}$). The breakdown of MSE($\hat{\boldsymbol{\beta}}$) into variance and squared bias is given in Table D.3.

Effect of Signal-to-Noise Ratio on Imputation Error

We now briefly investigate the performance using the simulated data having different signal-to-noise ratios. A signal-to-noise (SNR) ratio is defined as the ration of $\text{Var}(\mathbf{X}\boldsymbol{\beta})$ to $\text{Var}(\epsilon)$. We generate the simulated data with low and high SNR using $\epsilon \sim N(0, 1)$ and $\epsilon \sim N(0, 3)$ respectively. The results for average MSIEs for $n = 50$ are shown in Figure 9.1.It can be seen that the proposed methods `miNNboot` and `miNNseq` yield the smallest MSIEs in all the settings. It is also noteworthy that `MICE` yields the highest MSIEs for both low and high SNR data. The detailed average results of MSIEs in other settings for low and high SNR are presented in Table D.1 (Appendix D).

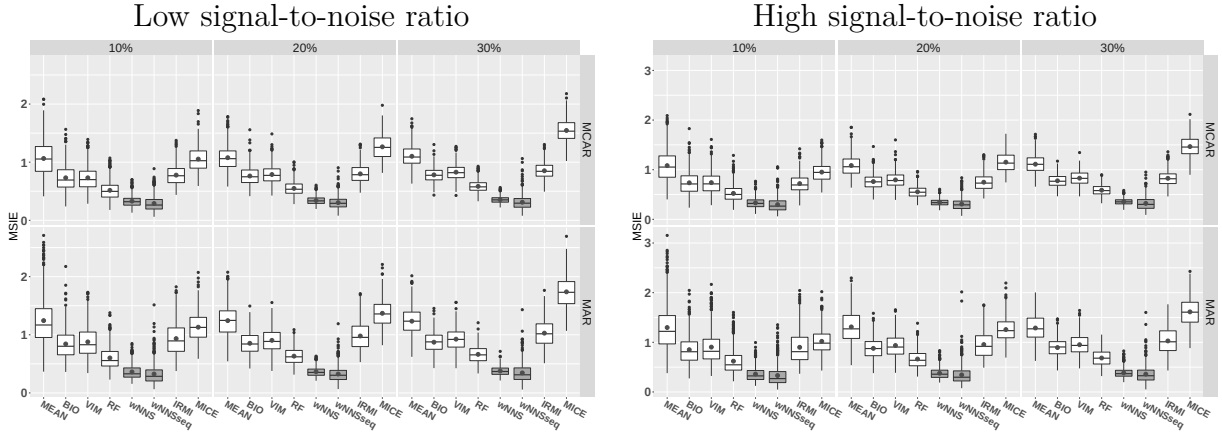


Figure 9.1.: Comparison of MSIE (mean squared imputation errors) for different multiple imputation methods using $n = 50$ and $p = 30$. Used algorithms are shown in gray boxes.

Effect of Signal-to-Noise Ratio on Inference

To see the effect of signal-to-noise ratio on the estimated parameters, we follow the same procedure as in previous subsection. We estimate the regression coefficients by fitting a linear model and compute $\text{MSE}(\hat{\boldsymbol{\beta}})$. The results for average $\text{MSE}(\hat{\boldsymbol{\beta}})$ using $n = 50$ and $p = 30$ are shown in Figure 9.2. It is interesting to note that `mice` shows good results inspite of yielding poor MSIEs (Figure 9.1). In contrast, `miNNboot` algorithm shows very high values of $\text{MSE}(\hat{\boldsymbol{\beta}})$ whereas its performance was better in terms of MSIE (Figure 9.1). All the methods based on bootstrap sampling show poor performance compared to the other algorithms. Overall, the `miNNseq` algorithm provides the best results although `mice` is very close in some cases.

Confidence Intervals

The main advantage of multiple imputation is that confidence intervals can be computed. It is in particular interesting how the differing methods behave in terms of the widths

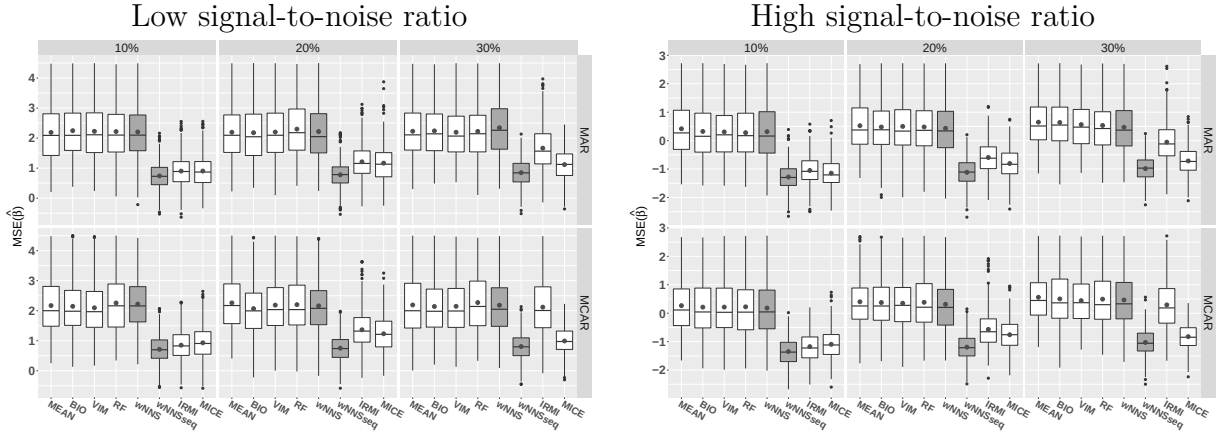


Figure 9.2.: Comparison of MSE($\hat{\boldsymbol{\beta}}$) (on log scale) for different multiple imputation methods using low and high signal-to-noise ratio for $n = 50$ and $p = 30$. Proposed methods are shown in gray boxes.

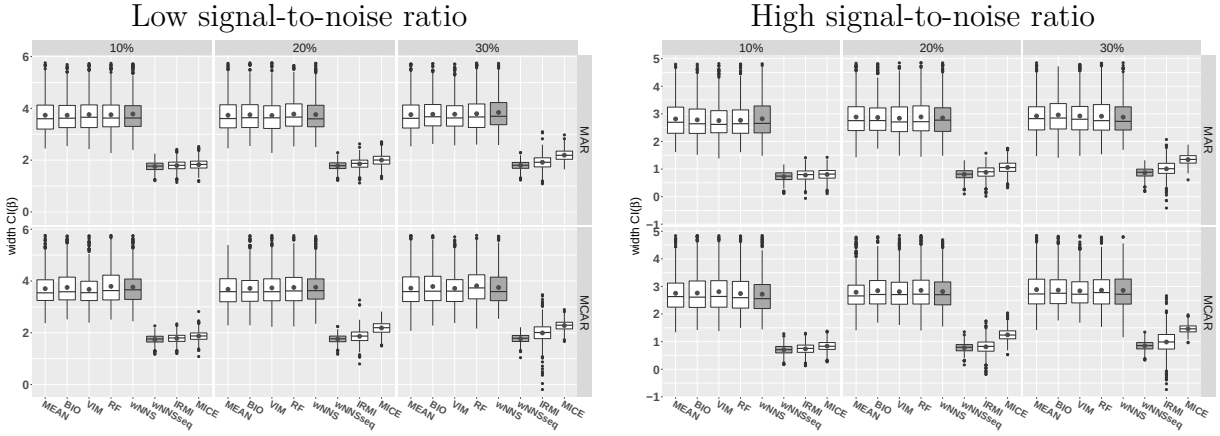


Figure 9.3.: Comparison of the widths of 95% CIs for $\hat{\boldsymbol{\beta}}$ (on log scale) for different multiple imputation methods using $n = 50$ and $p = 30$. Proposed algorithms are shown in gray boxes.

of confidence intervals and the coverage rate for $\hat{\boldsymbol{\beta}}$. Figure 9.3 shows the widths of 95% confidence intervals $n = 50$, and $p = 30$. It is seen that there are two groups of methods, the confidence intervals obtained by imputation methods based on bootstrap and the group formed by `miNNseq`, `IRMI` and `MICE`. The former group has much wider confidence interval. In the latter group the widths are rather homogeneous with `MICE` showing a tendency to wider intervals. The `miNNseq` method yields the smallest widths followed by `IRMI` and `MICE`.

The coverage probabilities of individual predictors are shown in Figure 9.4 for low and high SNR using $n = 100$. For low SNR data (Figure 9.4: left panel), the bootstrap based methods provide a higher coverage than the nominal level under MCAR and MAR mechanisms. `IRMI` shows lower coverage rate than the nominal rate with an increasing percentage of missing values, while `miNNseq` and `MICE` retain their coverage. With high SNR data (Figure 9.4:

right panel), the coverage of both `MICE` and `IRMI` methods gets worse for 30% missing data. The detailed results of average coverage probabilities of confidence intervals for other settings are given in the Appendix (Table D.2). The results show that all imputation methods that use bootstrapping (`MEAN`, `BIO`, `VIM`, `RF` and `miNNboot`), attain a very high coverage rate. For $n = 50$, the coverage rate of these methods is even 1 for both low and high SNR data, and for $n = 100$, the coverage rate is higher than the nominal level. The coverage rate for`IRMI` is much lower than the nominal level of 0.95, whereas `MICE` provides better results. Among the existing methods the `miNNseq` method seems to work best with coverage rate close to the nominal value.

Effect of Degrees of Freedom Approximation

The confidence intervals computed to compare the coverage rates in the previous subsection were constructed by using the normal distribution approximation. Rubin and Schenker (1986) proposed using a t-distribution and Lipsitz et al. (2002) provided an alternative approximation of the degrees of freedom for t-distribution. We compare the effect of all these methods of approximating the degrees of freedom on the coverage rate of the confidence intervals of regression coefficients. For the same data of low and high SNR, $n = 50,\ 100$ and $M = 5$, we compare the resulting coverage probabilities. The confidence intervals are constructed by using the normal (Normal), and the t-distribution using Rubin's (tRubin), Lipsitz (tLip) and Lipsitz with single degrees of freedom (tLipS). The overall coverage, by taking mean of coverage across all the predictors, for each method is shown in Figure 9.5 for the data with high SNR and low SNR. It is seen that there is not much difference in coverage obtained by the four methods. `MEAN` and `IRMI` show poor coverage compared to other methods. All other methods based on bootstrap (`BIO`, `VIM`, `RF`, `miNNboot`) show a higher coverage than the desired rate which is 0.95. The only two methods which achieve or are close to the nominal level are `miNNseq` and `MICE`. The detailed results are given in the Appendix (Table D.4 and Table D.5) for low and high SNR data. The coverage results for the considered methods of the df approximation are rather similar.

9.4.2. Simulation Study for High-Dimensional Data ($n < p$)

In specific fields one has to deal with cases in which the number of predictors is larger than the sample size. To evaluate the performance of the imputation methods in such cases, we consider the simulation settings used by Deng et al. (2016). We select $S = 100$ samples of size $n = 100$ and $p = 200$. We set $\beta_1 = \cdots = \beta_5 = 1$ and $\beta_6 = \cdots = \beta_{200} = 0$. The rest of the procedure is the same as in the previous simulations.

The average MSIEs are shown in Figure 9.6 (upper panel). It is obvious that the `miNNseq` method obtains the smallest MSIEs in all the settings. The average MSE($\hat{\boldsymbol{\beta}}$) obtained by lasso regression are shown in the lower panel of Figure 9.6. When the missing mechanism is MCAR, the performance of the methods using bootstrap (`MEAN`, `BIO`, `VIM`, `RF`, `miNNboot`) is similar to each other but poor as compared to others (`miNNseq`, `IRMI`, `MICE`). For 10%

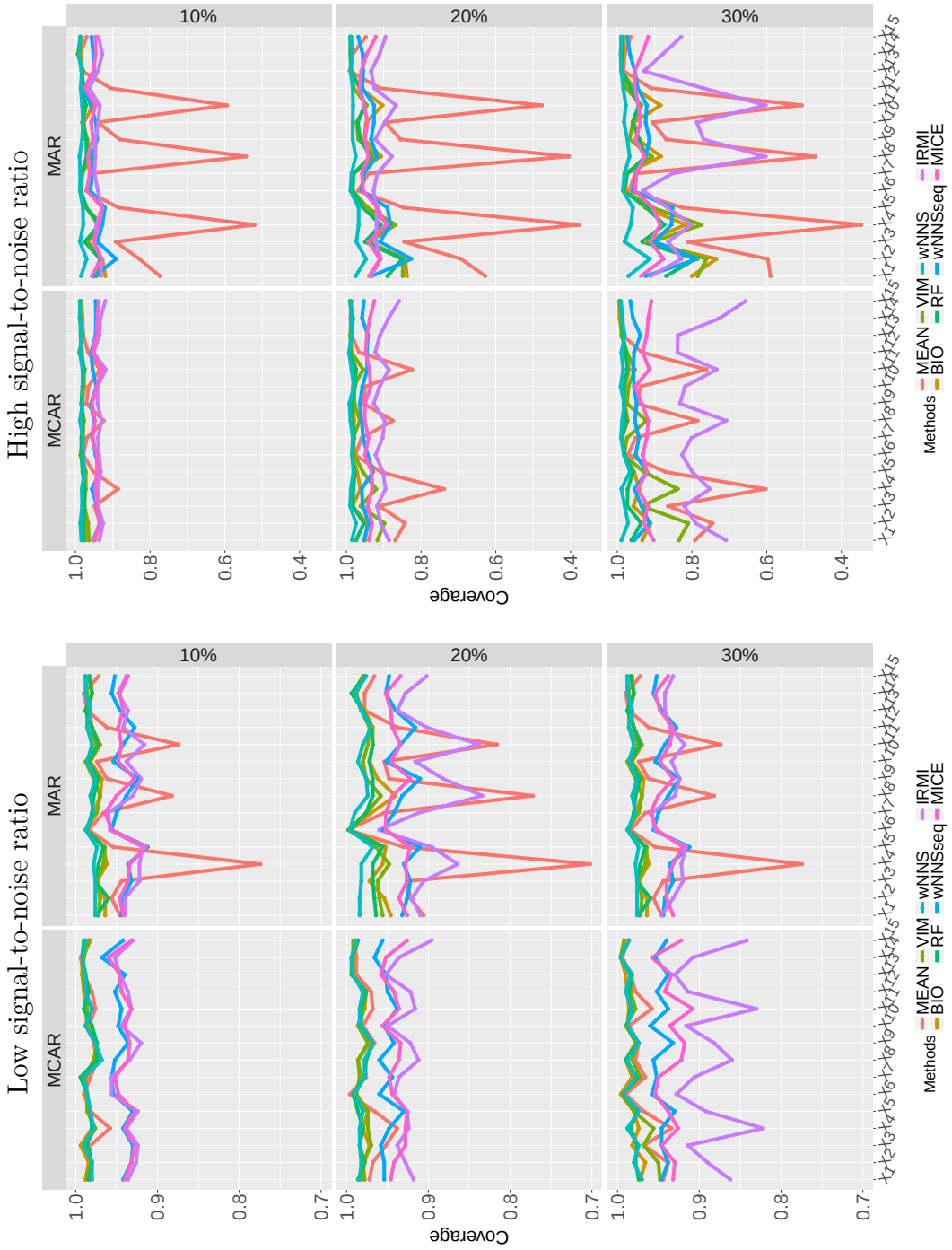


Figure 9.4.: Coverage probabilities for 95% confidence interval for $\hat{\boldsymbol{\beta}}$'s using $n = 100$ and $p = 15$

.

Figure 9.5.: Coverage probabilities for 95% confidence interval for $\hat{\beta}$'s using different methods of approximating degrees of freedom. The horizontal red line shows the nominal level of coverage.

missing data under MCAR, `miNNseq`, `IRMI` and `MICE` yield results similar to complete data. In contrast, for MAR data, the smallest values are obtained by the `miNNseq` method for 10, 20 and 30% missing values.

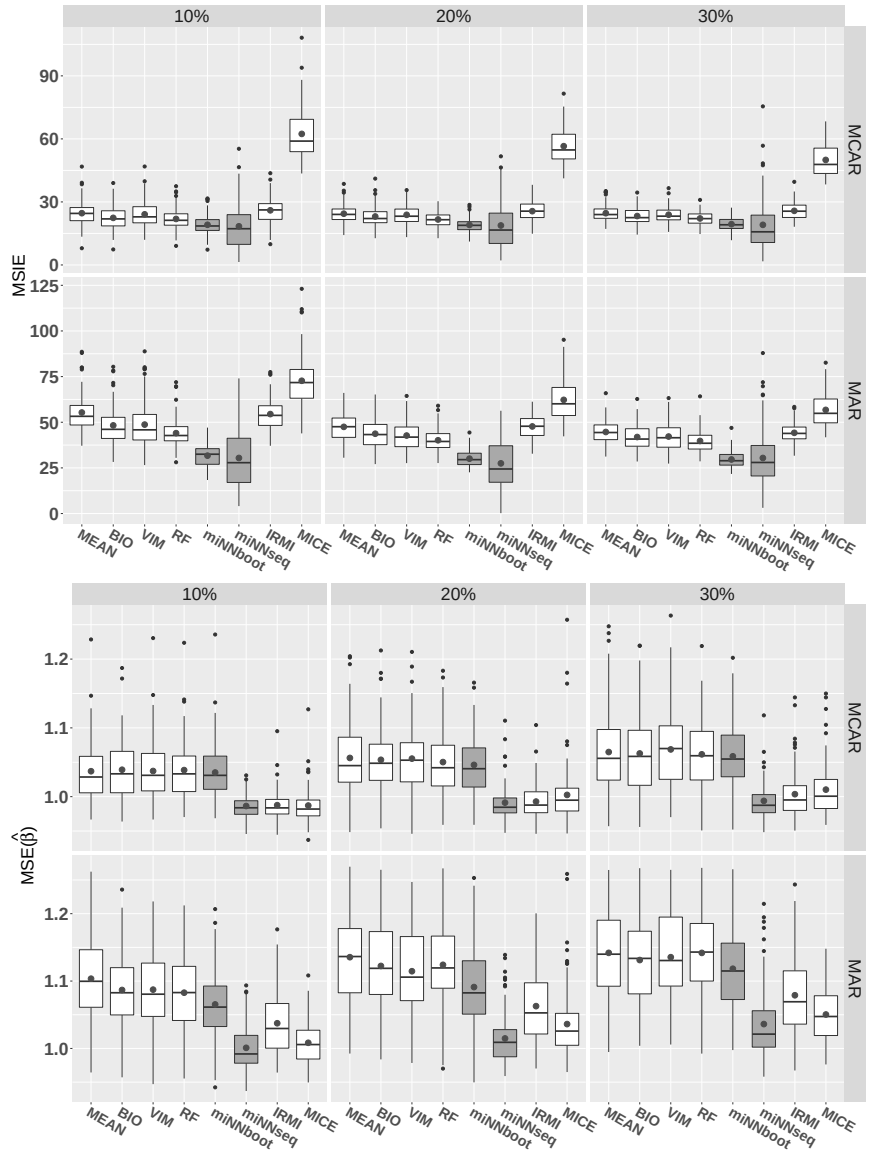


Figure 9.6.: Simulation ($n < p$): MSIE and MSE($\hat{\boldsymbol{\beta}}_{lasso}$) for different multiple imputation methods for $n = 100$ and $p = 200$

9.5. Real Data

In this section, we use two real data sets to compare the performance of imputation methods. One data has $n > p$ and the other $n < p$, both datasets are complete without any missing

values. We are using the complete data intentionally, so as to allow a comparison of the results. In the following, we briefly describe the data sets.

SpectF Data

This data set contains images for the diagnosis of cardiac Single Proton Emission Computed Tomography (SPECT). The response is binary, classifying the patients as normal or abnormal. The original data set contains SPECT images of 267 patients, which was processed to extract important features, resulting in 44 metric variables for each patient. The data can be accessed freely on the UCI Machine Learning Repository (Lichman, 2013). The data set contains two different files: training and test data. We use the training data with $n = 80$, $p = 44$ continuous predictors, and a binary response.

LSVT Voice Rehabilitation Data Set

Lee Silverman Voice Treatment (LSVT) Global, a company specializing in voice rehabilitation, assists people with Parkinson's disease (PD). The data set was originally collected to determine the most parsimonious feature subset that aids in the prediction of the binary response. The data are composed of a range of biomedical speech signal- processing algorithms from 14 people diagnosed with PD and undergoing LSVT. The original study used 310 algorithms (predictors) to characterize 126 speech signals (samples). The response variable is binary, acceptable vs. unacceptable phonation during rehabilitation. More information on the data can be found in Tsanas et al. (2014). The data is publicly available at the UCI Machine Learning Repository (Lichman, 2013). The data we use contains $n = 126$, $p = 309$ continuous predictors and a binary response variable.

9.5.1. Results for Imputation Error

Both data sets are standardized to zero mean and unit variance because some imputation methods are based on computing distances that require the data to be standardized. Then missing values are introduced artificially in half of the total predictors, selected randomly, under MCAR and MAR mechanism with miss = 10%, 20%, 30%. For all the considered MI methods, $M = 5$ imputed datasets are obtained. The whole process is repeated $S = 100$ times. The results of MSIEs for both data are given in Table 9.3 and the smallest value in each setting is shown in boldface. It is obvious that the `miNNseq` approach provides the smallest MSIEs in most of the settings. Sometimes the `miNNboot` method yields the smallest MSIE (e.g., for 30% missing in SpectF data and 20% missing in LVST data under MCAR and MAR) but these are closer to that of yielded by the `miNNseq` method.

Table 9.3.: Averages MSIEs for real data sets

Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
		SpectF data							
MCAR	10%	1.0361	0.8869	0.9376	0.7126	0.5948	**0.4796**	0.9592	1.1559
	20%	1.0402	0.8992	0.9620	0.7274	0.6344	**0.5109**	0.8972	1.2323
	30%	1.0242	0.8942	0.9732	0.7317	**0.6406**	0.6580	0.9405	1.4253
MAR	10%	0.9254	0.7946	0.8272	0.6330	0.5569	**0.4312**	0.8643	1.0949
	20%	0.9708	0.8546	0.9093	0.6847	0.6136	**0.5586**	0.8263	1.2043
	30%	0.9971	0.8825	0.9561	0.7242	**0.6416**	0.7217	0.8914	1.4360
		LVST data							
MCAR	10%	1.0416	0.9077	0.8609	0.6926	0.6477	**0.3856**	2.0947	5.4180
	20%	1.0081	0.9105	0.8630	0.6528	**0.6208**	0.6381	2.1842	4.9584
	30%	1.0964	1.0175	1.0045	0.7320	0.6834	**0.5366**	2.3440	4.0308
MAR	10%	1.1366	0.9874	0.9545	0.7534	0.7088	**0.6917**	2.2415	5.5383
	20%	1.0854	0.9778	0.9501	0.6965	**0.6411**	0.8723	2.2829	5.0767
	30%	1.0969	1.0018	0.9822	0.7365	0.6834	**0.6416**	2.4269	4.0690

9.5.2. Simulation Using Real Data

In this section we conduct a simulation study to compute the MSE($\hat{\boldsymbol{\beta}}$) and coverage of confidence intervals. As the true values of $\boldsymbol{\beta}$ are unknown for the real datasets, it is not possible the compute the true mean squared errors MSE($\hat{\boldsymbol{\beta}}$) and the true coverage of confidence intervals for the real datasets. Therefore, we generate simulated data using the information of the complete real data. More specifically, we generate the binary response variable using the data matrix $\boldsymbol{X}$. First we fit a regression model to the complete data to obtain the parameter estimates ($\hat{\boldsymbol{\beta}}_{complete}$). Then we generate new outcomes using a logistic model with logit

$$\boldsymbol{\eta} = \beta_0 + \boldsymbol{X}\hat{\boldsymbol{\beta}}_{complete}$$

The response y is drawn from the binomial distribution. In this manner, the value of $\hat{\boldsymbol{\beta}}_{complete}$ is the true value of the parameters for the simulated data.

To compare the coverage of confidence intervals, we proceed with SpectF data only, as the LVST data has too many predictors ($p \gg n$). Ridge regression is used to compute the parameter estimates for the SpectF data as the `glm` method did not converge due to high correlations among the predictors. An alternative to ridge regression is lasso (least absolute shrinkage and selection operator), but the standard errors of the estimated parameters are not available. Therefore, we apply ridge regression using the `R` package `penalized` (Goeman et al., 2017). The value of the penalty was selected using a 10-fold cross validation. The standard errors for the estimated parameters are obtained by the method given by Cule et al. (2011). The coverage of the confidence intervals is computed using the t-distribution,

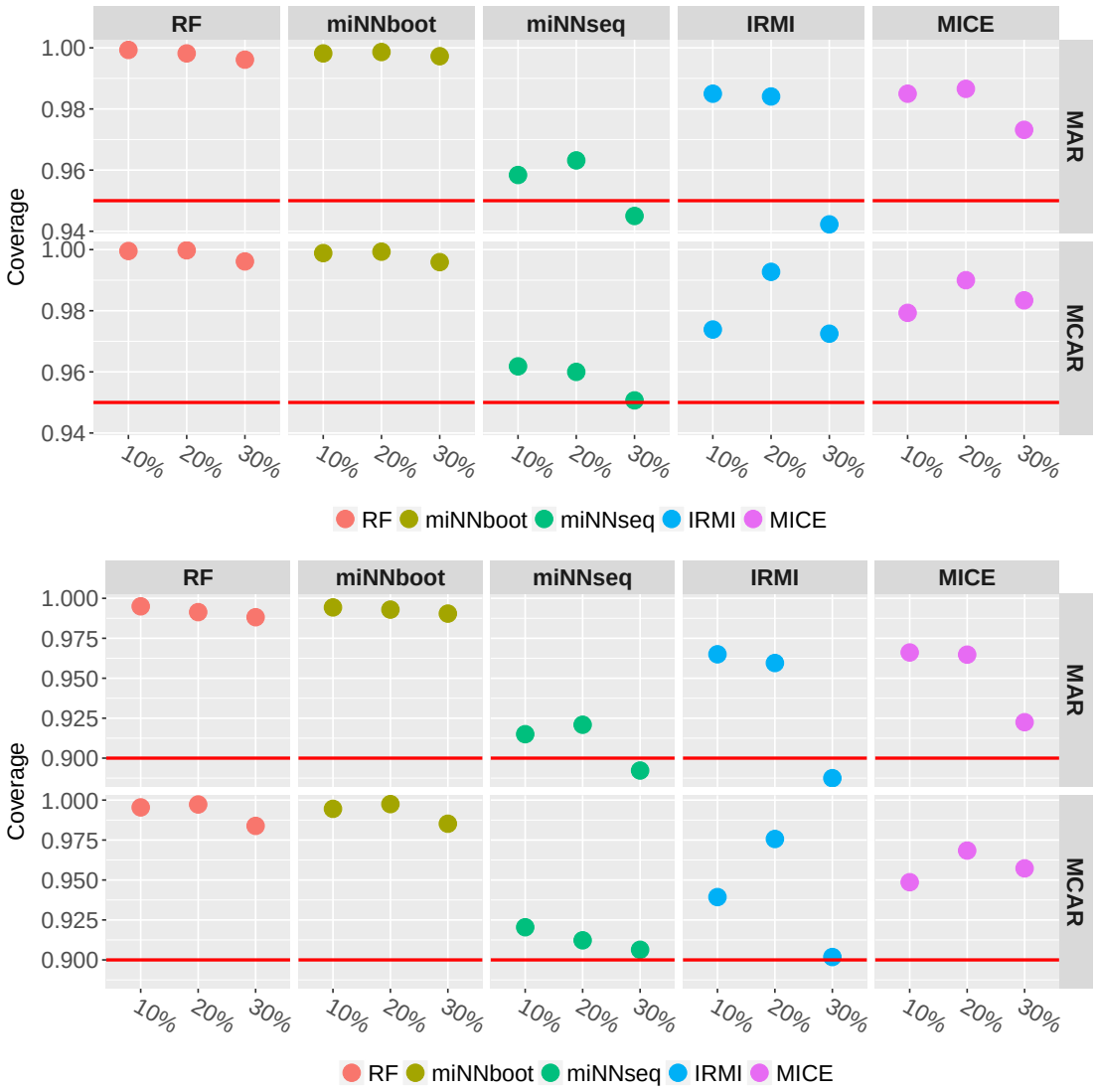


Figure 9.7.: SpectF data: average coverage probabilities for 95% (upper) and 90% (lower) confidence intervals. The horizontal red line shows the nominal level.

and Rubin's approximation for the degrees of freedom (df_{Rubin}), as the simulation results in the previous section showed no significant difference for other approximations of degrees of freedom.

For the presentation of the results, we pick only the competing methods that showed good performance in the simulations. The average coverage probabilities for 10%, 20% and 30% missing data under MCAR and MAR mechanism are shown in Figure 9.7. The upper panel and lower panels show the coverage rates for 95% and 90% confidence level respectively. The bootstrap based methods, `RF` and `miNNboot`, show a 100% coverage which is consistent with the results obtained in the simulation studies. The `MICE` method always yields a higher coverage than the nominal rate, in all the setting of 90% and 95% CIs. The coverage of the `IRMI` method decreases as the missing percentage increases. For 30% missing under

Table 9.4.: Simulated SpectF data: MSE of the estimated coefficient obtained by ridge regression

Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
					MSE($\hat{\boldsymbol{\beta}}$)				
MCAR	10%	0.1599	0.1545	0.1625	0.1622	0.1504	**0.0630**	0.0784	0.0706
	20%	0.1093	0.1077	0.1048	0.1098	0.1045	**0.0551**	0.0854	0.0862
	30%	0.1216	0.1191	0.1222	0.1299	0.1246	0.0439	**0.0434**	0.0841
MAR	10%	0.1373	0.1408	0.1391	0.1407	0.1445	**0.0523**	0.1026	0.0976
	20%	0.1176	0.1265	0.1164	0.1202	0.1335	**0.0541**	0.1468	0.1639
	30%	0.1673	0.1635	0.1690	0.1705	0.1701	**0.0509**	0.0618	0.1049
					Var($\hat{\boldsymbol{\beta}}$)				
MCAR	10%	0.0576	0.0562	0.0580	0.0585	0.0548	0.0308	0.0029	**0.0021**
	20%	0.0437	0.0440	0.0425	0.0442	0.0436	0.0254	**0.0062**	0.0072
	30%	0.0449	0.0440	0.0443	0.0476	0.0456	0.0190	**0.0008**	0.0054
MAR	10%	0.0497	0.0505	0.0496	0.0511	0.0528	0.0242	**0.0036**	**0.0036**
	20%	0.0421	0.0433	0.0411	0.0425	0.0454	0.0262	**0.0050**	0.0088
	30%	0.0541	0.0549	0.0563	0.0555	0.0557	0.0211	**0.0010**	0.0033
					Bias^2($\hat{\boldsymbol{\beta}}$)				
MCAR	10%	0.1023	0.0984	0.1045	0.1037	0.0956	**0.0322**	0.0754	0.0685
	20%	0.0656	0.0637	0.0622	0.0655	0.0609	**0.0297**	0.0792	0.0790
	30%	0.0767	0.0750	0.0780	0.0823	0.0790	**0.0249**	0.0425	0.0787
MAR	10%	0.0877	0.0904	0.0895	0.0896	0.0916	**0.0281**	0.0990	0.0939
	20%	0.0755	0.0832	0.0753	0.0776	0.0880	**0.0279**	0.1417	0.1551
	30%	0.1132	0.1086	0.1127	0.1150	0.1144	**0.0298**	0.0608	0.1017

MCAR and MAR mechanisms, the `miNNseq` method attains the desired coverage. Overall, the `miNNseq` method seems to provide the best performance in terms of coverage.

The average results of MSE($\hat{\boldsymbol{\beta}}$) and its breakdown into variance and bias for the simulated SpectF data are given in Table 9.4. It can be seen that the smallest MSE($\hat{\boldsymbol{\beta}}$) are obtained by `miNNseq` method except for 30% MCAR data, where it is very close to smallest MSIE. It is noteworthy that this method yields estimates that are less biased as compared to other methods. Thus `miNNseq` method yields estimates with smaller MSE($\hat{\boldsymbol{\beta}}$) as well as bias.

9.6. Concluding Remarks

Missing values are a major problem in many areas of quantitative research. The complete case analysis discards a lot of useful information, that may result in biased parameter estimates and inferences may suffer. Multiple imputation has been a useful strategy to handle

missing data problems. The existing methods for multiple imputation do not perform at all when $p > n$, or perform poorly when $p < n$ but large as compared to the samples size. Some techniques have been suggested in the literature that combine variable selection with multiple imputation that are based on strict assumptions.

We present multiple imputation methods based on nearest neighbors. The specific distances are computed using the information of correlation among the target and candidate predictors. Thus only the relevant predictors contribute to the computation of distances. The method successfully imputes missing values in high-dimensional settings, and does not require any distributional assumptions. Using a variety of simulated data with MCAR and MAR missing patterns, the proposed algorithm is compared to two existing methods, Multivariate Imputation by Chained Equations (MICE), and Iterative robust model-based imputation (IRMI). Our simulation studies show that the sequential imputation using weighted nearest neighbors (`miNNseq`) can be applied to a wide range of data settings. The proposed method outperforms or is close to the best when compared to existing methods. Two real data sets are also used that confirm the better performance of the proposed method compared to the existing methods for multiple imputation.

10. Conclusion and Outlook

The technical development of recent decades has significantly increased the availability of large volumes of data. One of the big problems in data analysis is the occurrence of missing values. The problem is particularly difficult to work with if the variables have different scale levels (metric, nominal or ordinal). Whatever the reason for the lack of values, the problem occurs in all areas of applied research. The focus of this thesis is on the imputation of missing values in low and high-dimensional data settings where the covariates may come from different distributions. In particular, improved nearest neighbors procedures and L_q-distance is used as basis and extended in various ways.

Nearest-neighbor (NN) procedures have been used successfully in classification problems, in clustering, and in imputation of missing values. We proposed improved procedures that lead to much better imputations. In particular, the developed localization technique, which relies on a weighting of nearest neighbors, has proven to be very efficient. In the high-dimensional case, a selection of the variables is used that explicitly takes into account the correlation of the variables. This clearly distinguishes the process from previously available methods. Both simulation studies and the analysis of real data sets show that the method achieves good imputation values. Beyond the classical case of metric data, methods are developed for binary and multi-category, as well as the case of different data type. It is shown that even in cases where the number of observations is smaller than the number of variables, the methods can be used efficiently and achieve better results than previously available alternatives.

Although the methods improve traditional NN-based imputation and the popular random forests imputation methods, they share the drawbacks of all single imputation methods. These include the fact that the valid inference obtained using some of the multiple imputation methods is not available, since the single imputation methods do not account the uncertainty of imputation associated with the missing values. Multiple imputation is an attractive procedure that allows the uncertainty generated by the imputation to be explicitly considered in the inference. But the existing methods for multiple imputation do not perform at all when $p > n$, or perform poorly when $p < n$ but large as compared to the samples size. We proposed two different methods of multiple imputation, the results achieved are better or at least comparable to the results expected from the best existing methods.

This thesis is a first step in multiple imputation of high-dimensional data using a non-parametric method. There is a need of developing such MI techniques that can handle missing values in high-dimensional data and time efficient too. The methods developed for

multiple imputation use the metric data only, but there is clearly room for extensions to other types of covariates, such as binary, multi-categorical and mixed type data. The NN methods developed in this thesis may also be extended for missing values in longitudinal studies and time-to-event data. Some further challenges still exist, for example, imputation in hierarchical models and latent trait models. One can find specific distance measure suitable for hierarchical models and latent trait model. We are planning to work in the above mentioned directions in future.

In conclusion, this thesis proposes a variety of improved methods to existing nearest neighbors imputation for metric, multi-categorical and mixed type data. It is demonstrated that the proposed improved nearest neighbors imputation methods provide a better solution for the imputation of missing values. Furthermore it is demonstrated that these methods perform equally well in low as well as in high-dimensional data settings. Nevertheless, there are many aspects left that require further attention and should be the target of future research. I hope, this thesis can provide a contribution to some unanswered questions within the topic of imputation of missing data.

In the firm believe that Banksy's theory also holds for dissertations:

> *I have a theory that you can make any sentence seem profound by writing the name of a dead philosopher at the end of it.*
>
> *Plato*

Appendices

A. Additional Results for Chapter 5

Unweighted k Nearest Neighbors Imputation

In this section we perform a small simulation study to see if the nearest neighbors imputation is influenced by the number of nearest neighbors (k) and the distance measure used to find these nearest neighbors.

The samples of size $n = 100$, $p = 50$ are randomly drawn from p-variate multivariate normal distribution with a mean of $\mathbf{0}$ and a covariance matrix ρ of AR(1)-type structure. The continuous covariates are converted to categorical attributes with $n_{cat} = 4$ categories. Then $m = 10\%, 20\%, 30\%$ of the total values are deleted artificially to evaluate the imputation procedure. The unweighted kNN imputation method for fixed values of $k = 1, \cdots, 40$ is used to fill the missing values. Figure A.1 shows the average results over $S = 100$ simulations for $m = 10\%$ missing only (results for 20 and 30% are not shown as they are similar to this case).

Figure A.1 reveals that the largest imputation errors are exhibited by the distance based on Pearson contingency coefficient d_{PCC} (second row in Figure A.1). Whereas d_{Cohen} and d_{SMC} perform almost similar with d_{SMC} to be slightly better than d_{Cohen}. The imputation error attains its minimum values when $10 \leq k \leq 20$, for the distances d_{SMC} and d_{Cohen} in the imputation procedure (first and last row in Figure A.1). But for d_{PCC}, the imputation error is still decreasing.

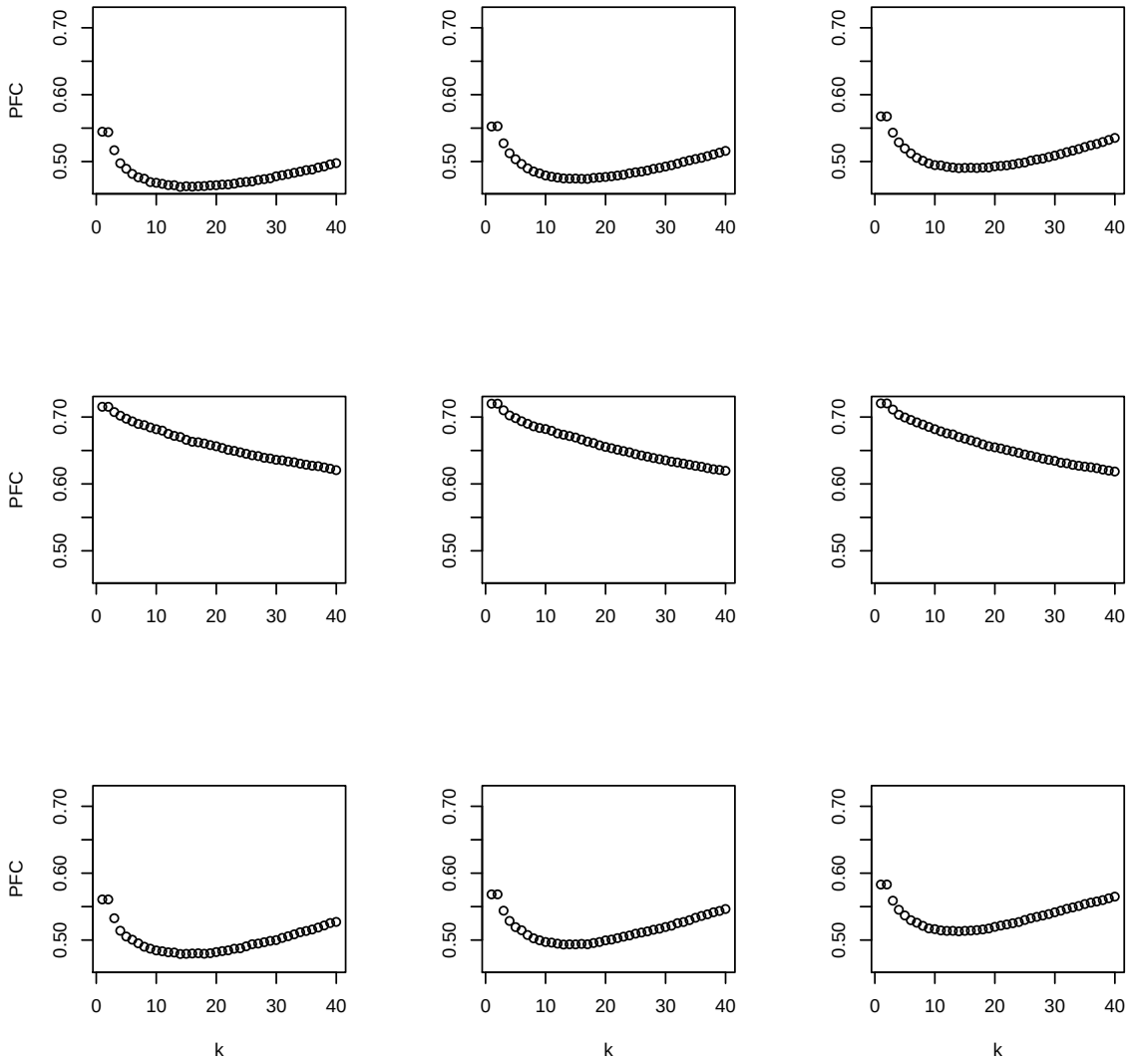


Figure A.1.: Illustration of simulation study: Average PFC (proportion of falsely imputed categories) from 100 simulation samples with $n = 100$, $p = 50$; when d_{SMC} (first row), d_{PCC} (second row) and d_{Cohen} (last row) is employed as distance measure in un-weighted kNN imputation to replace $m = 10\%$ missing values.

B. Appendix for Chapter 7

Table B.1.: Coverage rate for 90% confidence intervals for $\hat{\boldsymbol{\beta}}$

Mechanism	miss	method	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
			$n = 50$														
MCAR	10	comp	0.88	0.87	0.90	0.90	0.86	0.90	0.90	0.88	0.91	0.92	0.92	0.90	0.86	0.89	0.89
		noboot	0.88	0.85	0.88	0.88	0.86	0.88	0.91	0.88	0.92	0.92	0.90	0.90	0.85	0.88	0.89
		boot	0.92	0.90	0.94	0.94	0.90	0.92	0.92	0.94	0.95	0.96	0.96	0.92	0.92	0.91	0.94
	20	comp	0.88	0.87	0.90	0.90	0.86	0.90	0.90	0.88	0.91	0.92	0.92	0.90	0.86	0.89	0.89
		noboot	0.86	0.82	0.88	0.90	0.88	0.88	0.91	0.90	0.90	0.93	0.89	0.88	0.85	0.87	0.90
		boot	0.92	0.93	0.96	0.96	0.93	0.92	0.92	0.95	0.93	0.96	0.94	0.94	0.90	0.89	0.94
	30	comp	0.88	0.87	0.90	0.90	0.86	0.90	0.90	0.88	0.91	0.92	0.92	0.90	0.86	0.89	0.89
		noboot	0.90	0.78	0.86	0.91	0.88	0.88	0.89	0.89	0.91	0.91	0.90	0.88	0.85	0.89	0.90
		boot	0.94	0.88	0.93	0.95	0.95	0.95	0.94	0.95	0.95	0.96	0.94	0.94	0.92	0.91	0.95
MAR	10	comp	0.88	0.87	0.90	0.90	0.86	0.90	0.90	0.88	0.91	0.92	0.92	0.90	0.86	0.89	0.89
		noboot	0.88	0.85	0.90	0.88	0.86	0.90	0.90	0.91	0.91	0.91	0.89	0.90	0.84	0.86	0.89
		boot	0.92	0.91	0.95	0.94	0.93	0.94	0.93	0.94	0.93	0.95	0.94	0.93	0.91	0.90	0.96
	20	comp	0.88	0.87	0.90	0.90	0.86	0.90	0.90	0.88	0.91	0.92	0.92	0.90	0.86	0.89	0.89
		noboot	0.87	0.80	0.89	0.88	0.85	0.87	0.87	0.91	0.94	0.94	0.90	0.84	0.84	0.86	0.91
		boot	0.92	0.90	0.96	0.95	0.94	0.92	0.92	0.96	0.95	0.97	0.94	0.94	0.92	0.91	0.95
	30	comp	0.88	0.87	0.90	0.90	0.86	0.90	0.90	0.88	0.91	0.92	0.92	0.90	0.86	0.89	0.89
		noboot	0.86	0.79	0.84	0.92	0.88	0.90	0.90	0.91	0.93	0.93	0.88	0.86	0.88	0.86	0.90
		boot	0.95	0.88	0.92	0.97	0.96	0.95	0.94	0.94	0.95	0.97	0.92	0.93	0.93	0.92	0.93
			$n = 100$														
MCAR	10	comp	0.86	0.90	0.91	0.92	0.89	0.88	0.92	0.90	0.92	0.90	0.90	0.88	0.88	0.91	0.89
		noboot	0.84	0.90	0.90	0.89	0.91	0.88	0.94	0.90	0.90	0.89	0.88	0.90	0.89	0.92	0.89
		boot	0.86	0.92	0.92	0.91	0.91	0.90	0.93	0.90	0.89	0.88	0.88	0.89	0.90	0.90	0.88
	20	comp	0.86	0.90	0.91	0.92	0.89	0.88	0.92	0.90	0.92	0.90	0.90	0.88	0.88	0.91	0.89
		noboot	0.83	0.84	0.90	0.90	0.90	0.89	0.93	0.90	0.90	0.88	0.88	0.90	0.88	0.92	0.90
		boot	0.88	0.86	0.90	0.94	0.90	0.90	0.94	0.92	0.89	0.83	0.87	0.90	0.89	0.92	0.90
	30	comp	0.86	0.90	0.91	0.92	0.89	0.88	0.92	0.90	0.92	0.90	0.90	0.88	0.88	0.91	0.89
		noboot	0.78	0.73	0.81	0.83	0.92	0.92	0.92	0.92	0.89	0.87	0.87	0.87	0.89	0.90	0.88
		boot	0.80	0.73	0.85	0.88	0.94	0.94	0.94	0.92	0.90	0.86	0.90	0.87	0.89	0.90	0.88
MAR	10	comp	0.86	0.90	0.91	0.92	0.89	0.88	0.92	0.90	0.92	0.90	0.90	0.88	0.88	0.91	0.89
		noboot	0.85	0.87	0.90	0.88	0.90	0.90	0.92	0.90	0.88	0.88	0.90	0.88	0.88	0.88	0.88
		boot	0.88	0.90	0.92	0.90	0.92	0.90	0.93	0.90	0.88	0.89	0.90	0.88	0.90	0.90	0.90
	20	comp	0.86	0.90	0.91	0.92	0.89	0.88	0.92	0.90	0.92	0.90	0.90	0.88	0.88	0.91	0.89
		noboot	0.89	0.83	0.82	0.86	0.89	0.90	0.92	0.92	0.90	0.92	0.89	0.89	0.88	0.90	0.90
		boot	0.91	0.88	0.87	0.90	0.92	0.91	0.94	0.92	0.90	0.89	0.88	0.88	0.88	0.90	0.90
	30	comp	0.86	0.90	0.91	0.92	0.89	0.88	0.92	0.90	0.92	0.90	0.90	0.88	0.88	0.91	0.89
		noboot	0.84	0.78	0.80	0.88	0.91	0.90	0.90	0.93	0.89	0.88	0.92	0.90	0.90	0.91	0.88
		boot	0.88	0.81	0.85	0.94	0.94	0.92	0.90	0.91	0.90	0.87	0.92	0.89	0.89	0.90	0.88

Table B.2.: Median widths of 95% confidence intervals for $\hat{\boldsymbol{\beta}}$

Mechanism	miss	method	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
			$n = 50$														
MCAR	10	comp	1.51	1.93	1.98	2.15	2.19	2.04	2.16	2.01	1.99	2.11	2.26	2.38	2.12	1.98	1.46
		noboot	1.57	1.96	2.00	2.16	2.21	2.05	2.22	2.07	2.05	2.12	2.27	2.38	2.15	1.99	1.48
		boot	1.87	2.33	2.43	2.52	2.68	2.43	2.67	2.43	2.41	2.43	2.80	2.84	2.51	2.31	1.80
	20	comp	1.51	1.93	1.98	2.15	2.19	2.04	2.16	2.01	1.99	2.11	2.26	2.38	2.12	1.98	1.46
		noboot	1.60	1.99	2.05	2.20	2.28	2.06	2.23	2.09	2.08	2.11	2.27	2.46	2.16	2.02	1.52
		boot	1.98	2.41	2.47	2.64	2.72	2.46	2.71	2.49	2.52	2.56	2.79	2.87	2.54	2.38	1.82
	30	comp	1.51	1.93	1.98	2.15	2.19	2.04	2.16	2.01	1.99	2.11	2.26	2.38	2.12	1.98	1.46
		noboot	1.56	1.86	2.00	2.14	2.30	2.11	2.31	2.15	2.13	2.14	2.31	2.47	2.16	2.04	1.54
		boot	1.90	2.34	2.54	2.72	2.73	2.56	2.79	2.54	2.54	2.55	2.79	2.98	2.61	2.40	1.87
MAR	10	comp	1.51	1.93	1.98	2.15	2.19	2.04	2.16	2.01	1.99	2.11	2.26	2.38	2.12	1.98	1.46
		noboot	1.59	1.97	1.95	2.15	2.14	2.02	2.19	2.02	2.05	2.09	2.25	2.38	2.15	2.01	1.47
		boot	1.91	2.33	2.35	2.62	2.61	2.40	2.72	2.50	2.44	2.46	2.71	2.87	2.52	2.31	1.79
	20	comp	1.51	1.93	1.98	2.15	2.19	2.04	2.16	2.01	1.99	2.11	2.26	2.38	2.12	1.98	1.46
		noboot	1.63	2.00	2.08	2.25	2.21	2.07	2.27	2.11	2.10	2.16	2.30	2.45	2.21	2.06	1.54
		boot	2.00	2.47	2.55	2.80	2.70	2.47	2.76	2.58	2.53	2.57	2.75	2.96	2.64	2.40	1.86
	30	comp	1.51	1.93	1.98	2.15	2.19	2.04	2.16	2.01	1.99	2.11	2.26	2.38	2.12	1.98	1.46
		noboot	1.69	2.04	2.13	2.22	2.25	2.05	2.28	2.11	2.12	2.12	2.31	2.46	2.19	2.02	1.54
		boot	2.03	2.59	2.77	2.85	2.77	2.46	2.80	2.66	2.56	2.55	2.84	3.02	2.66	2.45	1.85
			$n = 100$														
MCAR	10	comp	0.97	1.37	1.40	1.34	1.27	1.38	1.36	1.29	1.32	1.38	1.43	1.40	1.34	1.38	1.00
		noboot	1.00	1.39	1.43	1.38	1.33	1.40	1.38	1.30	1.32	1.39	1.46	1.43	1.36	1.41	1.01
		boot	1.02	1.41	1.44	1.41	1.39	1.40	1.39	1.32	1.34	1.39	1.50	1.45	1.39	1.44	1.03
	20	comp	0.97	1.37	1.40	1.34	1.27	1.38	1.36	1.29	1.32	1.38	1.43	1.40	1.34	1.38	1.00
		noboot	1.00	1.33	1.40	1.42	1.38	1.42	1.40	1.33	1.34	1.41	1.47	1.45	1.38	1.42	1.03
		boot	1.04	1.37	1.44	1.47	1.41	1.43	1.46	1.34	1.36	1.38	1.50	1.45	1.39	1.46	1.04
	30	comp	0.97	1.37	1.40	1.34	1.27	1.38	1.36	1.29	1.32	1.38	1.43	1.40	1.34	1.38	1.00
		noboot	0.98	1.26	1.34	1.39	1.42	1.42	1.41	1.34	1.36	1.43	1.50	1.46	1.39	1.43	1.04
		boot	1.02	1.32	1.43	1.47	1.49	1.45	1.45	1.36	1.36	1.43	1.53	1.50	1.43	1.47	1.06
MAR	10	comp	0.97	1.37	1.40	1.34	1.27	1.38	1.36	1.29	1.32	1.38	1.43	1.40	1.34	1.38	1.00
		noboot	0.99	1.34	1.39	1.36	1.31	1.37	1.37	1.30	1.33	1.40	1.48	1.44	1.36	1.40	1.02
		boot	1.04	1.39	1.44	1.43	1.35	1.40	1.41	1.34	1.36	1.40	1.51	1.45	1.39	1.43	1.04
	20	comp	0.97	1.37	1.40	1.34	1.27	1.38	1.36	1.29	1.32	1.38	1.43	1.40	1.34	1.38	1.00
		noboot	1.01	1.32	1.36	1.39	1.35	1.39	1.40	1.33	1.35	1.43	1.49	1.46	1.38	1.42	1.04
		boot	1.10	1.41	1.45	1.48	1.40	1.42	1.45	1.36	1.36	1.44	1.52	1.49	1.43	1.48	1.06
	30	comp	0.97	1.37	1.40	1.34	1.27	1.38	1.36	1.29	1.32	1.38	1.43	1.40	1.34	1.38	1.00
		noboot	1.03	1.30	1.39	1.43	1.38	1.40	1.42	1.35	1.37	1.44	1.50	1.48	1.40	1.45	1.05
		boot	1.09	1.41	1.51	1.54	1.48	1.44	1.46	1.37	1.39	1.44	1.56	1.51	1.42	1.48	1.09

Table B.3.: Median widths of 90% confidence intervals for $\hat{\boldsymbol{\beta}}$

Mechanism	miss	method	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
			$n = 50$														
MCAR	10	comp	1.26	1.62	1.66	1.80	1.84	1.71	1.82	1.68	1.67	1.77	1.89	2.00	1.78	1.66	1.23
		noboot	1.32	1.65	1.68	1.81	1.86	1.72	1.86	1.74	1.72	1.78	1.91	2.00	1.80	1.67	1.24
		boot	1.54	1.93	1.99	2.17	2.19	2.00	2.20	2.06	2.05	2.03	2.29	2.37	2.08	1.96	1.49
	20	comp	1.26	1.62	1.66	1.80	1.84	1.71	1.82	1.68	1.67	1.77	1.89	2.00	1.78	1.66	1.23
		noboot	1.35	1.67	1.72	1.85	1.91	1.73	1.87	1.76	1.74	1.77	1.91	2.06	1.81	1.69	1.28
		boot	1.63	2.03	2.06	2.17	2.27	2.04	2.23	2.05	2.07	2.11	2.27	2.37	2.10	1.98	1.47
	30	comp	1.26	1.62	1.66	1.80	1.84	1.71	1.82	1.68	1.67	1.77	1.89	2.00	1.78	1.66	1.23
		noboot	1.31	1.56	1.68	1.80	1.93	1.77	1.94	1.80	1.79	1.80	1.94	2.07	1.81	1.71	1.30
		boot	1.60	1.94	2.09	2.22	2.25	2.08	2.31	2.09	2.12	2.12	2.29	2.43	2.14	1.98	1.56
MAR	10	comp	1.26	1.62	1.66	1.80	1.84	1.71	1.82	1.68	1.67	1.77	1.89	2.00	1.78	1.66	1.23
		noboot	1.34	1.66	1.63	1.80	1.80	1.69	1.84	1.70	1.72	1.76	1.89	2.00	1.80	1.68	1.24
		boot	1.60	1.94	1.97	2.17	2.15	1.94	2.22	2.06	2.01	2.06	2.26	2.40	2.11	1.89	1.48
	20	comp	1.26	1.62	1.66	1.80	1.84	1.71	1.82	1.68	1.67	1.77	1.89	2.00	1.78	1.66	1.23
		noboot	1.37	1.68	1.74	1.89	1.85	1.74	1.91	1.77	1.76	1.81	1.93	2.05	1.85	1.73	1.29
		boot	1.64	2.03	2.13	2.30	2.24	2.06	2.31	2.16	2.10	2.14	2.29	2.46	2.21	1.99	1.54
	30	comp	1.26	1.62	1.66	1.80	1.84	1.71	1.82	1.68	1.67	1.77	1.89	2.00	1.78	1.66	1.23
		noboot	1.41	1.71	1.79	1.86	1.89	1.72	1.92	1.77	1.78	1.78	1.94	2.07	1.84	1.70	1.29
		boot	1.69	2.15	2.27	2.34	2.30	2.05	2.29	2.19	2.14	2.09	2.35	2.49	2.21	2.02	1.56
			$n = 100$														
MCAR	10	comp	0.82	1.15	1.17	1.12	1.07	1.16	1.14	1.08	1.11	1.15	1.20	1.18	1.13	1.16	0.84
		noboot	0.84	1.17	1.20	1.16	1.11	1.17	1.16	1.09	1.11	1.17	1.22	1.20	1.14	1.18	0.85
		boot	0.86	1.19	1.21	1.19	1.16	1.17	1.17	1.11	1.13	1.16	1.24	1.22	1.16	1.20	0.86
	20	comp	0.82	1.15	1.17	1.12	1.07	1.16	1.14	1.08	1.11	1.15	1.20	1.18	1.13	1.16	0.84
		noboot	0.84	1.12	1.18	1.19	1.16	1.19	1.18	1.12	1.12	1.18	1.24	1.22	1.16	1.19	0.86
		boot	0.87	1.17	1.20	1.24	1.19	1.19	1.21	1.14	1.13	1.16	1.25	1.22	1.17	1.22	0.87
	30	comp	0.82	1.15	1.17	1.12	1.07	1.16	1.14	1.08	1.11	1.15	1.20	1.18	1.13	1.16	0.84
		noboot	0.82	1.06	1.13	1.17	1.20	1.19	1.19	1.13	1.14	1.20	1.26	1.22	1.17	1.20	0.87
		boot	0.86	1.11	1.19	1.23	1.27	1.22	1.23	1.16	1.14	1.21	1.29	1.26	1.20	1.24	0.90
MAR	10	comp	0.82	1.15	1.17	1.12	1.07	1.16	1.14	1.08	1.11	1.15	1.20	1.18	1.13	1.16	0.84
		noboot	0.83	1.12	1.17	1.14	1.10	1.15	1.15	1.09	1.11	1.18	1.24	1.21	1.14	1.18	0.86
		boot	0.87	1.18	1.21	1.21	1.14	1.19	1.17	1.12	1.14	1.17	1.26	1.23	1.17	1.20	0.88
	20	comp	0.82	1.15	1.17	1.12	1.07	1.16	1.14	1.08	1.11	1.15	1.20	1.18	1.13	1.16	0.84
		noboot	0.85	1.11	1.15	1.16	1.14	1.17	1.18	1.12	1.14	1.20	1.25	1.23	1.16	1.19	0.87
		boot	0.91	1.21	1.25	1.22	1.18	1.17	1.21	1.14	1.17	1.19	1.27	1.27	1.19	1.26	0.88
	30	comp	0.82	1.15	1.17	1.12	1.07	1.16	1.14	1.08	1.11	1.15	1.20	1.18	1.13	1.16	0.84
		noboot	0.87	1.09	1.16	1.20	1.16	1.18	1.19	1.14	1.15	1.21	1.26	1.25	1.18	1.22	0.88
		boot	0.91	1.18	1.27	1.29	1.22	1.21	1.23	1.18	1.17	1.22	1.30	1.27	1.20	1.25	0.91

Table B.4.: Standard errors for $\hat{\boldsymbol{\beta}}$

Mechanism	miss	method	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}
								$n = 50$									
MCAR	10	comp	0.1565	0.2645	0.2686	0.3207	0.3164	0.2852	0.3215	0.2895	0.2773	0.3155	0.3529	0.4001	0.2996	0.2624	0.1481
		noboot	0.1668	0.2761	0.2862	0.3307	0.3380	0.2884	0.3305	0.3016	0.2874	0.3215	0.3564	0.4067	0.3063	0.2664	0.1524
		boot	0.1730	0.2821	0.2950	0.3398	0.3440	0.3030	0.3563	0.3224	0.3035	0.3362	0.3776	0.4280	0.3219	0.2743	0.1605
	20	comp	0.1565	0.2645	0.2686	0.3207	0.3164	0.2852	0.3215	0.2895	0.2773	0.3155	0.3529	0.4001	0.2996	0.2624	0.1481
		noboot	0.1695	0.2679	0.2888	0.3310	0.3500	0.2915	0.3372	0.3109	0.3001	0.3238	0.3601	0.4113	0.3066	0.2721	0.1588
		boot	0.1789	0.2758	0.2986	0.3378	0.3550	0.3049	0.3598	0.3331	0.3182	0.3383	0.3782	0.4333	0.3232	0.2799	0.1669
	30	comp	0.1565	0.2645	0.2686	0.3207	0.3164	0.2852	0.3215	0.2895	0.2773	0.3155	0.3529	0.4001	0.2996	0.2624	0.1481
		noboot	0.1683	0.2425	0.2855	0.3247	0.3583	0.3039	0.3551	0.3256	0.3162	0.3361	0.3625	0.4187	0.3161	0.2789	0.1664
		boot	0.1837	0.2589	0.3014	0.3358	0.3610	0.3196	0.3817	0.3488	0.3360	0.3496	0.3827	0.4427	0.3357	0.2883	0.1757
MAR	10	comp	0.1565	0.2645	0.2686	0.3207	0.3164	0.2852	0.3215	0.2895	0.2773	0.3155	0.3529	0.4001	0.2996	0.2624	0.1481
		noboot	0.1705	0.2783	0.2782	0.3152	0.3127	0.2777	0.3286	0.3034	0.2858	0.3165	0.3536	0.4151	0.3075	0.2654	0.1534
		boot	0.1775	0.2874	0.2922	0.3296	0.3200	0.2922	0.3556	0.3299	0.3035	0.3302	0.3743	0.4404	0.3261	0.2749	0.1628
	20	comp	0.1565	0.2645	0.2686	0.3207	0.3164	0.2852	0.3215	0.2895	0.2773	0.3155	0.3529	0.4001	0.2996	0.2624	0.1481
		noboot	0.1813	0.2891	0.3086	0.3398	0.3281	0.2850	0.3482	0.3201	0.2994	0.3291	0.3566	0.4197	0.3201	0.2739	0.1606
		boot	0.1921	0.2984	0.3197	0.3532	0.3369	0.2995	0.3764	0.3493	0.3189	0.3443	0.3767	0.4461	0.3384	0.2855	0.1690
	30	comp	0.1565	0.2645	0.2686	0.3207	0.3164	0.2852	0.3215	0.2895	0.2773	0.3155	0.3529	0.4001	0.2996	0.2624	0.1481
		noboot	0.1987	0.2936	0.3339	0.3594	0.3548	0.2886	0.3617	0.3380	0.3150	0.3347	0.3624	0.4238	0.3237	0.2788	0.1671
		boot	0.2137	0.3092	0.3543	0.3836	0.3636	0.3024	0.3927	0.3696	0.3382	0.3513	0.3849	0.4539	0.3459	0.2924	0.1772
								$n = 100$									
MCAR	10	comp	0.0612	0.1209	0.1265	0.1241	0.1071	0.1240	0.1209	0.1123	0.1146	0.1241	0.1421	0.1306	0.1192	0.1269	0.0655
		noboot	0.0662	0.1271	0.1330	0.1317	0.1166	0.1262	0.1231	0.1148	0.1162	0.1258	0.1446	0.1333	0.1214	0.1293	0.0669
		boot	0.0671	0.1275	0.1334	0.1323	0.1178	0.1279	0.1256	0.1173	0.1180	0.1263	0.1462	0.1345	0.1230	0.1318	0.0685
	20	comp	0.0612	0.1209	0.1265	0.1241	0.1071	0.1240	0.1209	0.1123	0.1146	0.1241	0.1421	0.1306	0.1192	0.1269	0.0655
		noboot	0.0666	0.1190	0.1289	0.1350	0.1281	0.1309	0.1278	0.1190	0.1193	0.1298	0.1495	0.1384	0.1255	0.1337	0.0692
		boot	0.0676	0.1193	0.1287	0.1347	0.1284	0.1321	0.1311	0.1217	0.1213	0.1307	0.1513	0.1400	0.1270	0.1364	0.0706
	30	comp	0.0612	0.1209	0.1265	0.1241	0.1071	0.1240	0.1209	0.1123	0.1146	0.1241	0.1421	0.1306	0.1192	0.1269	0.0655
		noboot	0.0641	0.1063	0.1206	0.1343	0.1378	0.1339	0.1300	0.1209	0.1207	0.1314	0.1513	0.1400	0.1276	0.1354	0.0702
		boot	0.0660	0.1079	0.1212	0.1331	0.1372	0.1356	0.1333	0.1243	0.1232	0.1325	0.1532	0.1422	0.1298	0.1383	0.0721
MAR	10	comp	0.0612	0.1209	0.1265	0.1241	0.1071	0.1240	0.1209	0.1123	0.1146	0.1241	0.1421	0.1306	0.1192	0.1269	0.0655
		noboot	0.0646	0.1175	0.1267	0.1284	0.1119	0.1244	0.1240	0.1154	0.1173	0.1272	0.1467	0.1364	0.1234	0.1310	0.0678
		boot	0.0652	0.1176	0.1266	0.1281	0.1117	0.1258	0.1265	0.1179	0.1191	0.1277	0.1479	0.1377	0.1250	0.1335	0.0694
	20	comp	0.0612	0.1209	0.1265	0.1241	0.1071	0.1240	0.1209	0.1123	0.1146	0.1241	0.1421	0.1306	0.1192	0.1269	0.0655
		noboot	0.0687	0.1159	0.1263	0.1299	0.1203	0.1283	0.1291	0.1196	0.1212	0.1325	0.1518	0.1415	0.1281	0.1352	0.0705
		boot	0.0699	0.1165	0.1278	0.1311	0.1197	0.1288	0.1324	0.1221	0.1227	0.1332	0.1532	0.1429	0.1299	0.1377	0.0721
	30	comp	0.0612	0.1209	0.1265	0.1241	0.1071	0.1240	0.1209	0.1123	0.1146	0.1241	0.1421	0.1306	0.1192	0.1269	0.0655
		noboot	0.0712	0.1137	0.1288	0.1359	0.1295	0.1299	0.1322	0.1231	0.1240	0.1352	0.1554	0.1466	0.1317	0.1390	0.0724
		boot	0.0734	0.1168	0.1315	0.1375	0.1295	0.1302	0.1357	0.1257	0.1260	0.1367	0.1576	0.1484	0.1334	0.1418	0.0743

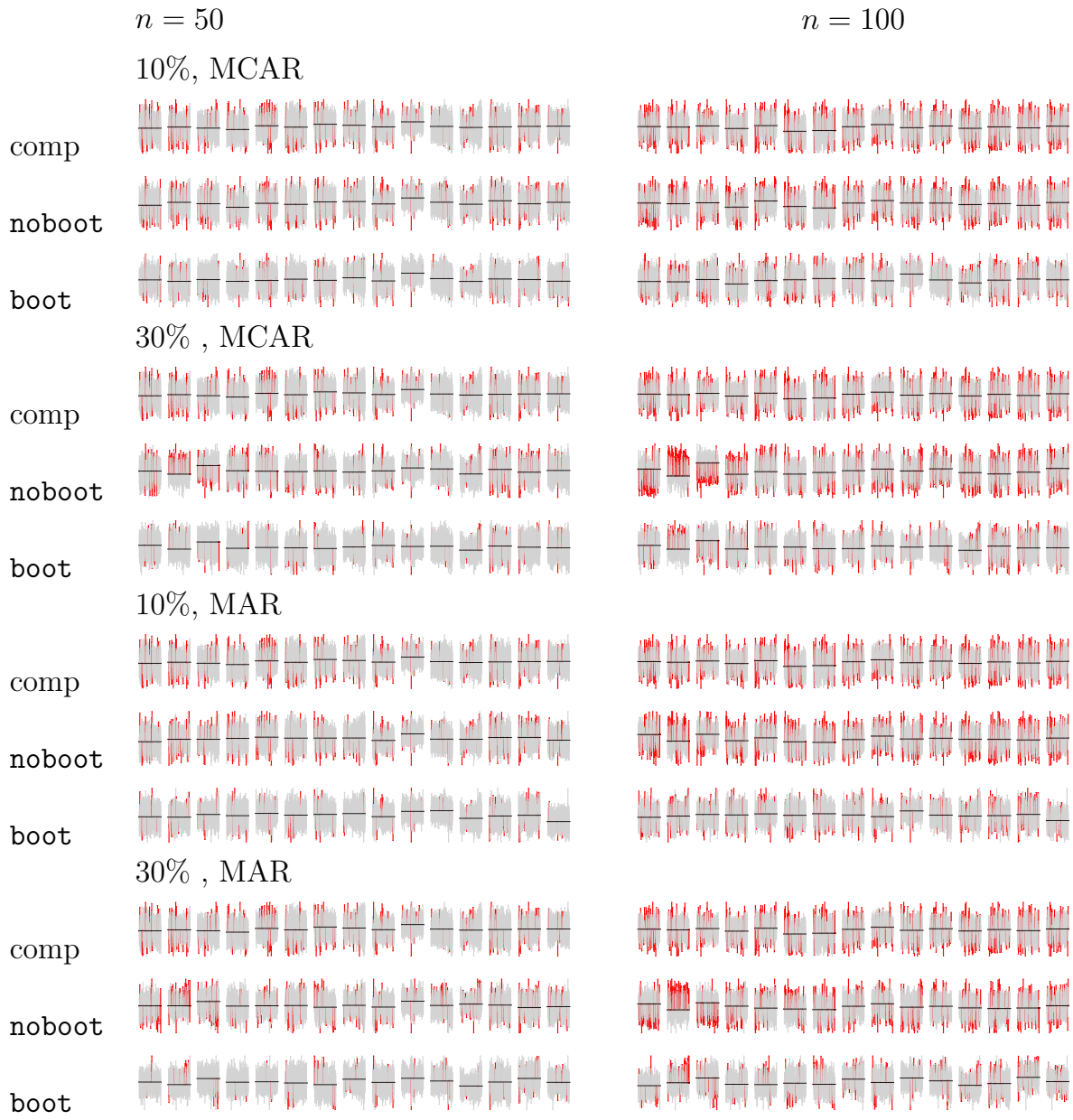


Figure B.1.: Simulation study: 95% CI plots of estimated regression coefficients for $n = 50, 100$ and $p = 15$ with 10% and 30% missing observations using MCAR (upper two panels) and MAR(lower two panels) mechanisms respectively. Likelihood estimates were used to construct CIs. In each plot, the black horizontal line represents the true value of the regression coefficient, the gray lines are for the samples (out of $S = 200$) for which CI covered the true parameter value, and the red lines are for the samples for which CI did not cover the true parameter value.

C. Appendix for Chapter 8

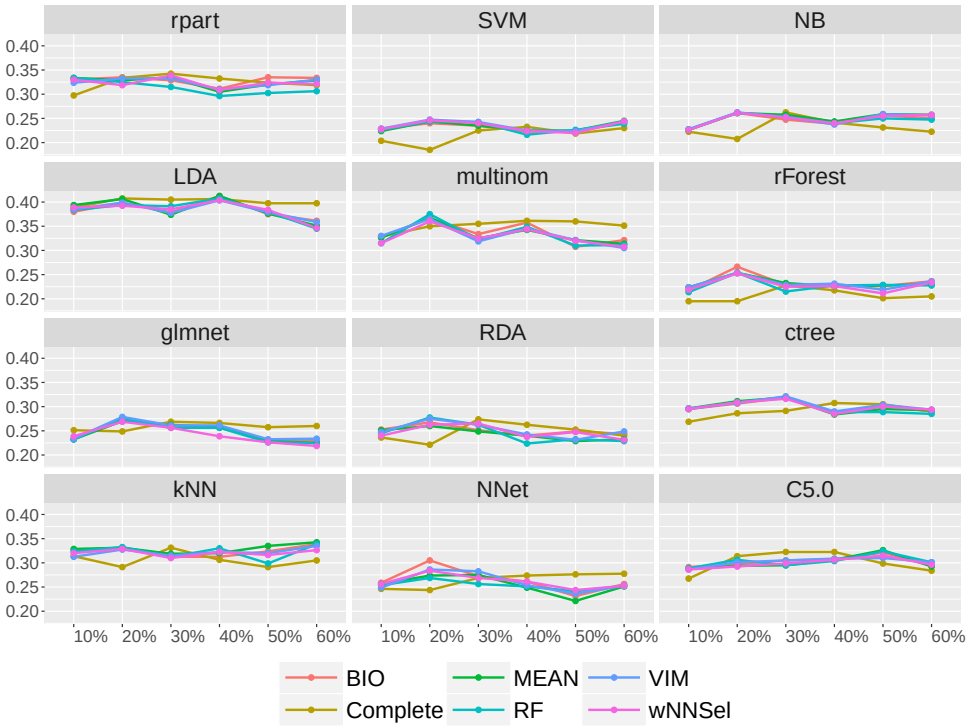


Figure C.1.: SPECT data: average misclassification error

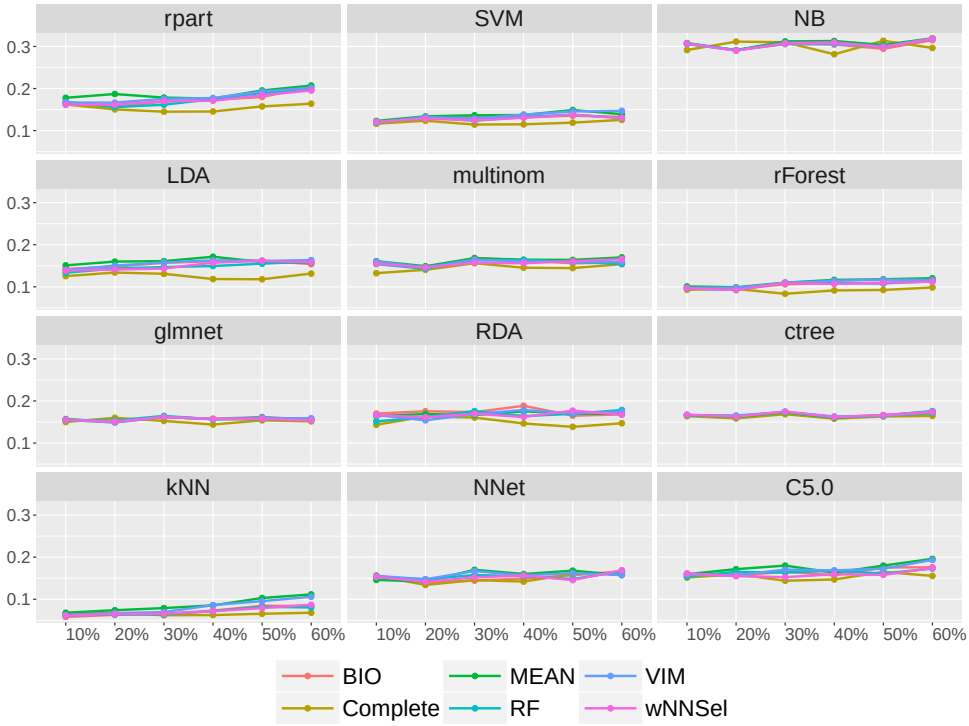


Figure C.2.: Parkinson data: average misclassification error

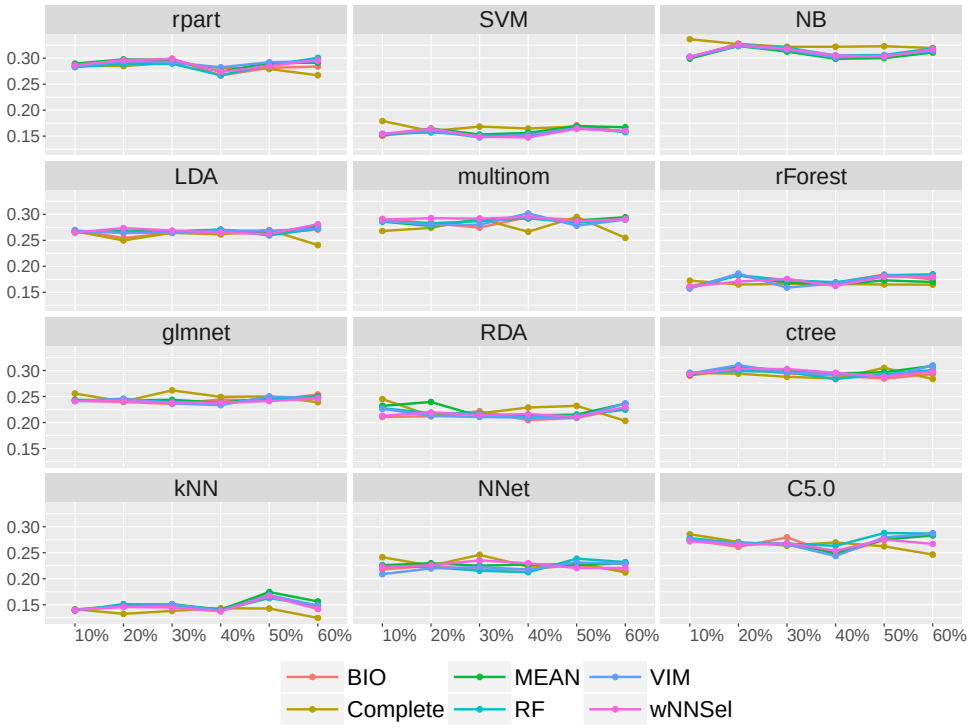


Figure C.3.: Sonar data: average misclassification error

D. Appendix for Chapter 9

Effect of Signal-to-Noise Ratio

Table D.1.: Average MSIEs for different imputation methods.

n	p	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
				Low signal-to-noise ratio							
50	30	MAR	10%	1.30	0.85	0.90	0.62	0.36	**0.33**	0.90	1.02
			20%	1.31	0.88	0.94	0.66	0.38	**0.34**	0.95	1.26
			30%	1.29	0.89	0.95	0.68	0.38	**0.36**	1.03	1.61
		MCAR	10%	1.08	0.74	0.74	0.52	0.33	**0.30**	0.72	0.95
			20%	1.08	0.76	0.80	0.55	0.34	**0.31**	0.75	1.15
			30%	1.11	0.78	0.83	0.59	0.36	**0.32**	0.83	1.47
100	15	MAR	10%	2.09	0.88	0.91	0.69	0.47	**0.42**	0.50	0.62
			20%	1.88	0.88	0.91	0.70	0.48	**0.45**	0.62	0.65
			30%	1.80	0.93	0.96	0.74	0.50	**0.48**	0.86	0.67
		MCAR	10%	1.05	0.58	0.61	0.45	0.35	**0.31**	0.48	0.51
			20%	1.07	0.57	0.66	0.46	0.36	**0.30**	0.48	0.53
			30%	1.06	0.58	0.70	0.47	0.36	**0.33**	0.48	0.55
				High signal-to-noise ratio							
50	30	MAR	10%	1.24	0.84	0.88	0.60	0.36	**0.33**	0.93	1.13
			20%	1.24	0.85	0.90	0.63	0.36	**0.33**	0.98	1.37
			30%	1.23	0.87	0.92	0.66	0.38	**0.34**	1.03	1.74
		MCAR	10%	1.06	0.73	0.73	0.52	0.33	**0.29**	0.78	1.05
			20%	1.08	0.76	0.79	0.55	0.34	**0.30**	0.80	1.26
			30%	1.10	0.78	0.83	0.58	0.35	**0.31**	0.85	1.55
100	15	MAR	10%	1.88	0.88	0.89	0.66	0.47	**0.45**	0.65	0.71
			20%	1.69	0.84	0.89	0.65	0.45	**0.41**	0.76	0.70
			30%	1.88	0.88	0.89	0.66	0.47	**0.45**	0.65	0.70
		MCAR	10%	1.01	0.57	0.60	0.44	0.34	**0.33**	0.58	0.61
			20%	1.01	0.58	0.66	0.46	0.36	**0.32**	0.58	0.63
			30%	1.00	0.58	0.69	0.47	0.36	**0.32**	0.58	0.65

Table D.2.: Coverage rates for 95% confidence intervals for regression coefficients using different imputation methods.

n	p	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
				Low signal-to-noise ratio							
50	30	MAR	10%	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.93
			20%	1.00	1.00	1.00	1.00	1.00	0.94	0.89	0.94
			30%	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.98
		MCAR	10%	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.93
			20%	1.00	1.00	1.00	1.00	1.00	0.94	0.86	0.96
			30%	1.00	1.00	1.00	1.00	1.00	0.94	0.74	0.99
100	15	MAR	10%	0.94	0.97	0.98	0.98	0.98	0.94	0.94	0.94
			20%	0.91	0.97	0.97	0.97	0.98	0.93	0.90	0.94
			30%	0.94	0.97	0.98	0.98	0.98	0.94	0.93	0.94
		MCAR	10%	0.98	0.99	0.98	0.98	0.98	0.94	0.94	0.94
			20%	0.97	0.98	0.98	0.98	0.98	0.95	0.93	0.94
			30%	0.97	0.98	0.98	0.98	0.98	0.95	0.89	0.93
				High signal-to-noise ratio							
50	30	MAR	10%	1.00	1.00	1.00	1.00	1.00	0.93	0.92	0.93
			20%	1.00	1.00	1.00	1.00	1.00	0.93	0.87	0.95
			30%	1.00	1.00	1.00	1.00	1.00	0.93	0.81	0.98
		MCAR	10%	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.94
			20%	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.97
			30%	1.00	1.00	1.00	1.00	1.00	0.93	0.70	0.99
100	15	MAR	10%	0.84	0.97	0.97	0.97	0.98	0.95	0.94	0.95
			20%	0.78	0.94	0.94	0.95	0.98	0.93	0.91	0.94
			30%	0.78	0.92	0.92	0.94	0.97	0.92	0.81	0.93
		MCAR	10%	0.95	0.98	0.98	0.98	0.98	0.95	0.93	0.94
			20%	0.92	0.98	0.96	0.98	0.99	0.95	0.90	0.94
			30%	0.87	0.97	0.94	0.97	0.98	0.95	0.78	0.92

Table D.3.: Average variance and squared bias for ($\hat{\boldsymbol{\beta}}$) obtained by different imputation methods.

n	p	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
				Var($\hat{\boldsymbol{\beta}}$)							
50	15	MCAR	10%	0.3364	0.3450	0.3492	0.3632	0.3689	**0.2935**	0.3605	0.3333
			20%	0.3232	0.3402	0.3379	0.3658	0.3762	**0.3014**	0.4857	0.3919
			30%	0.3204	0.3373	0.3429	0.3776	0.3787	**0.2922**	1.2322	0.4364
		MAR	10%	0.3522	0.3588	0.3611	0.3735	0.3815	**0.2919**	0.3388	0.3085
			20%	0.3512	0.3637	0.3556	0.3859	0.3798	**0.3090**	0.4258	0.3690
			30%	0.3753	0.3735	0.3537	0.3990	0.4045	**0.3113**	0.6244	0.4417
	30	MCAR	10%	0.0120	0.0122	0.0122	0.0126	0.0124	**0.0103**	0.0106	0.0107
			20%	0.0137	0.0134	0.0137	0.0141	0.0138	**0.0106**	0.0119	0.0116
			30%	0.0124	0.0127	0.0123	0.0135	0.0138	**0.0105**	0.0134	0.0124
		MAR	10%	0.0143	0.0142	0.0139	0.0142	0.0140	**0.0105**	0.0114	0.0113
			20%	0.0144	0.0143	0.0140	0.0142	0.0140	**0.0105**	0.0121	0.0116
			30%	0.0149	0.0147	0.0147	0.0156	0.0151	**0.0113**	0.0154	0.0137
100	15	MCAR	10%	0.1241	0.1342	0.1332	0.1403	0.1411	**0.1214**	0.1297	0.1266
			20%	0.1235	0.1296	0.1289	0.1403	0.1447	**0.1182**	0.1616	0.1435
			30%	**0.1224**	0.1341	0.1289	0.1502	0.1473	0.1225	0.3339	0.1732
		MAR	10%	0.1313	0.1313	0.1321	0.1399	0.1378	**0.1186**	0.1297	0.1194
			20%	0.1276	0.1339	0.1319	0.1441	0.1419	**0.1181**	0.1487	0.1320
			30%	0.1244	0.1311	0.1261	0.1426	0.1358	**0.1182**	0.1792	0.1539
	45	MCAR	10%	0.2802	0.2802	0.2812	0.2844	0.2857	**0.1917**	0.2200	0.2191
			20%	0.2862	0.2885	0.2881	0.2952	0.2950	**0.1984**	0.2957	0.3125
			30%	0.2981	0.2945	0.2943	0.3061	0.3090	**0.2033**	0.6453	0.4446
		MAR	10%	0.2946	0.2894	0.2886	0.2896	0.2882	**0.1968**	0.2247	0.2227
			20%	0.2952	0.2913	0.2905	0.2982	0.2955	**0.1988**	0.2547	0.2526
			30%	0.3036	0.3004	0.2949	0.3055	0.3037	**0.2036**	0.3521	0.3544
	75	MCAR	10%	0.0086	0.0079	0.0079	0.0074	0.0068	**0.0061**	0.0072	0.0066
			20%	0.0110	0.0099	0.0102	0.0086	0.0078	**0.0066**	0.0094	0.0078
			30%	0.0139	0.0127	0.0129	0.0109	0.0096	**0.0076**	0.0115	0.0100
		MAR	10%	0.0157	0.0134	0.0122	0.0122	0.0096	**0.0070**	0.0097	0.0079
			20%	0.0212	0.0177	0.0149	0.0154	0.0124	**0.0086**	0.0141	0.0115
			30%	0.0245	0.0212	0.0177	0.0194	0.0150	**0.0101**	0.0173	0.0161

n	p	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
				Bias^2($\hat{\boldsymbol{\beta}}$)							
50	15	MCAR	10%	0.0200	0.0069	0.0076	0.0029	0.0027	0.0023	0.0022	**0.0021**
			20%	0.0396	0.0203	0.0232	0.0133	0.0108	0.0061	**0.0029**	0.0037
			30%	0.0535	0.0344	0.0352	0.0196	0.0214	0.0176	**0.0037**	0.0103
		MAR	10%	0.0190	0.0064	0.0078	0.0044	0.0042	0.0015	0.0021	**0.0012**
			20%	0.0316	0.0143	0.0143	0.0071	0.0074	0.0059	0.0042	**0.0032**
			30%	0.0425	0.0263	0.0284	0.0173	0.0174	0.0104	**0.0094**	0.0097
	30	MCAR	10%	0.1962	0.1929	0.1926	0.1907	**0.1901**	0.1921	0.1934	0.1926
			20%	0.1977	0.1947	0.1949	0.1914	**0.1902**	0.1936	0.1950	0.1946
			30%	0.2026	0.1984	0.1991	0.1932	**0.1915**	0.1954	0.1944	0.1950
		MAR	10%	0.1945	0.1908	0.1928	0.1890	**0.1886**	0.1945	0.1960	0.1941
			20%	0.1975	0.1937	0.1954	0.1915	**0.1907**	0.1964	0.1976	0.1963
			30%	0.2012	0.1965	0.1976	0.1908	**0.1902**	0.1958	0.1957	0.1974
100	15	MCAR	10%	0.0241	0.0066	0.0082	0.0019	0.0009	0.0007	0.0002	**0.0001**
			20%	0.0426	0.0195	0.0236	0.0094	0.0085	0.0046	0.0011	**0.0005**
			30%	0.0473	0.0266	0.0325	0.0117	0.0145	0.0098	0.0034	**0.0011**
		MAR	10%	0.0189	0.0075	0.0076	0.0023	0.0025	0.0011	0.0010	**0.0003**
			20%	0.0303	0.0144	0.0159	0.0057	0.0074	0.0053	0.0024	**0.0006**
			30%	0.0370	0.0192	0.0215	0.0082	0.0133	0.0090	0.0071	**0.0009**
	45	MCAR	10%	0.0103	0.0069	0.0068	0.0036	0.0028	0.0021	0.0015	**0.0011**
			20%	0.0185	0.0138	0.0143	0.0075	0.0053	0.0040	**0.0018**	0.0020
			30%	0.0211	0.0175	0.0189	0.0109	0.0091	0.0059	0.0053	**0.0019**
		MAR	10%	0.0121	0.0067	0.0071	0.0036	0.0029	0.0019	0.0018	**0.0010**
			20%	0.0166	0.0120	0.0115	0.0066	0.0062	0.0046	0.0040	**0.0014**
			30%	0.0199	0.0163	0.0145	0.0094	0.0086	0.0064	0.0047	**0.0019**
	75	MCAR	10%	0.0075	0.0074	0.0074	0.0072	0.0074	0.0068	**0.0067**	0.0080
			20%	0.0085	0.0079	0.0078	0.0074	0.0077	0.0074	**0.0072**	0.0102
			30%	0.0102	0.0088	0.0088	**0.0075**	0.0081	0.0078	0.0078	0.0159
		MAR	10%	0.0214	0.0157	0.0122	0.0131	0.0114	0.0108	0.0181	**0.0101**
			20%	0.0321	0.0234	0.0155	0.0177	0.0152	0.0140	0.0287	**0.0139**
			30%	0.0425	0.0328	0.0200	0.0241	0.0203	**0.0180**	0.0388	0.0261

Table D.4.: Low signal-to-noise ratio: average coverage rates for 95% confidence intervals for regression coefficients using different imputation methods.

n	p	Mechanism	miss	df	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
50	30	MAR	10%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.93
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.93	0.94
				tLipS	—	—	—	—	—	0.94	0.92	0.93
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.93
			20%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.89	0.94
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.91	0.95
				tLipS	—	—	—	—	—	0.94	0.90	0.95
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.90	0.94
			30%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.98
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.84	0.98
				tLipS	—	—	—	—	—	0.94	0.83	0.98
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.98
		MCAR	10%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.93
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.93	0.94
				tLipS	—	—	—	—	—	0.94	0.92	0.93
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.93
			20%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.86	0.96
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.87	0.97
				tLipS	—	—	—	—	—	0.94	0.86	0.97
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.86	0.97
			30%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.74	0.99
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.75	0.99
				tLipS	—	—	—	—	—	0.94	0.75	1.00
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.74	0.99
100	15	MAR	10%	Normal	0.94	0.97	0.98	0.98	0.98	0.94	0.94	0.94
				tLip	0.96	0.98	0.98	0.98	0.99	0.94	0.94	0.95
				tLipS	0.96	0.99	0.99	0.99	0.99	0.94	0.94	0.94
				tRubin	0.95	0.98	0.98	0.98	0.99	0.94	0.94	0.94
			20%	Normal	0.91	0.97	0.97	0.97	0.98	0.93	0.90	0.94
				tLip	0.93	0.98	0.98	0.98	0.99	0.94	0.91	0.94
				tLipS	0.94	0.98	0.98	0.99	0.99	0.93	0.90	0.94
				tRubin	0.93	0.98	0.98	0.98	0.99	0.93	0.90	0.94
			30%	Normal	0.94	0.97	0.98	0.98	0.98	0.94	0.93	0.94
				tLip	0.96	0.98	0.98	0.98	0.99	0.94	0.94	0.95
				tLipS	0.96	0.99	0.99	0.99	0.99	0.94	0.93	0.94
				tRubin	0.95	0.98	0.98	0.98	0.99	0.94	0.93	0.94
		MCAR	10%	Normal	0.98	0.99	0.98	0.98	0.98	0.94	0.94	0.94
				tLip	0.99	0.99	0.99	0.99	0.99	0.95	0.94	0.94
				tLipS	0.99	0.99	0.99	0.99	0.99	0.94	0.94	0.94
				tRubin	0.99	0.99	0.99	0.99	0.99	0.94	0.94	0.94
			20%	Normal	0.97	0.98	0.98	0.98	0.98	0.95	0.93	0.94
				tLip	0.98	0.99	0.99	0.99	0.99	0.95	0.93	0.95
				tLipS	0.99	0.99	0.99	0.99	0.99	0.95	0.93	0.94
				tRubin	0.98	0.99	0.99	0.99	0.99	0.95	0.93	0.94
			30%	Normal	0.97	0.98	0.98	0.98	0.98	0.95	0.89	0.93
				tLip	0.98	0.99	0.98	0.99	0.99	0.95	0.89	0.94
				tLipS	0.98	0.99	0.99	0.99	0.99	0.95	0.89	0.94
				tRubin	0.98	0.99	0.98	0.99	0.99	0.95	0.89	0.94

Table D.5.: High signal-to-noise ratio: average coverage rates for 95% confidence intervals for regression coefficients using different imputation methods.

n	p	Mechanism	miss	df	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
50	30	MAR	10%	Normal	1.00	1.00	1.00	1.00	1.00	0.93	0.92	0.93
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.93	0.94
				tLipS						0.93	0.92	0.93
				tRubin	1.00	1.00	1.00	1.00	1.00	0.93	0.92	0.93
			20%	Normal	1.00	1.00	1.00	1.00	1.00	0.93	0.87	0.95
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.88	0.96
				tLipS						0.93	0.87	0.96
				tRubin	1.00	1.00	1.00	1.00	1.00	0.93	0.87	0.95
			30%	Normal	1.00	1.00	1.00	1.00	1.00	0.93	0.81	0.98
				tLip	1.00	1.00	1.00	1.00	1.00	0.94	0.82	0.99
				tLipS						0.93	0.81	0.99
				tRubin	1.00	1.00	1.00	1.00	1.00	0.93	0.81	0.99
		MCAR	10%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.94
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.93	0.94
				tLipS						0.94	0.92	0.94
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.92	0.94
			20%	Normal	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.97
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.85	0.98
				tLipS						0.94	0.83	0.98
				tRubin	1.00	1.00	1.00	1.00	1.00	0.94	0.83	0.98
			30%	Normal	1.00	1.00	1.00	1.00	1.00	0.93	0.70	0.99
				tLip	1.00	1.00	1.00	1.00	1.00	0.95	0.71	1.00
				tLipS						0.93	0.71	1.00
				tRubin	1.00	1.00	1.00	1.00	1.00	0.93	0.71	0.99
100	15	MAR	10%	Normal	0.84	0.97	0.97	0.97	0.98	0.95	0.94	0.95
				tLip	0.88	0.98	0.98	0.98	0.99	0.95	0.94	0.96
				tLipS	0.89	0.98	0.98	0.98	0.99	0.95	0.94	0.95
				tRubin	0.87	0.97	0.98	0.98	0.99	0.95	0.94	0.95
			20%	Normal	0.78	0.94	0.94	0.95	0.98	0.93	0.91	0.94
				tLip	0.82	0.95	0.96	0.96	0.98	0.93	0.92	0.94
				tLipS	0.84	0.96	0.97	0.97	0.99	0.93	0.92	0.94
				tRubin	0.82	0.95	0.96	0.96	0.98	0.93	0.92	0.94
			30%	Normal	0.78	0.92	0.92	0.94	0.97	0.92	0.81	0.93
				tLip	0.82	0.94	0.94	0.95	0.98	0.92	0.83	0.94
				tLipS	0.83	0.95	0.95	0.96	0.99	0.92	0.82	0.94
				tRubin	0.82	0.93	0.94	0.95	0.98	0.92	0.82	0.94
		MCAR	10%	Normal	0.95	0.98	0.98	0.98	0.98	0.95	0.93	0.94
				tLip	0.97	0.99	0.98	0.99	0.99	0.95	0.94	0.95
				tLipS	0.97	0.99	0.99	0.99	0.99	0.95	0.94	0.95
				tRubin	0.96	0.98	0.98	0.99	0.99	0.95	0.94	0.95
			20%	Normal	0.92	0.98	0.96	0.98	0.99	0.95	0.90	0.94
				tLip	0.94	0.98	0.97	0.99	0.99	0.95	0.91	0.95
				tLipS	0.94	0.99	0.98	0.99	0.99	0.95	0.91	0.95
				tRubin	0.93	0.98	0.97	0.98	0.99	0.95	0.91	0.95
			30%	Normal	0.87	0.97	0.94	0.97	0.98	0.95	0.78	0.92
				tLip	0.90	0.98	0.95	0.98	0.99	0.95	0.80	0.94
				tLipS	0.91	0.98	0.96	0.99	0.99	0.95	0.80	0.95
				tRubin	0.90	0.98	0.95	0.98	0.99	0.95	0.79	0.94

Table D.6.: Average MSE($\hat{\boldsymbol{\beta}}$) using ridge regression for different multiple imputation methods for $n = 100$ and $p = 200$, when true active set is known.

SNR	Mechanism	miss	MEAN	BIO	VIM	RF	miNNboot	miNNseq	IRMI	MICE
Low	MCAR	10%	0.0151	0.0151	0.0150	0.0151	0.0149	**0.0128**	0.0130	0.0130
		20%	0.0166	0.0166	0.0165	0.0162	0.0157	**0.0131**	0.0137	0.0139
		30%	0.0175	0.0170	0.0171	0.0166	0.0160	**0.0132**	0.0142	0.0145
	MAR	10%	0.0216	0.0195	0.0194	0.0193	0.0167	**0.0136**	0.0167	0.0147
		20%	0.0261	0.0235	0.0230	0.0236	0.0192	**0.0148**	0.0196	0.0166
		30%	0.0304	0.0276	0.0276	0.0272	0.0217	**0.0158**	0.0217	0.0182
High	MCAR	10%	0.0078	0.0074	0.0075	0.0074	0.0073	**0.0071**	0.0076	0.0080
		20%	0.0094	0.0089	0.0089	0.0087	0.0083	**0.0081**	0.0091	0.0098
		30%	0.0113	0.0104	0.0105	0.0100	0.0094	**0.0090**	0.0108	0.0113
	MAR	10%	0.0138	0.0120	0.0118	0.0117	0.0098	**0.0090**	0.0120	0.0099
		20%	0.0178	0.0153	0.0149	0.0151	0.0122	**0.0112**	0.0149	0.0121
		30%	0.0215	0.0186	0.0174	0.0182	0.0145	**0.0126**	0.0177	0.0138

References

Acuña, E. and C. Rodriguez (2004). The treatment of missing values and its effect in the classifier accuracy. In D. Banks, L. House, F. R. McMorris, P. Arabie, and W. Gaul (Eds.), *Classification, Clustering, and Data Mining Applications*, pp. 639–647. Springer, Berlin.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE transactions on automatic control 19*(6), 716–723.

Alizadeh, A. A., M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, et al. (2000). Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. *Nature 403*(6769), 503–511.

Allison, P. D. (2001). *Missing data*. Number 136. Sage.

Allison, P. D. (2005). Imputation of categorical variables with proc mi. *SUGI 30 proceedings 113*(30), 1–14.

Anderberg, M. R. (1973). *Cluster analysis for applications*. Probability and Mathematical Statistics, New York: Academic Press, 1973.

Anders, S., P. T. Pyl, and W. Huber (2014). Htseq–a python framework to work with high-throughput sequencing data. *Bioinformatics*, btu638.

Andridge, R. R. and R. J. Little (2010). A review of hot deck imputation for survey non-response. *International statistical review 78*(1), 40–64.

Atkeson, C., A. Moore, and S. Schaal (1997). Locally weighted learning. *Artificial Intelligence Review 11*(1-5), 11–73.

Audigier, V., F. Husson, and J. Josse (2016). A principal component method to impute missing values for mixed data. *Advances in Data Analysis and Classification 10*(1), 5–26.

Batista, G. E. and M. C. Monard (2002). A study of k-nearest neighbour as an imputation method. *HIS 87*, 251–260.

Batista, G. E. A. P. A. and M. C. Monard (2003). An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence 17*, 519–533.

Bello, A. (1994). A bootstrap method for using imputation techniques for data with missing values. *Biometrical journal 36*(4), 453–464.

Bø, T. H., B. Dysvik, and I. Jonassen (2004). Lsimpute: accurate estimation of missing values in microarray data with least squares methods. *Nucleic Acids Research 32*(3), e34–e34.

Branden, K. V. and S. Verboven (2009). Robust data imputation. *Computational Biology and Chemistry 33*(1), 7–13.

Brás, L. P. and J. C. Menezes (2007). Improving cluster-based missing value estimation of dna microarray data. *Biomolecular Engineering 24*(2), 273–282.

Breiman, L. (2001). Random forests. *Machine learning 45*(1), 5–32.

Brock, G. N., J. R. Shaffer, R. E. Blakesley, M. J. Lotz, and G. C. Tseng (2008). Which missing value imputation method to use in expression profiles: a comparative study and two selection schemes. *BMC Bioinformatics 9*(1), 12.

Cai, Z., M. Heydari, and G. Lin (2006). Iterated local least squares microarray missing value imputation. *Journal of Bioinformatics and Computational Biology 4*(05), 935–957.

Carpenter, J. R., M. G. Kenward, and I. R. White (2007). Sensitivity analysis after multiple imputations under missing at random: a weighting approach. *Statistical Methods in Medical Research 16(3)*, 259–275.

Chen, J. and J. Shao (2000). Nearest neighbor imputation for survey data. *Journal of official statistics 16*(2), 113.

Chen, J. and J. Shao (2001). Jackknife variance estimation for nearest-neighbor imputation. *Journal of the American Statistical Association 96*(453), 260–269.

Chiodi, M. (1990). A partition type method for clustering mixed data. *Rivista di statistica applicata 2*, 135–147.

Clark, P. and T. Niblett (1989a). The CN2 induction algorithm. *Machine Learning 3*, 261–283.

Clark, P. and T. Niblett (1989b). The cn2 induction algorithm. *Machine learning 3*(4), 261–283.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychosocial Measurement 20*, 37–46.

Cramér, H. (1946). Methods of mathematical statistics. *Princeton: Princeton Univer-sity Press. CramerMethods of Mathematical Statistics1946*.

Cranmer, S. J. and J. Gill (2013). We have to be discrete about this: A non-parametric imputation technique for missing categorical data. *British Journal of Political Science 43*(02), 425–449.

Cule, E., P. Vineis, and M. De Iorio (2011). Significance testing in ridge regression for genetic data. *BMC bioinformatics 12*(1), 372.

De Micheaux, P. L., R. Drouilhet, and B. Liquet (2011). *Le logiciel R: Maitriser le langage-Effectuer des analyses statistiques*. Springer Science & Business Media.

De'ath, G. and K. E. Fabricius (2000). Classification and regression trees: a powerful yet simple technique for ecological data analysis. *Ecology 81*(11), 3178–3192.

Deng, Y., C. Chang, M. S. Ido, and Q. Long (2016). Multiple imputation for general missing data patterns in the presence of high-dimensional data. *Scientific reports 6*, 21689.

Dias, D. B., R. C. Madeo, T. Rocha, H. H. Biscaro, and S. M. Peres (2009). Hand movement recognition for brazilian sign language: a study using distance-based neural networks. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pp. 697–704. IEEE.

Dobin, A., C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras (2013). Star: ultrafast universal rna-seq aligner. *Bioinformatics 29*(1), 15–21.

Dudoit, S., J. Fridlyand, and T. P. Speed (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association 97*, 77–87.

Edgar Acuna, E. and C. Rodriguez (2004). The treatment of missing values and its effect in the classifier accuracy. In *Proceedings of the Meeting of the International Federation of Classification Societies (IFCS)*, pp. 639–47.

Efron, B. (1979). Bootstrap methods: another look at the jackknife. *Annals of statistics 7*, 14–26.

Efron, B. (1987). Better bootstrap confidence intervals. *Journal of the American statistical Association 82*(397), 171–185.

Efron, B. (1994). Missing data, imputation, and the bootstrap. *Journal of the American Statistical Association 89*(426), 463–475.

Eisemann, N., A. Waldmann, and A. Katalinic (2011). Imputation of missing values of tumour stage in population-based cancer registration. *BMC medical research methodology 11*(1), 1.

Erosheva, E. A., S. E. Fienberg, and B. W. Junker (2002). Alternative statistical models and representations for large sparse multi-dimensional contingency tables. In *Annales de la Faculté des sciences de Toulouse: Mathématiques*, Volume 11, pp. 485–505.

Eskelson, B. N., H. Temesgen, V. Lemay, T. M. Barrett, N. L. Crookston, and A. T. Hudak (2009). The roles of nearest neighbor methods in imputing missing data in forest inventory and monitoring databases. *Scandinavian Journal of Forest Research 24*(3), 235–246.

Ezzati-Rice, T. M., W. Johnson, M. Khare, R. J. Little, D. B. Rubin, and J. L. Schafer (1995). A simulation study to evaluate the performance of model-based multiple imputations in nchs health examination surveys. In *Proceedings of the Annual research Conference*, Volume 257266.

Faisal, S. (2017). *wNNSel: Weighted Nearest Neighbor Imputation of Missing Values using Selected Variables*. R package version 0.1.

Faisal, S. and G. Tutz (2017a). Imputation for missing values in high-dimensional data structures. International Workshop on Perspectives on High Dimensional Data VII, June 15-18, Guanajuato, Mexico.

Faisal, S. and G. Tutz (2017b). Imputation in high-dimensional mixed-type data by nearest neighbors. In *Proceedings of the 32th International Workshop on Statistical Modelling*, Volume II, Groningen, Netherlands, pp. 177–180. Statistical Modelling Society.

Faisal, S. and G. Tutz (2017c). Missing value imputation for gene expression data by tailored nearest neighbors. *Statistical Applications in Genetics and Molecular Biology 16*(2), 95–106.

Faisal, S. and G. Tutz (2017d, August 16-20). Multiple imputation using sequential nearest neighbors. In *The Third Annual Kliakhandler Conference on Bayesian Inference in Statistics and Statistical Genetics*, Michigan, USA.

Faisal, S. and G. Tutz (2017e). Nearest neighbor imputation for categorical data by weighting of attributes. *arXiv preprint arXiv:1710.01011 [stat.ME]*.

Farhangfar, A., L. Kurgan, and J. Dy (2008). Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition 41*, 3692–3705.

Feten, G., T. Almoy, and A. H. Aastveit (2005). Prediction of missing values in microarray and use of mixed models to evaluate the predictors. *Statistical Applications in Genetics and Molecular Biology 4*(1), 10.

Fix, E. and J. L. Hodges (1951). Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, DTIC Document.

Frank, I. E. and R. Todeschini (1994). *The data analysis handbook*, Volume 14. Elsevier.

Frazee, A. C., B. Langmead, and J. T. Leek (2011). Recount: a multi-experiment resource of analysis-ready rna-seq gene count datasets. *BMC Bioinformatics 12*(1), 449.

Friedman, J. H., R. Kohavi, and Y. Yun (1996). Lazy decision trees. In *AAAI/IAAI, Vol. 1*, pp. 717–724.

García-Laencina, P. J., J. Sancho-Gómez, and A. R. Figueiras-Vidal (2010). Pattern classification with missing data: a review. *Neural Computing and Applications 19*, 263–282.

Gheyas, I. A. and L. S. Smith (2010). A neural network-based framework for the reconstruction of incomplete data sets. *Neurocomputing 73*, 3039–3065.

Goeman, J. J., R. J. Meijer, and N. Chaturvedi (2017). *Penalized: L1 (lasso and fused lasso) and L2 (ridge) penalized estimation in GLMs and in the Cox model*. R package version 0.9-50.

Gower, J. C. (1971). A general coefficient of similarity and some of its properties. *Biometrics 27*(4), 857–871.

Grzymala-Busse, J. W. (1991). On the unknown attribute values in learning from examples. In *International Symposium on Methodologies for Intelligent Systems*, pp. 368–377. Springer.

Grzymala-Busse, J. W. and M. Hu (2001). A comparison of several approaches to missing attribute values in data mining. In W. Ziarko and Y. Yao (Eds.), *Rough Sets and Current Trends in Computing. RSCTC 2000. Lecture Notes in Computer Science, vol 2005.*, pp. 378–385. Springer, Berlin, Heidelberg.

Harley, C. B. and R. P. Reynolds (1987). Analysis of e. coli pormoter sequences. *Nucleic acids research 15*(5), 2343–2361.

Harrel, O. and X. H. Zhou (2007). Multiple imputation: Review of theory, implementation and software. *Statistics in Medicine 26*, 3057–3077.

Hastie, T., R. Tibshirani, B. Narasimhan, and G. Chu (2013). impute: impute: Imputation for microarray data. `http://www.bioconductor.org/packages/release/bioc/html/impute.html`. R package version 1.36.0.

Hastie, T., R. Tibshirani, G. Sherlock, M. Eisen, P. Brown, and D. Botstein (1999). Imputing missing data for gene expression arrays.

He, R. and T. Belin (2014). Multiple imputation for high-dimensional mixed incomplete continuous and binary data. *Statistics in medicine 33*(13), 2251–2262.

Hill, J. (2012). Four techniques for dealing with missing data in criminal justice.

Honaker, J., G. King, and M. Blackwell (2011). Amelia II: A program for missing data. *Journal of Statistical Software 45*(7), 1–47.

Horton, N. J. and K. P. Kleinman (2007). Much ado about nothing: A comparison of missing data methods and software to fit incomplete data regression models. *American Statistician 61*, 79–90.

Horton, N. J., S. R. Lipsitz, and M. Parzen (2003). A potential for bias when rounding in multiple imputation. *The American Statistician 57*(4), 229–232.

Hron, K., M. Templ, and P. Filzmoser (2010). Imputation of missing values for compositional data using classical and robust methods. *Computational Statistics and Data Analysis 54*(12), 3095–3107.

Hruschka, E. R. J., E. R. Hruschka, and N. F. F. Ebecken (2007). Bayesian networks for imputation in classification problems. *Journal of Intelligent Information Systems 29*, 231–252.

Hudak, A. T., N. L. Crookston, J. S. Evans, D. E. Hall, and M. J. Falkowski (2008). Nearest neighbor imputation of species-level, plot-scale forest structure attributes from lidar data. *Remote Sensing of Environment 112*(5), 2232–2245.

Jiang, C. and Z. Yang (2015). Cknni: an improved knn-based missing value handling technique. In *International Conference on Intelligent Computing*, pp. 441–452. Springer.

Johansson, P. and J. Hakkinen (2006). Improving missing value imputation of microarray data by using spot quality weights. *BMC Bioinformatics 7*(1), 306.

Jung, K., A. Gannoun, B. Sitek, O. Apostolov, A. Schramm, H. E. Meyer, K. Stühler, and W. Urfer (2006). Statistical evaluation of methods for the analysis of dynamic protein expression data from a tumor study. *RevStat-Statistical Journal 4*(1), 67–80.

Jung, K., A. Gannoun, B. Sitek, H. E. Meyer, K. Stühler, and W. Urfer (2005). Analysis of dynamic protein expression data. *RevStat-Statistical Journal 3*, 99–111.

Junninen, H., H. Niska, K. Tuppurainen, J. Ruuskanen, and M. Kolehmainen (2004). Methods for imputation of missing values in air quality data sets. *Atmospheric Environment 38*(18), 2895–2907.

Khan, J., J. S. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, et al. (2001). Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature medicine 7*(6), 673–679.

Kim, H., G. H. Golub, and H. Park (2005). Missing value estimation for dna microarray gene expression data: local least squares imputation. *Bioinformatics 21*(2), 187–198.

Kim, K.-Y., B.-J. Kim, and G.-S. Yi (2004). Reuse of imputed data in microarray analysis increases imputation efficiency. *BMC Bioinformatics 5*(1), 160.

Klambauer, G., T. Unterthiner, and S. Hochreiter (2013). Dexus: identifying differential expression in rna-seq studies with unknown conditions. *Nucleic acids research*, gkt834.

Kurgan, L. A., K. J. Cios, R. Tadeusiewicz, M. Ogiela, and L. S. Goodenday (2001). Knowledge discovery approach to automated cardiac spect diagnosis. *Artificial intelligence in medicine 23*(2), 149–169.

Li, K.-H. (1988). Imputation using markov chains. *Journal of Statistical Computation and Simulation 30*(1), 57–79.

Liao, S. G., Y. Lin, D. D. Kang, D. Chandra, J. Bon, N. Kaminski, F. C. Sciurba, and G. C. Tseng (2014). Missing value imputation in high-dimensional phenomic data: imputable or not, and how? *BMC bioinformatics 15*(1), 346.

Lichman, C. (2013). UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. http://archive.ics.uci.edu/ml.

Liew, A. W.-C., N.-F. Law, and H. Yan (2011). Missing value imputation for gene expression data: computational techniques to recover missing data from available information. *Briefings in Bioinformatics 12*(5), 498–513.

Lipsitz, S., M. Parzen, and L. P. Zhao (2002). A degrees-of-freedom approximation in multiple imputation. *Journal of Statistical Computation and Simulation 72*(4), 309–318.

Little, R. J. (1988). A test of missing completely at random for multivariate data with missing values. *Journal of the American Statistical Association 83*(404), 1198–1202.

Little, R. J. and D. B. Rubin (1987). *Statistical analysis with missing data*, Volume 539. Wiley New York.

Little, R. J. and D. B. Rubin (2002). *Statistical analysis with missing data. 2nd ed.* John Wiley & Sons.

Little, R. J. and D. B. Rubin (2014). *Statistical analysis with missing data.* John Wiley & Sons.

Little, R. J. and M. D. Schluchter (1985). Maximum likelihood estimation for mixed continuous and categorical data with missing values. *Biometrika*, 497–512.

Lock, R. H. (1993). 1993 new car data. *Journal of Statistics Education 1*(1).

Long, Q. and B. A. Johnson (2015). Variable selection in the presence of missing data: resampling and imputation. *Biostatistics 16*(3), 596–610.

Luengo, J., S. García, and F. Herrera (2012). On the choice of the best imputation methods for missing values considering three groups of classification methods. *Knowledge and information systems*, 1–32.

Malarvizhi, M. R. and D. A. S. Thanamani (2012). K-nearest neighbor in missing data imputation. *International Journal of Engineering Research and Development 5*.

Montgomery, S. B., M. Sammeth, M. Gutierrez-Arcelus, R. P. Lach, C. Ingle, J. Nisbett, R. Guigo, and E. T. Dermitzakis (2010). Transcriptome genetics using second generation sequencing in a caucasian population. *Nature 464*(7289), 773–777.

Moorthy, K., M. Saberi Mohamad, and S. Deris (2014). A review on missing value imputation algorithms for microarray gene expression data. *Current Bioinformatics 9*(1), 18–22.

Myrtveit, I., E. Stensrud, and U. H. Olsson (2001). Analyzing data sets with missing data: an empirical evaluation of imputation methods and likelihood-based methods. *Software Engineering, IEEE Transactions on 27*(11), 999–1013.

Newman, D. A. (2009). Missing data techniques and low response rates: The role of systematic nonresponse parameters. In C. E. Lance and R. J. Vandenberg (Eds.), *Statistical and Methodological Myths and Urban Legends*, Chapter 1, pp. 7–36. New York: Routledge: Tylor & Francis Group.

Nguyen, D. V., N. Wang, and R. J. Carroll (2004). Evaluation of missing value estimation for microarray data. *Journal of Data Science 2*(4), 347–370.

Nielsen, S. F. (2003). Proper and improper multiple imputation. *International Statistical Review 71*(3), 593–607.

Ouyang, M., W. J. Welsh, and P. Georgopoulos (2004). Gaussian mixture clustering and imputation of microarray data. *Bioinformatics 20*(6), 917–923.

Pantanowitz, A. and T. Marwala (2009). Missing data imputation through the use of the random forest algorithm. In *Advances in Computational Intelligence*, pp. 53–62. Springer.

Pickrell, J. K., J. C. Marioni, A. A. Pai, J. F. Degner, B. E. Engelhardt, E. Nkadori, J.-B. Veyrieras, M. Stephens, Y. Gilad, and J. K. Pritchard (2010). Understanding mechanisms underlying human gene expression variation with rna sequencing. *Nature 464*(7289), 768–772.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.

Quinlan, J. R. (2014). *C4. 5: programs for machine learning*. Elsevier.

R Core Team (2013). R: A language and environment for statistical computing. `http://www.R-project.org/`.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing.

Raghunathan, T. E. (2004). What do we do with missing data? some options for analysis of incomplete data. *Annual Review of Public Health 25*(1), 99–117. PMID: 15015914.

Raghunathan, T. E., J. M. Lepkowski, J. Van Hoewyk, and P. Solenberger (2001). A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey methodology 27*(1), 85–96.

Ramzan, S., C. Heumann, and G. Tutz (2016a, March 14-16). Bootstrap confidence interval after nearest neighbors imputation. In *14th International Conference on Statistical Sciences: Statistics for Better Decision-Making and Development*, Karachi, Pakistan.

Ramzan, S., C. Heumann, and G. Tutz (2016b, August 23-26). Inference when using nearest neighbors methods and the bootstrap. In *22nd International Conference on Computational Statistics*, Oviedo, Spain.

Ramzan, S. and G. Tutz (2016). Improved methods for the imputation of missing data by nearest neighbor methods. In *Proceedings of the 31th International Workshop on Statistical Modelling*, Volume I, Rennes, France, pp. 261–266. Statistical Modelling Society.

Ramzan, S., G. Tutz, and C. Heumann (2014). Improved methods for the imputation of missing data by nearest neighbor methods. In *Proceedings of the 29th International Workshop on Statistical Modelling*, Volume I, Göttingen, Germany, pp. 279–302. Statistical Modelling Society.

Rancourt, E. (1999). Estimation with nearest neighbour imputation at statistics canada. In *Proceedings of the Survey Research Methods Section*, pp. 131–138.

Rancourt, E., C. Särndal, and H. Lee (1994). Estimation of the variance in the presence of nearest neighbor imputation. In *Proceedings of the section on survey research methods*, pp. 888–893.

Rao, J. N. and C. Wu (1988). Resampling inference with complex survey data. *Journal of the american statistical association 83*(401), 231–241.

Rieger, A., T. Hothorn, and C. Strobl (2010). Random forests with missing values in the covariates.

Rizopoulos, D. (2006). ltm: An r package for latent variable modelling and item response theory analyses. *Journal of Statistical Software 17*(5), 1–25.

Romesburg, C. (2004). *Cluster analysis for researchers*. Lulu. com.

Rubin, D. B. (1976). Inference and missing data. *Biometrika 63*, 581–592.

Rubin, D. B. (1987). *Multiple imputation for nonresponse in surveys*. New York: Wiley.

Rubin, D. B. (1996). Multiple imputation after 18+ years. *Journal of the American statistical Association 91*(434), 473–489.

Rubin, D. B. (2004). *Multiple imputation for nonresponse in surveys*, Volume 81. John Wiley & Sons.

Rubin, D. B. and J. L. Schafer (1990). Efficiently creating multiple imputations for incomplete multivariate normal data. In *Proceedings of the Statistical Computing Section of the American Statistical Association*, Volume 83, pp. 88.

Rubin, D. B. and N. Schenker (1986). Multiple imputation for interval estimation from simple random samples with ignorable nonresponse. *Journal of the American Statistical Association 81*(394), 366–374.

Saar-Tsechansky, M. and F. Provost (2007). Handling missing values when applying classification models. *Journal of Machine Learning Research 8*, 1625–1657.

Schäfer, J., K. Strimmer, et al. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical applications in genetics and molecular biology 4*(1), 32.

Schafer, J. L. (1997). *Analysis of incomplete multivariate data*. CRC press.

Schafer, J. L. (2010). *Analysis of incomplete multivariate data*. CRC press.

Schafer, J. L. and J. W. Graham (2002). Missing data: our view of the state of the art. *Psychological methods 7*(2), 147–177.

Scheel, I., M. Aldrin, I. K. Glad, R. SÃ¸rum, H. Lyng, and A. Frigessi (2005). The influence of missing value imputation on detection of differentially expressed genes from microarray data. *Bioinformatics 21*(23), 4272–4279.

Schomaker, M. and C. Heumann (2016). Bootstrap inference when using multiple imputation. *arXiv preprint arXiv:1602.07933*.

Schuurmans, D. and R. Greiner (1997). Learning to classify incomplete examples. *Computational Learning Theory and Natural Learning Systems, Making Learning Systems Practical 4*, 87–105.

Schwender, H. (2012). Imputing missing genotypes with weighted k nearest neighbors. *Journal of Toxicology and Environmental Health, Part A 75*(8-10), 438–446.

Schwender, H. and A. Fritsch (2013). *scrime: Analysis of High-Dimensional Categorical Data such as SNP Data*. R package version 1.3.3.

Segal, M. R. (2004). Machine learning benchmarks and random forest regression. *Center for Bioinformatics & Molecular Biostatistics*.

Sehgal, M. S. B., I. Gondal, and L. Dooley (2004). K-ranked covariance based missing values estimation for microarray data classification. In *Hybrid Intelligent Systems, 2004. HIS'04. Fourth International Conference on*, pp. 274–279. IEEE.

Sehgal, M. S. B., I. Gondal, and L. S. Dooley (2005). Collateral missing value imputation: a new robust missing value estimation algorithm for microarray data. *Bioinformatics 21*(10), 2417–2423.

Shah, A. D., J. W. Bartlett, J. Carpenter, O. Nicholas, and H. Hemingway (2014). Comparison of random forest and parametric imputation models for imputing missing data using mice: a caliber study. *American journal of epidemiology 179*(6), 764–774.

Shao, J. (2009). Nonparametric variance estimation for nearest neighbor imputation. *Journal of Official Statistics 25*(1), 55–62.

Shao, J. and R. R. Sitter (1996). Bootstrap for imputed survey data. *Journal of the American Statistical Association 91*(435), 1278–1288.

Shao, J. and H. Wang (2008). Confidence intervals based on survey data with nearest neighbor imputation. *Statistica Sinica*, 281–297.

Sokal, R. R. and C. D. Michener (1958). A statistical method for evaluating systematic relationships. *The University of Kansas science bulletin 38*, 1409–1438.

Song, J. and T. R. Belin (2004). Imputation for incomplete high-dimensional multivariate normal data using a common factor model. *Statistics in medicine 23*(18), 2827–2843.

Städler, N. and P. Bühlmann (2014). Pattern alternating maximization algorithm for missing data in high-dimensional problems. *Journal of Machine Learning Research 15*, 1903–1928.

Stekhoven, D. J. (2013). *missForest: Nonparametric Missing Value Imputation using Random Forest*. R package version 1.4.

Stekhoven, D. J. and P. Bühlmann (2012). Missforest: non-parametric missing value imputation for mixed-type data. *Bioinformatics 28*(1), 112–118.

Tarsitano, A. and M. Falcone (2011). Missing-values adjustment for mixed-type data. *Journal of Probability and Statistics 2011*.

Templ, M., A. Alfons, A. Kowarik, and B. Prantner (2016). *VIM: Visualization and Imputation of Missing Values*. R package version 4.5.0.

Tibshirani, R., G. Chu, B. Narasimhan, and J. Li (2011). samr: Significance analysis of microarrays. *R package version 2*.

Towell, G. G., J. W. Shavlik, and M. O. Noordewier (1990). Refinement of approximate domain theories by knowledge-based neural networks. In *Proceedings of the eighth National conference on Artificial intelligence*, pp. 861–866. Boston, MA.

Tritchler, D., E. Parkhomenko, and J. Beyene (2009). Filtering genes for cluster and network analysis. *BMC bioinformatics 10*(1), 193.

Troyanskaya, O., D. Botstein, and R. Altman (2003). Missing value estimation. In D. Berrar, W. Dubitzky, and M. Granzow (Eds.), *A Practical Approach to Microarray Data Analysis*, pp. 65–75. Springer US.

Troyanskaya, O., M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman (2001). Missing value estimation methods for dna microarrays. *Bioinformatics 17*(6), 520–525.

Tsanas, A., M. A. Little, C. Fox, and L. O. Ramig (2014). Objective automatic assessment of rehabilitative speech treatment in parkinson's disease. *IEEE Transactions on Neural Systems and Rehabilitation Engineering 22*(1), 181–190.

Tuikkala, J., L. L. Elo, O. S. Nevalainen, and T. Aittokallio (2008). Missing value imputation improves clustering and interpretation of gene expression microarray data. *BMC bioinformatics 9*(1), 202.

Tutz, G. and S. Ramzan (2014). Improved methods for the imputation of missing data by nearest neighbor methods. Technical Report 172, Department of Statistics, Ludwig-Maximilians-University Munich, Germany.

Tutz, G. and S. Ramzan (2015). Improved methods for the imputation of missing data by nearest neighbor methods. *Computational Statistics and Data Analysis 90*, 84–99.

Van Buuren, S. (2007). Multiple imputation of discrete and continuous data by fully conditional specification. *Statistical methods in medical research 16*(3), 219–242.

van Buuren, S. and K. Groothuis-Oudshoorn (2011). mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software 45*(3), 1–67.

Van Buuren, S. and K. Oudshoorn (1999). *Flexible mutlivariate imputation by MICE.* Leiden: TNO.

Verboven, S., K. V. Branden, and P. Goos (2007). Sequential imputation for missing values. *Computational Biology and Chemistry 31*(5), 320–327.

Waljee, A. K., A. Mukherjee, A. G. Singal, Y. Zhang, J. Warren, U. Balis, J. Marrero, J. Zhu, and P. D. Higgins (2013). Comparison of imputation methods for missing laboratory data in medicine. *BMJ Open 3*(8).

Wang, H. and S. Wang (2010). Mining incomplete survey data through classification. *Knowledge and Information System 24(2)*, 221–233.

Wasito, I. and B. Mirkin (2005). Nearest neighbour approach in the least-squares data imputation algorithms. *Information Sciences 169*(1), 1–25.

White, I. R., P. Royston, and A. M. Wood (2011). Multiple imputation using chained equations: Issues and guidance for practice. *Statistics in Medicine 30*, 377–399.

Wong, J. (2013). imputation: imputation. `http://CRAN.R-project.org/package=imputation`. R package version 2.0.1.

Yoon, D., E.-K. Lee, and T. Park (2007). Robust imputation method for missing values in microarray data. *BMC Bioinformatics 8*(Suppl 2), S6.

Zhang, X., X. Song, H. Wang, and H. Zhang (2008). Sequential local least squares imputation estimating missing value of microarray data. *Computers in Biology and Medicine 38*(10), 1112–1120.

Zhao, Y. and Q. Long (2016). Multiple imputation in the presence of high-dimensional data. *Statistical methods in medical research 25*(5), 2021–2035.

Eidesstattliche Versicherung

(Siehe Promotionsordnung vom 12. Juli 2011, Â§ 8 Abs. 2 Pkt. 5)

Hiermit erkläre ich an Eidesstatt, dass die Dissertation von mir selbstständig, ohne unerlaubte Beihilfe angefertigt ist.

München, den Shahla Faisal