MISSING DATA IMPUTATION USING MACHINE LEARNING AND NATURAL LANGUAGE PROCESSING FOR CLINICAL DIAGNOSTIC CODES

Arkopal Choudhury

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Biostatistics in the Gillings School of Global Public Health.

Chapel Hill
2020

Approved by:

Michael R. Kosorok

Feng-Chang Lin

Anna Kucharska-Newton

John S. Preisser

Haibo Zhou

ABSTRACT

Arkopal Choudhury: Missing Data Imputation Using Machine Learning and Natural Language Processing for Clinical Diagnostic Codes
(Under the direction of Michael R. Kosorok)

Imputation of missing data is a common application in supervised classification problems, where the feature matrix of the training dataset has various degrees of missingness. Most of the former studies do not take into account the presence of the class label in the classification problem with missing data. A widely used solution to this problem is missing data imputation based on the lazy learning technique, k-Nearest Neighbor (KNN) approach. We work on a variant of this imputation algorithm using Gray's distance and Mutual Information (MI), called Class-weighted Gray's k-Nearest Neighbor (CGKNN) approach. Gray's distance works well with heterogeneous mixed-type data with missing instances, and we weigh distance with mutual information (MI), a measure of feature relevance, between the features and the class label. This method performs better than traditional methods for classification problems with mixed data, as shown in simulations and applications on University of California, Irvine (UCI) Machine Learning datasets (http://archive.ics.uci.edu/ml/index.php).

Data being lost to follow up is a common problem in longitudinal data, especially if it involves multiple visits over a long period of time. If the outcome of interest is present in each time point, despite missing covariates due to follow-up (like outcome ascertained through phone calls), then random forest imputation would be a good imputation technique for the missing covariates. The missingness of the data involves more complicated interactions over time since most of the covariates and the outcome have repeated measurements over time. Random forests are a good non-parametric

learning technique which captures complex interactions between mixed type data. We propose a proximity imputation and missForest type covariate imputation with random splits while building the forest. The performance of the imputation techniques used is compared to existing techniques in various simulation settings.

The Atherosclerosis Risk in Communities (ARIC) Study Cohort is a longitudinal study which started in 1987-1989 to collect data on participants across 4 states in the USA, aimed at studying the factors behind heart diseases. We consider patients at the 5th visit (occurred in 2013) and enrolled in continuous Medicare Fee-For-Service (FFS) insurance in the last 6 months prior to their visit, so that their hospitalization diagnostic (ICD) codes are available. Our aim is to characterize the hospitalization of patients having cognitive status ascertainment (classified into dementia, mild cognitive disorder or no cognitive disorder) in the 5th visit. Diagnostic codes for inpatient and outpatient visits identified from CMS (Centers for Medicare & Medicaid Services) Medicare FFS data linked with ARIC participant data are stored in the form of International Classification of Diseases and related health problems (ICD) codes. We treat these codes as a bag-of-words model to apply text mining techniques and get meaningful cluster of ICD codes.

To my late Thakuma (grandma) and my late Babusona - who instilled in me the love of Science and Maths. To my beloved family, thank you for inspiring a fourth Doctor.

ACKNOWLEDGMENTS

It has been quite a journey over the last 6 years. When I got into this program, I never imagined graduating with so many experiences - I have really learnt a lot about myself as well as life in general. I would like to thank all the wonderful people who helped me shape this journey. For academic help, I would like to thank my advisor Dr. Michael Kosorok, my committee members Dr. Anna Kucharska Newton, who worked with me in the ARIC-based projects, Dr. John Preisser, who helped me write my first collaborative manuscript in UNC Chapel Hill, Dr. Haibo Zhou who taught me Linear Models and Dr. Feng-Chang Lin, who gave me valuable suggestions to improve my first paper. I would also like to thank Crystal, Phoebe and Sujatro for helping me with various programming tips and concepts throughout my PhD life. The members of the Kosorok lab, Sebastian, Daniel, Michael (Lawson), Sean, Duyeol, Ben, Hunyong, Nikki and John - you really helped me with your ideas and insights.

I would also like to acknowledge the constant mental support provided by my parents, my uncle and my sister from Kolkata and the constant social support received from my friends in Chapel Hill, Durham and Raleigh. It was especially difficult moving from a big city like Kolkata, India to an extremely sparsely populated US college town. I am especially thankful to Suman (Chakraborty) for getting Sujatro, Sayan (Banerjee) and myself to form our band Phir Kaahe (Why again?) and perform on stage in the Carrboro Music Festival, 2017. The three of you really made my time in Chapel Hill colorful with music, positivity and endless parties - all of which helped me get over a dark 3rd year of my PhD life. I would also like to appreciate my other close friends in the Triangle area, Arkaprava, Sohini (Raha), Aniket, Sapna, Sumit, Lakshita, Tanmay, Sayan (Patra), Arnab (Hazra), Priyam and Debraj - all of you have helped

me by always being there to talk to or have fun. Thanks to all those who helped my party and liven up the college town experience. I would thank Jyotishka and Shalini (Choudhury) for the wonderful parties they organized, and I really missed those. It has been tough seeing most of you leave the Triangle area, and life hasn't been the same over the last one year without you. I would also like to thank my friends in Raleigh, Moumita, Indranil, Salil, Arnab (Chakraborty), Suman (Majumdar), Rahul (Ghoshal), Rahul (Chakraborty), Dhrubajyoti, Tuhin and Sukanya - it was a pleasure playing Mafia, cricket and board games with you. I also had the pleasure hanging out with my friends from Durham, Aritra (Dasgupta), Debarati and Sneha.

Over the last few years, I have realized what a gem my school friends and ISI friends are - keeping in touch all the time, even though we were separated by oceans. I would firstly thank my ISI friends, Sunil, Soumya Subhra, Rohit, Saketh, Joydeep, Abhishek, Aritra (Guha), Diptavo, Paromita, Nilanjana, Snehashis, Sagnik, Sutanoy, Avi, Purvasha and Debanjan (Majumdar), all of whom regularly check up and take the effort to meet up. I am fortunate to have a group of school friends whom I have known for decades and can safely call them my closest friends in the world. I would especially mention Prasun, Soumo, Tathagata, Atri, Shankha, Sagar and Ankit among them. I am also thankful to Shalini (Banerjee), Sohini (Bhattacharyya), Sweta, and Banani for being an active part of our Common Room. A person who deserves special mention is my cousin, Debanjan (Tapu) - you have helped me in more ways than I can name. Lastly, words cannot justify my gratitude to my two rivals and childhood best friends - Sayak and Subhojit. Both of you have shaped me during my SJC days and I never heard of kids fighting over highest test scores, but also being the best friends that can be. The 3 of us will all graduate in Spring 2020, albeit in an era of uncertainty surrounding Covid-19 - but it will be great to all end our education at the same time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ARIC	Atherosclerosis Risk in Communities
CA	Classification Accuracy
CART	Classification and Regression Trees
CGKNN	Classs-weighted Gray′s k-Nearest Neighbor
CMS	Centers for Medicare & Medicaid Services
EHR	Electronic Health Records
EM	Expectation Maximization
FFS	Fee-for-service
FWGKNN	Feature-Weighted Gray′s KNN Imputation
GKNN	Gray's KNN Imputation
GRA	Gray Relational Analysis
GRC	Gray Relational Coefficient
GRG	Gray Relational Grade
HEOM	Heterogeneous Euclidean Overlap Metric
ICD	International Statistical Classification of Diseases
IKNN	Iterative k-Nearest Neighbors
KNN	k-Nearest Neighbors Approach
KNNI	k-Nearest Neighbors Imputation
LVQ	Linear Vector Quantization
MAR	Missing At Random
MCAR	Missing Completely At Random
MI	Mutual Information
MICE	Multiple Imputation Using Chained Equations
MI-KNN	Mutual Information based k-Nearest Neighbor imputation
MNAR	Missing Not At Random

NMF	Non-negative Matrix Factorization
NNI	Nearest Neighbor Imputation
OOB	Out of Bag
OTFI	On The Fly Imputation
PCA	Principal Components Analysis
PLANN	Partial Logistic Artificial Neural Network
PMM	Predictive Mean Matching
RF	Random Forest
RMSE	Root Mean Squared Error
SRMI	Sequential Regression Multivariate Imputation
UCI	University of California, Irvine

CHAPTER 1: INTRODUCTION

Missing data imputation has been an area of research for quite some time now. It had been been first worked on by Donald B. Rubin and Roderick J.A. Little since the 1970s (Rubin 1977, Little and Rubin 1987). However most of the applications has been on simple problems involving missing data where parametric models have been chosen as a method of imputation and in most cases, the data matrix had just a few missing values (the percentage of missingness was low). There has been relatively less work done on imputation (substitution by estimation) of missing values by non-parametric methods like nearest neighbors, principal components, trees, etc. Non-parametric methods not only provide a flexible setting to apply the imputation methods on, but also are less biased than parametric methods in general. The only disadvantage suffered by non-parametric methods is the interpretability but it is counteracted by the other advantages and thus we prefer to look at these methods. The other setting where missing data imputation has not been worked out on is in relation to supervised classification problems when the outcome variable is known. Thus, we explore methods which will make use of the outcome variable while imputing the values in the data matrix. In Chapter 2 of this manuscript we look into the Class-weighted Gray's k-Nearest Neighbor (CGKNN) technique, which is a method of imputation for classification problems and works well for heterogeneous mixed type data too. This technique takes the class variable into account while imputing the data matrix in classification problem and this leads to better classification performance after imputation of the data matrix.

We also look into longitudinal data cases where the measures of various

covariates are missing over different time periods, but the outcome is present at each time period. This is different from loss due to follow up as in the latter case, we would have the outcome missing as well for later time periods. An example of the former type of missingness would be the Atherosclerosis Risk in Communities (ARIC) Study Cohort where 15,792 participants who took part in the 1st visit back in 1987-1989 across 4 states in the USA (ARIC investigators 1989) and only 6,538 people returned for the 5th visit in 2011-2013. Out of the 9,254 individuals who did not come to the 5th visit, for a fraction of them, certain variables of interest could be ascertained through phone calls. Diabetes is one such outcome measured in the 5th visit for the 6,538 participants and some of the remaining 9,254 participants partially lost to follow up. However, a lot of covariate information is missing for participants whose diabetic status was ascertained through phone calls. There are not many studies in non-parametric missing data imputation techniques in longitudinal studies. We use a modified random forest technique using random splits (Ishwaran and Lu 2008) to impute the characteristics of the longitudinal dataset which has missingness due to follow-up or other reasons, followed by a prediction of the outcome using a suitable model. We are particularly interested in extending this idea from simulation settings to measuring the prevalence of diabetes among the ARIC participants and the factors which affect it, but due to the partial drop-outs, this measure would be biased. We use the Out-Of-Bag (OOB) error estimate of random forests to compute our imputation performance. The details of this is discussed in Chapter 3 which deals with variety of random forest imputation techniques in longitudinal datasets..

Present day medication and insurance health care are run on ICD codes (International Classification of Diseases and related health problems) which are clinical diagnostic codes. We propose to transform these ICD-9 codes to a bag-of-words model by treating them as text. In the ARIC dataset, we consider

participants at the 5th visit and enrolled in continuous Medicare Fee-For-Service (FFS) Insurance in the 6 months prior to their visit, so that their hospitalization diagnostic (ICD-9) codes are available. Our aim is to distinguish between the hospitalization of patients who have been diagnosed with dementia from those without dementia at the 5th visit in the ARIC study, by clustering their hospitalization diagnostic codes using text mining methods. The cognitive status diagnosis is taken from the 5th visit of the ARIC study. Our initial clustering of ICD codes using Non-negative Matrix Factorization (NMF) looks good when combined with Gram Schmidt Orthogonalization. We hope to get meaningful patterns of comorbidity preceding dementia using better text mining methods, to predict the onset of the disease. The details of this is discussed in Chapter 5, which deals with using natural language processing to cluster ICD codes in hospitalization records and distinguish between the comorbidity patterns of the patients with and without a dementia, before the official diagnosis of the disease.

The rest of this manuscript is organized as follows. A new method of missing data imputation for classification problems used nearest neighbors is introduced and shown in rigorous details in Chapter 2. A random forest based missing data imputation technique for longitudinal datasets and its proposed application to the ARIC dataset is described in Chapter 3. Chapter 4 deals with a new way of looking into Clinical Diagnostic Codes by applying Natural Language Processing on them, and we look into applying the techniques of text mining on Medicare FFS Insurance enrolled patients of the ARIC cohort, in particular. The future work of each chapter is mentioned in the discussion section at the end of each chapter.

CHAPTER 2: MISSING DATA IMPUTATION FOR CLASSIFICATION PROBLEMS

2.1 Introduction

Many of the commonly used classification algorithms such as Classification and Regression Trees (CART) (Breiman et al. 1984) and Random Forests (Breiman 2001) do not have rigorous techniques for handling missing values in training data. Ignoring the datapoints with missing values and running the classification algorithm on complete cases only leads to loss of vital information (Little and Rubin 2002). The occurrence of missing data is one of the biggest challenges for data scientists solving classification problems in real-world data (Duda et al. 2012). These datasets can come from any walk of life, ranging from medical data (Troyanskaya et al. 2001) and survey responses to equipment faults and limitations (Le Gruenwald 2005). The reason for missingness can be human error in inputting data, incorrect measurements, non-response to surveys, etc. For example, an industrial database maintained by Honeywell, a company manufacturing and servicing complex equipment, has more than 50% missing data (Lakshminarayan et al. 1999) despite regulatory requirements for data collection. In wireless sensor networks, often due sensor faults, local interference or power outage (Le Gruenwald 2005) we can encounter missing data. In medical fields, patient health care records are often a by-product of patient care activities rather than an organized research protocol which leads to significant loss of information (Cios and Moore 2002). This leads to almost every patient record lacking some values as well as each attribute/feature having missing values. More than 40% of the datasets in the UCI Machine Learning Repository have missing values

(Newman et al. 2008).

Classification problems are aimed at developing a classifier from training data so that a new test observation can be correctly classified into one of the groups/classes. The class membership is assumed to be known for each observation of the training set whereas the corresponding attributes/features may have some missing values. The test dataset consists of new observations having the corresponding features but no class labels. The goal of the classification problem is to assign class labels to the test set (Alpaydin 2009). In our problem setup, we assume that some of the features are missing at random (MAR) for the training as well as the test dataset. One approach to classification is ignoring the observations with missing values and building a classifier. This is only feasible when the missingness is insignificant, however, and it has been demonstrated that even with a 5% missingness, proper imputation increases the classification accuracy (Farhangfar et al. 2008). We focus on imputation of missing values in the training as well as the test dataset so as to improve the overall performance of the classifier on the test data. Our proposed method takes into account the class label during imputation of the training features, and this ensures an overall improvement in classification.

The work related to missing data imputation can be divided into two categories, single imputation and multiple imputation. Single imputation strategies provide a single value for the missing datum. The earliest single value imputation strategy was Mean Imputation (Little and Rubin 2002) which ignores the input data distribution by imputing just one value for all missing instances of a feature. Other highly used single imputation methods are hot deck and cold deck imputation (Little and Rubin 2002), C4.5 (Quinlan 1993) and prediction based models (Schafer 1997). C4.5 works well with discrete data but not with numerical data, which has to be discretized to apply the algorithm (Tsai et al. 2008). Prediction based models depend on the correct

modeling of missing features and the relationship between them. Usually, incomplete datasets obtained from studies cannot be modeled accurately. The problem with single imputation techniques, in general, is they reduce the variance of the imputed dataset. These techniques are unable to calculate the standard error or confidence interval of the imputed values in the dataset. They are also very case-specific as they can meaningfully impute data only when the model is known or when the data is either numerical or discrete.

To solve the problems of single imputation, multiple imputation strategies generate several imputed datasets from which confidence intervals can be calculated. Multiple imputation is a process where several imputed datasets are created and the variance between these datasets reflect their uncertainty measures (Rubin 1977, Farhangfar et al. 2007). The earliest multiple imputation technique was the Expectation-Maximization (EM) Algorithm (Dempster et al. 1977). The EM Algorithm and its variants such as EM with bootstrapping (Honaker et al. 2011), assumes a parametric density function which fails miserably for features without a parametric density. A recent generalization of the EM Algorithm called Pattern Alternating Maximization with Lasso Penalty (MissPALasso) (Städler et al. 2014) has been applied to datasets with high dimensionality ($p >> n$), but also assuming normality. Bayesian multiple imputation algorithms have been applied only to multivariate normal samples (Li 1988, Rubin and Schafer 1990).

Regression Imputation (Gelman and Hill 2006) is also a popular multiple imputation technique where each feature is imputed using other features as predictor variables for the regression model. Sequential Regression Multivariate Imputation (SRMI) improves upon this by fitting appropriate predictive regression models and drawing values from the calculated model (Raghunathan et al. 2001). Incremental Attribute Regression Imputation (IARI) constructs a sequence of regression models to

iteratively impute the missing values and also uses the class label of each sample as a predictor variable (Stein and Kowalczyk 2016). In Multiple Imputation using Chained Equations (MICE), the conditional distribution of each missing feature must be specified given the other features (Buuren and Oudshoorn 1999). It is assumed that the feature matrix has a full multivariate distribution from which the conditional distribution of each feature is derived. The full distribution need not be specified, as long as the distribution of each feature is stated, a feature called fully conditional specification (Buuren 2007). MICE can handle mixed types of data. It has options for predictive mean matching, linear regression, binary and polytomous logistic regression, etc., and uses the Gibbs sampler to generate multiple imputations. However, for a given set of conditional distributions, a multivariate distribution may not exist (Buuren et al. 2006). The ideas of MICE and SRMI are combined in the MissForest approach (Stekhoven and Bühlmann 2011) which fits a random forest on the missing feature, using the other features as covariates and then predicts the missing values. This procedure is iterative and can handle mixed data, complex interactions, and high dimensions.

Machine Learning techniques such as Fuzzy c-Means (Sefidian and Daneshpour 2019), Multilayer Perceptrons (MLP) (García-Laencina et al. 2013) and k-Nearest Neighbors (KNN) (Batista and Monard 2002) are useful non-parametric approaches to imputation of missing data. Various machine learning algorithms such as k-Nearest Neighbors (KNN), Support Vector Machines (SVM) and decision trees have been used in imputation by framing the imputation problem as an optimization problem and solving it (Bertsimas et al. 2017). The Nearest Neighbor Imputation (NNI) approach is simple since there is no need to build a predictive model for the data. The basic KNN Imputation (KNNI) algorithm was first used for estimating DNA microarrays with the contribution of the k-Nearest Neighbors weighted by Euclidean distance

(Troyanskaya et al. 2001). The sequential KNN method was proposed using cluster-based imputation (Kim et al. 2004), followed by an iterative variant of the KNN imputation (IKNN) (Brás and Menezes 2007), both of which improves on KNNI. The Shelly Neighbors (SN) method improves the KNN rule by selecting only neighbors forming a shell around the missing datum, among the k closest neighbors (Zhang 2011). The first significant work in improving KNN imputation for classification based problems uses a Mutual Information (MI)-weighted distance metric as a measure of closeness of a feature to the class label (García-Laencina et al. 2009). The method is called Mutual Information based k-Nearest Neighbor (MI-KNN) Imputation. However, the distance metric used is Euclidean distance, which does not perform well with mixed-type data (Huang and Lee 2006). Alternatively, Grey Relational Analysis is shown to be more appropriate for capturing proximity between two instances with mixed data as well as missingness. Based on this, a Grey KNN (GKNN) imputation approach was built based on Grey distance instead of Euclidean distance and it was shown to outperform traditional KNN imputation techniques (Huang and Lee 2004, Zhang 2012). This grey distance-based KNN imputation is weighted by mutual information between features (measure of inter-feature relevance) and shown to outperform IKNN, GKNN and Fuzzy k-Means Imputation (FKMI) (Li et al. 2004) in most settings, and is called the Feature Weighted Grey k-Nearest Neighbor (FWGKNN) method (Pan et al. 2015). However, this method does not take into account each feature's association with the class label, which is crucial when dealing with classification problems. The FWGKNN method also assumes inter-dependency of features.

We propose a Class-weighted Grey k-Nearest Neighbor (CGKNN) imputation method where we calculate the MI of each feature with respect to the class label in the training dataset, use it for calculating the weighted Grey distance between the

instances, and then find the k-Nearest Neighbors of an instance with missing values. Using k-Nearest Neighbors, the missing value is imputed according to the weighted Grey distance. Our contributions can be summarized as follows:

1. We use a combination of Mutual Information between each feature and the classifier variable Y to weigh the Grey distance between instances in the feature matrix X. This metric is suited for tuning out any unnecessary features for classification and then finding the nearest neighbors relevant for imputation.

2. We solve an imputation problem with no underlying assumptions on the structure of the feature matrix X except that the data is missing completely at random (MCAR) or missing at random (MAR). Our method (CGKNN) is non-parametric in nature and does not assume any dependence between the individual features. This performs well even when the features are independent of each other.

3. The proposed CGKNN imputation method is suited well for classification problems where the training as well as the test datasets have missing values. The feature matrix can be mixed-type, i.e., have categorical and numeric data. Our method is suitable for mixed-data classification problems faced with missing values. Moreover, our problem approach takes much less time to initialize than the most similar alternative method, Feature Weighted Grey k-Nearest Neighbor (FWGKNN).

The remainder of this paper is organized as follows. In Section 2, we review the KNN imputation techniques used in previous work and then provide a detailed outline of our method. We also discuss how it can be extended to the test dataset for classification and also derive the time complexity of our algorithm. In section 3, we test our proposed method against 6 standard methods in simulation settings. We

evaluate our imputation method (CGKNN) in different simulation settings with classification where we artificially introduce missingness. We compare it with standard multiple imputation algorithms MICE and MissForest as well as the previous KNN based algorithms, Iterative KNNI (IKNN), Mutual Information based KNNI (MI-KNN), Grey KNNI (GKNN) and Feature-Weighted Grey KNNI (FWGKNN). In section 4, we demonstrate how our algorithm performs with classification tasks involving 3 UCI Machine Learning Repository datasets. We also check for improvement of classification accuracy after imputation of the missing data. Our method gives the best classification performance out of all evaluated methods. We conclude with a discussion and scope for future work in section 5.

2.2 Methodology

In this section, we pose the missing data problem which is encountered in classification tasks. We introduce the nearest neighbor (NN) approach and the previous works done on implementing variations on the KNN imputation approach. This is followed by the concepts of mutual information (MI) and grey relational analysis (GRA) used by our method of Class-weighted Grey k-Nearest Neighbor (CGKNN) imputation approach (Choudhury and Kosorok 2020). We then formalize our imputation algorithm and calculate its time complexity.

2.2.1 Formulation of the Problem

Let $X = \{X_i\}_{i=1}^{n}$ be an $n \times p$-dimensional dataset of n independent observations with p features/attributes and Y a response variable of the class labels influenced by X. We assume no dependence structure between the features in X. Let D be an $n \times p$-dimensional matrix indicating the missingness of corresponding values in the

dataset X. In practice, we obtain a random sample of size n of incomplete data associated with a population (X, Y, D), called the training data (Hastie et al. 2009) used to train the classifier

$$\mathscr{D} = \{(X_i, Y_i, D_i)\}_{i=1}^{n}, \tag{2.1}$$

where all the class labels in $\{Y_i\}_{i=1}^{n}$ are observed, $X_i = (X_{ij})_{j=1}^{p} = (X_{i1}, ..., X_{ip})$ represents the p features of the i-th observation along with indicator variables $D_i = (D_{ij})_{j=1}^{p}$ such that

$$D_{ij} = \begin{cases} 0, & X_{ij} \text{ is missing} \\ 1, & \text{otherwise.} \end{cases} \tag{2.2}$$

Without loss of generality, we can assume for each i, the observation $X_i = (X_{ij})_{j=1}^{p}$ contains p_0 categorical features for $j \in \{1, 2, ..., p_0\}$ and p_1 continuous features for $j \in \{p_0 + 1, ..., p_0 + p_1\}$ such that $p_0 + p_1 = p$. Let the j-th categorical feature contain k_j different values and the j-th continuous variable representing the $(p_0 + j)$-th feature of X_i, indexed by $j \in \{1, ..., p_1\}$ take values from a continuous set $C_j \subset \mathbb{R}$. For each of the categorical features, we can map the k_j different values to the first k_j natural numbers, such that $X_i \in \{1, ..., k_1\} \times ... \times \{1, ..., k_{p_0}\} \times C_1 \times ... \times C_{p_1} \subset \mathbb{R}^p$.

In this setting, we can assume that $\{(X_i, Y_i)\}_{i=1}^{n}$ satisfy the model

$$Y_i = g(X_i), \quad i = 1, 2, ..., n, \tag{2.3}$$

where $g(.)$ is an unknown function mapping a p-dimensional number (belonging to a subspace of $\mathbb{R}^p$) to a discrete set $\mathscr{G}$ representing the class labels and $Y_i \in \mathscr{G}$. We assume that $\mathscr{G}$ contains m values and thus the classification problem is based on m classes.

The task of any classification algorithm is to use the training dataset

$\{(X_i, Y_i)\}_{i=1}^n$ to estimate $g(.)$, which is referred to as 'training' a classifier $\hat{g}(.)$. Given a new set of ℓ observations, $X' = \{X_i'\}_{i=1}^{\ell}$, called the test dataset (Hastie et al. 2009), the classifier predicts the corresponding class $Y' = \{Y_i'\}_{i=1}^{\ell}$ using $\hat{Y_i'} = \hat{g}(X_i')$. Note that the test dataset X' can also contain missing values. Many classification algorithms have been shown to perform better in terms of classification accuracy after imputing the missing values in the feature matrix X (Farhangfar et al. 2008, Luengo et al. 2012) and then training the classifier. In this paper, we propose a nearest neighbor based imputation algorithm which is used to impute the missing values in X and then train the classifier $\hat{g}(.)$. The same algorithm can be extended to the test dataset and impute the missing values in X'.

In general, there are three different missing data mechanisms as defined in the statistical literature (Little and Rubin 2002):

1. Missing Completely at Random (MCAR): When the missingness of X does not depend on the missing or observed values of X. In other words, using D is as defined in (3.2),

$$P(D|X) = P(D), \quad \text{for all } X \tag{2.4}$$

2. Missing at Random (MAR): When the missingness of X depends on the observed values of X but not on the missing values of X. If we split the training dataset X into two parts, observed X_{obs} and missing X_{mis}, then

$$P(D|X) = P(D|X_{obs}), \quad \text{for all } X_{mis} \tag{2.5}$$

3. Not Missing at Random (NMAR): When the data is neither MCAR or MAR, the missingness of X depends on the missing values of X itself. This sort of missingness is difficult to model as the observed values of X give biased

estimates of the missing values.

$$P(D|X) = P(D|X_{obs}, X_{mis}) \tag{2.6}$$

For our problem, we assume that the missing data mechanism of X is either MCAR or MAR.

2.2.2 k-Nearest Neighbors (KNN) Imputation Algorithm

KNN is a widely used instance-based, lazy-learning algorithm (Wu et al. 2008). The basic assumption behind instance-based learning methods is that the instances of a dataset with missing values would lie "close" to other instances with similar properties (Aha et al. 1991). The KNN approach has been extended to imputation of missing data in various datasets (Troyanskaya et al. 2001). KNN imputation techniques work well when the distribution of the dataset is unknown. The basic algorithm works by calculating k nearest observations (out of the $n-1$ possible observations) from a particular observation with missing values. The distances are calculated using pre-imputed values in each observation. After calculating the k closest neighbors, mean of the other observations is used for imputation of continuous features and mode is used for imputation of the categorical features. Note that we do not create any predictive model in KNN imputation since it is an instance-based learning algorithm. Observations with multiple missing values of different type (continuous or categorical) can be imputed by KNN imputation.

2.2.2.1 Distance Metric for Mixed Data

Let there be two input vectors, X_a and X_b - whose features can be both continuous as well as categorical. The Heterogeneous Euclidean Overlap Metric

(HEOM) (Batista and Monard 2003), denoted as $d(X_a, X_b)$, is defined as

$$d(X_a, X_b) = \sqrt{\sum_{j=1}^{p} d_j(X_{aj}, X_{bj})^2}\ , \tag{2.7}$$

$$d_j(X_{aj}, X_{bj}) = \begin{cases} 1, & D_{aj} * D_{bj} = 0 \text{ from (3.2)} \\ d_0(X_{aj}, X_{bj}), & X_j \text{ is categorical} \\ d_N(X_{aj}, X_{bj}), & X_j \text{ is quantitative} \end{cases} \tag{2.8}$$

$$d_0(X_{aj}, X_{bj}) = \begin{cases} 0, & X_{aj} = X_{bj} \\ 1, & X_{aj} \neq X_{bj} \end{cases} \tag{2.9}$$

$$d_N(X_{aj}, X_{bj}) = \frac{|X_{aj} - X_{bj}|}{max(X_{.j}) - min(X_{.j})}\ , \tag{2.10}$$

where $max(X_{.j})$ means the maximum value of n observations of feature $X_{.j}$ and $min(X_{.j})$ means the minimum value of $X_{.j}$ when it is quantitative. The distance ranges from 0 to 1 and also takes the value 1, when either of the observations are missing.

There are two challenges in successfully implementing the KNN imputation approach to missing data - selecting k and suitable neighbors. One obvious option is choosing k using only non-missing parts (Kim et al. 2004). The algorithm moves forward by artificially inducing missingness in the non-missing data and for various values of k, it checks the predictive performance of imputing the artificial missing data. The value of k with the least error in prediction of missing values is chosen. In our proposed approach, we determine this parameter optimally using cross-validation on an initially mean or mode imputed dataset (Stone 1974).

2.2.2.2 KNN Imputation

Let us focus on the problem where the j-th input feature of X_i is missing (i.e., $D_{ij} = 0$ from (2.2)) and has to be imputed. The distances from X_i to all other instances ($\{X_k\}_{k=1,k\neq i}^{n}$) in the training set are computed using HEOM defined by (2.7)-(2.10), the k-nearest neighbors are chosen with least distances. Let $\mathscr{A}_{X_i} = \{a_\ell\}_{\ell=1}^{k}$ represents the set of k-nearest neighbors of X_i arranged in increasing order of its distance as defined by (2.7)-(2.10). The k-closest cases are selected after instances with missing entries in the incomplete feature are imputed using mean or mode imputation, depending on the type of feature (Troyanskaya et al. 2001).

After choosing k-nearest neighbors, the missing value imputation is estimated from the feature values of $\mathscr{A}_{X_i}$. For continuous variables, the imputed value ($\widetilde{X}_{ij}$) is $\widetilde{X}_{ij} = (1/k)\sum_{\ell=1}^{k} a_{\ell j}$. The weighted version of this average is based X_i (Dudani 1976), such that

$$\widetilde{X}_{ij} = \frac{\sum_{\ell=1}^{k} w_\ell \, a_{\ell j}}{k * \sum_{\ell=1}^{k} w_\ell}, \quad w_\ell = \frac{1}{d(X_i, a_\ell)^2}, \tag{2.11}$$

where w_ℓ denotes the corresponding weight of the ℓ-th nearest neighbor a_ℓ and $d(X_i, a_\ell)$ is as defined in (2.7)-(2.10).

For categorical or discrete variables, we impute the mode of the $j-$th feature of $\{a_\ell\}_{\ell=1}^{k}$ to $\widetilde{X}_{ij}$. This assumes all neighbors have the same importance in the imputation stage (Troyanskaya et al. 2001). An improvement to this is assigning a weight w_ℓ to each a_ℓ, with closer neighbors having greater weights. Using an approach similar to a distance-weighted KNN classifier (Dudani 1976), a suitable choice of w_ℓ is

$$w_\ell(X_i) = \frac{d(a_k, X_i) - d(a_\ell, X_i)}{d(a_k, X_i) - d(a_1, X_i)}\,, \tag{2.12}$$

where $d(.,.)$ is defined in (2.7)-(2.10) and w_ℓ is assigned a value of 1 when

$d(a_k, X_i) = d(a_1, X_i)$, that is, all the distances are equal. Otherwise, for k (> 1) neighbors, $0 \leq w_\ell \leq 1$. Suppose the j-th input feature $X_{.j}$ has V possible discrete values with n_v being the number of samples in $\mathscr{A}_{X_i}$ whose $j-$th feature has value $v,\ v = 1, 2, ..., V$. The weighted mode is chosen by the category v^* calculated by the category with the maximum weight in $\mathscr{A}_{X_i}$ given by

$$v^* = \arg\max_v \left\{ \sum_{\ell=1}^{n_v} w_\ell(X_i) \right\}. \tag{2.13}$$

Algorithm 1 Iterative KNN (IKNN) Imputation (García-Laencina et al. 2009)

Input: (X, Y, D) with $X \subset \mathbb{R}^{n \times p}$ containing missing entries and Y the class labels.

Output: Imputed feature matrix $\widetilde{X}$ with no missing values.

Procedure:

1. **Initialization:** Given the training dataset X, the missing values of p_0 categorical features are imputed by mode imputation and the missing values of the p_1 continuous features are imputed by mean imputation using the observed data. We call the initially imputed matrix $\widetilde{X}^0$.

2. **Choosing k:** We use this imputed matrix, $\widetilde{X}^0$, to calculate the optimum value of k using 10-fold cross validation (Stone 1974) to minimize the misclassification rate of predicting the class labels Y. This is the k used for choosing the nearest neighbors.

3. **Iterative Step:** Consider the iteration number t (≥ 1). In the t-th iteration, the imputed matrix $\widetilde{X}^t$ is obtained by imputing the missing continuous features (with corresponding $D_{ij} = 0$) using (2.11) and missing categorical features using (2.12)-(2.13). This step is repeated until the stopping criteria is reached.

4. **Stopping Criterion:** We stop at the d-th iteration when a stopping criteria is met. The stopping criteria we propose is

$$\max_{i,j:D_{ij}=0} |\widetilde{X}_{ij}^{d} - \widetilde{X}_{ij}^{d-1}| < \epsilon, \tag{2.14}$$

where $\epsilon = 10^{-4}$ is the chosen accuracy level.

2.2.3 Mutual Information (MI) for Classification

We can see that the above imputation algorithm does not consider the class label Y while computing the k-nearest neighbors. We can solve this using an effective procedure where the neighborhood is selected by considering the input feature relevance for classification (García-Laencina et al. 2009). This input feature relevance for classification is measured by calculating the Mutual Information (MI) between the feature $X_{.j}$ and the class variable Y.

2.2.3.1 Notion of MI

Suppose a discrete random variable X has a probability density function (pdf) given by $p(x) = P(X = x)$ where $x \in Support(X)$. The entropy of a random variable X is given by

$$H(X) = -\sum_{x \in Supp(X)} p(x)\ log\ p(x), \tag{2.15}$$

where log has base 2 in information theory, with the unit of entropy being bits. Now consider two random variables, X and Y. The joint entropy of X and Y is defined as

$$H(X,Y) = -\sum_{x \in \mathscr{X}} \sum_{y \in \mathscr{Y}} p(x,y)\ log\ p(x,y), \tag{2.16}$$

where $p(x,y)$ is the joint pdf of X and Y, both of them being discrete. The conditional entropy for the same pair of variables is given by

$$H(Y|X) = -\sum_{x \in \mathscr{X}} \sum_{y \in \mathscr{Y}} p(x,y)\ log\ p(y|x), \tag{2.17}$$

where $p(y|x)$ is the conditional pdf of Y given X.

For continuous random variables, the entropy of X is defined by

$$H(X) = -\int_x p(x)\ log\ p(x)\ dx, \tag{2.18}$$

and the joint and conditional entropy of continuous random variables X and Y is given by

$$\begin{aligned} H(X,Y) &= -\int_x \int_y p(x,y)\ log\ p(x,y)\ dy\ dx, & (2.19)\\ H(Y|X) &= -\int_x \int_y p(x,y)\ log\ p(y|x)\ dy\ dx,. & (2.20) \end{aligned}$$

Mutual information (MI) is based on entropy and it quantifies the uncertainty of X given Y. For discrete random variables X and Y it is

$$I(X;Y) = \sum_{x\in\mathscr{X}} \sum_{y\in\mathscr{Y}} p(x,y)\ log\ \frac{p(x,y)}{p(x)p(y)}\ , \tag{2.21}$$

and for continuous random variables it is

$$I(X;Y) = \int_x \int_y p(x,y)\ log\ \frac{p(x,y)}{p(x)p(y)}\ dx\ dy \tag{2.22}$$

The entropy and MI satisfy the following relationship

$$I(X;Y) = H(Y) - H(Y|X), \tag{2.23}$$

which is the reduction of the uncertainty of Y when X is known (Kullback 1997) since it can be re-written as $I(X;Y) = H(X) + H(Y) - H(X,Y)$. Compared to the Pearson correlation coefficient which only measures linear relationships, MI can measure any relationship between variables (Kullback 1997). MI ranges from -1 to 1

with MI being 0 for two independent variables.

2.2.3.2 Computation of MI in Classification Problems

Consider the class label Y for an m-class classification problem and let the number of observations in the y-th class be n_y such that $n_1 + n_2 + ... + n_m = n$, as mentioned in (2.1). In terms of classification problems, we are interested in finding the relevance of the j-th feature $X_{.j}$ with the class label Y, which is measured by their Mutual Information (MI) given by

$$I(X_{.j}; Y) = H(Y) - H(Y|X_{.j}), \tag{2.24}$$

In this equation, we have to estimate $H(Y)$ and $H(Y_{.j})$ to get $\hat{I}(X_{.j}; Y)$. Note that Y is always discrete and the entropy of class variable Y can be computed using (2.15) as

$$\hat{H}(Y) = -\sum_{y=1}^{m} \hat{p}(y)\ log\ \hat{p}(y), \tag{2.25}$$

where we estimate $p(y)$ by $\hat{p}(y) = n_y/n$. The estimation of $H(Y|X_{.j})$ can be obtained from (2.17) when $X_{.j}$ is discrete and from (2.20) when $X_{.j}$ is continuous. For discrete feature variables, estimating the probability densities can be achieved by means of a histogram approximation (Kwak and Choi 2002). We can estimate $\hat{p}(x, y)$ and $\hat{p}(y|x)$ by histogram approximation to get

$$\hat{H}(Y|X_{.j}) = -\sum_{x \in Supp(X_{.j})} \sum_{y=1}^{m} \hat{p}(x, y)\ log\ \hat{p}(y|x), \tag{2.26}$$

For continuous features, entropy estimation is not straightforward due to the problem of estimation of $p(y|x_{.j})$, where y is discrete and $x_{.j}$ is continuous. Note that

we need to estimate the conditional density of $x_{.j}$ at the m classes represented by y and not the joint density. We can use a Parzen window estimation approach to estimate $p(x_{.j})$ (Kwak and Choi 2002) given by

$$\hat{p}(x_{.j}) = \frac{1}{n}\sum_{i=1}^{n}\phi(x_{.j} - x_{ij}, h), \tag{2.27}$$

where $\phi(.)$ is the window function and h is smoothing parameter. Rectangular and Gaussian functions are suitable window functions (Duda et al. 2012) and if h is selected appropriately, $\hat{p}(x_{.j})$ converges to $p(x_{.j})$ (Kwak and Choi 2002). We can calculate $p(x_{.j}|y)$ using the Parzen window approach

$$\hat{p}(x_{.j}|y) = \frac{1}{n_y}\sum_{i\in I_y}\phi(x_{.j} - x_{ij}, h), \tag{2.28}$$

where I_y is the set of observations with class label y. Finally, we the use Bayes rule and (2.27)-(2.28) to estimate $p(y|x_{.j})$ as

$$\hat{p}(y|x_{.j}) = \frac{\hat{p}(x_{.j}|y)\ \hat{p}(y)}{\hat{p}(x_{.j})}\ , \tag{2.29}$$

and then estimate $H(Y|X_{.j})$ from (2.20) by replacing the integral by summation over training observations and using $p(x, y) = p(x)\ p(y|x)$ to arrive at

$$\widehat{H}(Y|X_{.j}) = -\sum_{x_{.j}}\hat{p}(x_{.j})\sum_{y=1}^{n_y}\hat{p}(y|x_{.j})\ log\ \hat{p}(y|x_{.j}). \tag{2.30}$$

Using the Parzen window approach, along with (2.25) and (2.30), we can calculate the Mutual Information from (2.24) between any feature $X_{.j}$ and the class variable Y, which measures the relevance of the feature $X_{.j}$ in classification. Using this, a weight λ_j is assigned to each feature $X_{.j}$ in Mutual Information based KNNI (MI-KNN)

(García-Laencina et al. 2009), such that

$$\lambda_j = \frac{I(X_{.j}; Y)}{\sum_{j'=1}^{p} I(X_{.j'}; Y)} \ , \tag{2.31}$$

and then the distance between instances is calculated similar to (2.7):

$$d_I(X_a, X_b) = \sqrt{\sum_{j=1}^{p} \lambda_j d_j(X_{aj}, X_{bj})^2}, \tag{2.32}$$

where $d_j(X_{aj}, X_{bj})$ is as defined in (2.8). Using this feature relevance weighted distance, replacing d with d_I (from (2.32)) in (2.11)-(2.12), and following Algorithm 1, we obtain the MI-KNN imputation algorithm (García-Laencina et al. 2009).

2.2.4 Grey Relational Analysis (GRA) based KNNI

Grey System Theory (GST) has been developed to tackle systems with partially known and partially missing information (Deng 1982). The system was named grey since missing data is represented by black whereas known data is white, and this system contains both missing and known data. To obtain Grey-based k-nearest neighbors, we used Grey Relational Analysis (GRA) in our algorithm which is calculating Grey Distance between two instances. Grey distance measures similarity of two random instances, which involves the Grey Relational Coefficient (GRC) and the Grey Relational Grade (GRG).

Consider the setup in (2.1) where the training dataset has n observations and p features. The Grey Relational Coefficient (GRC) between two instances/observation X_a and X_b, when the j-th feature is continuous and observed for both instances, is

$$GRC\ (X_{aj}, X_{bj}) = \frac{\Delta_{\min} \ + \ \rho\ \Delta_{\max}}{|X_{aj} - X_{bj}| \ + \ \rho\ \Delta_{\max}} \ , \tag{2.33}$$

where $\Delta_{\min} = \min_c \min_k |X_{ak} - X_{ck}|$, $\Delta_{\max} = \max_c \max_k |X_{ak} - X_{ck}|$, $\rho \in [0, 1]$ (usually $\rho = 0.5$ is taken (Deng 1982)), $b, c \in \{1, 2, ..., n\}$, and, $k, j \in \{1, 2, ..., p\}$ and for categorical feature j, $GRC(X_{aj}, X_{bj})$ is 1 if they have the same values, 0 otherwise. If either X_{aj} or X_{bj} is missing, then $GRC(X_{aj}, X_{bj})$ is 0. The Grey Relational Gradient (GRG) between the instances is defined as:

$$GRG(X_a, X_b) = \frac{1}{p} \sum_{j=1}^{p} GRC(X_{aj}, X_{bj}), \tag{2.34}$$

where $a \in \{1, 2, ..., n\}$. We note that if $GRG(X_a, X_b)$ is larger than $GRG(X_a, X_c)$ then the difference between X_a and X_b is less than the difference between X_a and X_c, which is the opposite of the Heterogenous Euclidean Overlap Metric (HEOM) (2.7) defined in Section 2.2.2.1. The Grey Relational Gradient satisfies the following axioms which makes it a distance metric (Deng 1982):

1. Normality: The value of $GRG(X_a, X_b)$ is between 0 and 1.

2. Dual Symmetry: Given only two observations X_a and X_b in the relational space, then $GRG(X_a, X_b) = GRG(X_b, X_a)$.

3. Wholeness: If 3 or more observations are made in the relational space then $GRG(X_a, X_b)$ is generally not equal to $GRG(X_b, X_a)$ for any b.

4. Approachability: $GRG(X_a, X_b)$ decreases as the difference between X_{aj} and X_{bj} increases, other values in (2.33) and (2.34) remaining constant.

GRA is generally preferred over metrics such as Heterogeneous Euclidean Overlap Metric (HEOM) for grey systems with missing data (Huang and Lee 2004). It gives us a normalized measuring function for both missing/available and categorical/continuous data due to its normality. It also gives whole relational orders

due to its wholeness over the entire relational space. So instead of $d(X_a, X_b)$ in (2.7), if we use $GRG(X_a, X_b)$ to select the k-nearest neighbors and then proceed with the KNN Imputation technique without using weights, then it becomes Grey KNN (GKNN) Imputation (Zhang 2012).

2.2.5 Transformation of the Data

Before we apply our version of the algorithm, we make some transformation of the continuous features contained in the training dataset, since we deal with a wide variety of features whose ranges vary vastly. For example, the range of marks in a 10 point exam would be less than the range of marks for a 100 point exam, and both these marks may be in the same training dataset. The distance metric and subsequently the k-nearest neighbor would be biased unless the ranges of the continuous variables are normalized. In our algorithm, we transformed the j-th feature of observation X_i as

$$X'_{ij} = \frac{X_{ij} - \min_a X_{aj}}{\max_a X_{aj} - \min_a X_{aj}} \,, \tag{2.35}$$

where $a, i \in \{1, 2, ..., n\}$ and $j \in \{1, 2, ..., p\}$. Thus (2.35) ensures all the continous variables are between 0 and 1. Note that the distance metric associated with categorical variables (Euclidean or Grey-based) lie within 0 and 1 as well.

2.2.6 The Proposed Class-weighted Grey k-Nearest Neighbor (CGKNN) Algorithm

We consider the class weight λ_j associated with the j-th attribute $X_{.j}$ and use this to weigh the Grey Relational Gradient (GRG) between two observations X_a and

X_b as follows

$$GRG(X_a, X_b) = \sum_{j=1}^{p} \lambda_j \, GRC(X_{aj}, X_{bj}). \tag{2.36}$$

Since $GRG(X_a, X_b)$ increases for closer neighbors unlike the other distance metrices, we use $d(X_a, X_b) = 1 - GRG(X_a, X_b)$ in section 2.2.2.2 and then measure the distance between instances to choose the k-nearest neighbors, $\{a_\ell\}_{\ell=1}^{k}$. From (2.11), we derive that the corresponding weights of a_ℓ are

$$w_\ell = \frac{1}{(1 - GRG(X_i, a_\ell))^2}. \tag{2.37}$$

Using these weights, we impute the continuous variables, and the new definition of $d(X_a, X_b)$ in (2.12)-(2.13) is used to impute the categorical variables for our Class-weighted Grey KNN (CGKNN) Imputation Algorithm. The algorithm is:

Algorithm 2 Class-weighted Grey k-Nearest Neighbor (CGKNN) Imputation

Input: (X, Y, D) with $X \subset \mathbb{R}^{n \times p}$ containing missing entries and Y the m class labels.

Output: Imputed feature matrix $\widetilde{X}$ with no missing values.

Procedure:

1. **Data pre-processing:** First we transform the continuous features of X as suggested by 2.2.5 using (2.35) so that their ranges equal 1.

2. **Initialization:** We use the class labels in Y to split X into $\{X^y\}_{y=1}^m$. For each class y, given X^y, we pre-impute the missing values of p_0 categorical features by mode imputation and the missing values of the p_1 continuous features by mean imputation using the observed data in that class. We call it $\widetilde{X}^0$.

3. **Mutual Information:** Calculate the mutual information or the class weights λ_j of the attributes $X_{\cdot j}$ using (2.24)-(2.31).

4. **Choosing k:** We use this imputed matrix, $\widetilde{X}^0$, to calculate the optimum value of k using 10-fold cross validation by minimizing the misclassification rate.

5. **Iterative Step:** Consider iteration t (≥ 1) and class y. For each instance i in the class y which has a missing value, calculate the GRG of that instance with all other instances of the class y. We find the k nearest neighbors $\{v_\ell\}_{\ell=1}^k$. Using the weights w_ℓ as described in (2.37), the imputed matrix $\widetilde{X}^{y,t}$ is obtained with $d(X_a, X_b) = 1 - GRG(X_a, X_b)$. This is repeated for each y until all missing values are imputed to obtain $\widetilde{X}^t = \{\widetilde{X}^{y,t}\}_{y=1}^m$. If the stopping criterion is not met, then the iteration on t continues.

6. **Stopping Criterion:** We stop at the d-th iteration when a stopping criteria is met. The stopping criteria we propose is

$$\max_{i,j:D_{ij}=0} |\widetilde{X}_{ij}^d - \widetilde{X}_{ij}^{d-1}| < 10^{-4}, \tag{2.38}$$

2.2.7 Time Complexity of the Algorithm

Consider the setup (2.1) with n observations, p features and m classes. The time complexity for calculating the GRG in the biggest class containing (say) n_j observations is $O(n_j p)$ and the average processing time for sorting the $GRGs$ is $O(n_j \log n_j)$. If we assume d iterations are taken for the algorithm to converge, then the algorithm has a complexity of $O(d\, n_j^2\, p \log n_j)$ to impute an $n_j \times p$ matrix. We do this for m classes and thus the time complexity for imputing an $n \times p$ matrix is $O(md\, n_j^2\, p \log n_j)$. Now, generally $n_j < n$ whenever $m > 1$, which implies $\log n_j < \log n$, and $n_j * m \geq n$ since n_j was the biggest class. This gives rise to the inequality

$$O(md\, n_j^2\, p \log n_j) < O(d\, n^2\, p \log n).$$

We initially calculate the Mutual Information of each attribute with the class variable, which takes $O(n\, p)$ time along with the imputation of the mean/mode which again takes $O(p)$ time and choosing an optimum k which takes $O(10 * n\, p\, r)$ time if we assume r values of k are tested using 10-fold cross-validation. So our total complexity becomes $O(md\, n_j^2\, p \log n_j + np + p + 10npr)$ which can be approximated to $O(md\, n_j^2\, p \log n_j)$ if the value of n_j is large compared to r. We note that this time complexity is similar to Grey KNNI (GKNN) and Feature-Weighted Grey KNNI (FWGKNN) but less than the $O(d\, n^2\, p \log n)$ complexity of Iterative KNNI (IKNN) and the Grey-Based Nearest Neighbor (GBNN) algorithm (Huang and Lee 2004).

2.3 Simulation Studies

In this section we explore the performance of our proposed **Class-weighted Grey KNN (CGKNN)** algorithm in recovering missing entries and improving the classification accuracy, and we report on computational efficiency of the algorithm.

We compare our method with 6 other well-established methods which are as follows:

- **MICE (Multiple Imputation using Chained Equations):** The MICE algorithm developed by Van Buuren and Oudshoorn (Buuren and Oudshoorn 1999) uses multiple imputation assuming that the columns of X are Fully Conditionally Specified (FCS). We assume an imputation model for predicting missing values in each variable, based on the other variables. Generally, for continuous covariates, predictive mean matching is used for imputation. For categorical covariates, logistic regression is used for unordered covariates and proportional odds model for ordered covariates.

- **MissForest:** This is an iterative imputation method based on a random forest developed by Stekhoven and Buhlmann (Stekhoven and Bühlmann 2011). This non-parametric algorithm is basically similar to MICE except that each predictive mode for imputation is random forest for both categorical and continuous variables. This method has an inbuilt imputation error estimate using the out-of-bag error estimate.

- **Iterative k-Nearest Neighbor imputation (IKNN):** In this method, k nearest instances are computed from the instance with a missing value, using Euclidean Distance as metric. Initial imputation is done using mean or mode imputation, followed by a calculation of the weighted mean or mode of the k nearest neighbors for each missing attribute. This process is done iteratively until the matrix is imputed with convergence between successive iteration steps.

- **Mutual Information based k-Nearest Neighbor imputation (MI-KNN):** Mean or mode imputation is used as a preliminary estimate in this approach. We measure the relevance of each feature in the classification problem similar to the approach described in (2.2.3), and uses a weighted

Euclidean distance to measure the distance between instances, with the mutual information being the weights (García-Laencina et al. 2009). All imputed instances and all complete instances are considered to be known information for estimating missing values iteratively. The missing values are then imputed based on the weighted mean or mode of the nearest neighbors.

- **Grey k-Nearest Neighbor imputation (GKNN):** We use mean or mode imputation for an initial imputed matrix. This imputation method uses GRA to calculate the distance between instances and thus calculate k nearest neighbors for missing value imputation (Zhang 2012). The dataset is divided into separate parts based on the class label and imputation method is simultaneously performed on each of them. The imputed values are again weighted mean or mode of the k nearest neighbors, with the distances and weights calculated by GRA.

- **Feature Weighted Grey k-Nearest Neighbor (FWGKNN):** This approach employs Mutual Information (MI) to measure inter-feature relevance in the X matrix. It then uses a weighted version of GRA to find the distance between instances and weighs them by the inter-feature relevance (Pan et al. 2015). The difference between FWGKNN and our CGKNN algorithm is that the mutual information is computed between the class variable Y and the features $X_{.j}$ in our algorithm whereas it is $I(X_{.i}, X_{.j})$ for the FWGKNN algorithm. Our approach is focused towards classification relevance instead of inter-feature relevance.

2.3.1 Performance Measure

We measure the performance of each algorithm according to the following metrics:

- **Root mean square error (RMSE)**: This measures how accurate or precise the imputation algorithm is as follows:

$$RMSE = \sqrt{\frac{1}{np}\sum_{i=1}^{n}\sum_{j=1}^{p}(X_{ij} - \tilde{X}_{ij})^2}, \tag{2.39}$$

  where e_i is the true value, $\tilde{e}_i$ is the imputed value of the missing data, and m denotes the number of missing values.

- **Classification accuracy (CA):** We develop the imputation algorithm to assist in classification. After imputation of the dataset X, it is used in a suitable classification algorithm, whose accuracy is defined intuitively as follows:

$$CA = \frac{1}{n}\sum_{i=1}^{n} I(\tilde{Y}_i = Y_i), \tag{2.40}$$

  where n is the number of observations of X, $\tilde{Y}_i$ is the predicted class value, Y_i is the actual class value and $I(.)$ is the indicator function.

2.3.2 Simulation Scenarios:

2.3.2.1 Missing Completely at Random (MCAR) Example

We use an artificial example to demonstrate the effect of mutual information with the class variable while selecting the k-Nearest Neighbors. We took a separable example with four cubes drawn in a three dimensional space. Fig. 4.7 shows this

artificial problem. Two cubes belong to class 1, and they are centered on $(0, 0, 0)$ and $(-0.2, -0.4, 0.4)$. The remaining two cubes are labeled with the class 2, being centered on $(-0.6, -0.6, 0.5)$ and $(0.4, -0.2, -0.2)$. In all the cubes, the width is equal to 0.20, and they are composed of 100 samples which are uniformly distributed inside the cube. In this problem, the MI values between the three attributes and the target class are computed: 0.69 for x_1, 0.67 for x_2, and 0.38 for x_3.

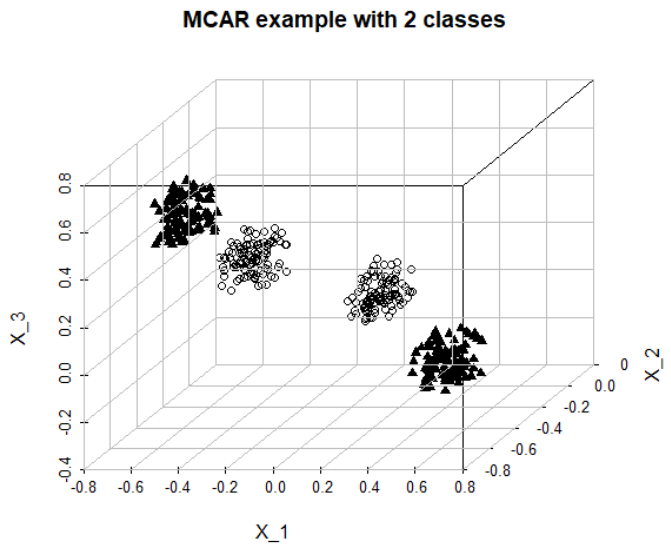

Figure 2.1: 3D Centers of Each Class Represented Without Noise Variables

To this 3 dimensional, 2 class dataset we add 20 U[-1,1] variables. For these irrelevant variables, the MI between the feature and class variable is almost 0. We try to find out what happens when we add irrelevant attributes to classification. We insert 10% and 20% of missing data to x_1, which is most relevant according to MI. The missingness of data in x_1 is generated completely at random, which means it does not depend on the variable values in the matrix X.

This advantage is clearer for higher percentages of missing values, as it is shown by the differences in Table 1. The class weighting procedure based on the MI concept

discards the irrelevant features, and the selected neighborhood for missing data estimation tends to provide reliable values for solving the classification task. We provide a detailed analysis of how all the 6 algorithms performed in this simulation setting with $n = 400$, $p = 23$ and $m = 2$ classes in Table 2.1. Note that we used predictive mean matching as the imputation model for MICE. We also

Table 2.1: RMSE Upon Convergence for the Toy Dataset

Missing Rate	MICE	MissForest	IKNN	MI-KNN	GKNN	FWGKNN	CGKNN
10%	0.2067	**0.0915**	0.2585	0.1023	0.2443	0.1155	0.0983
20%	0.3847	0.1943	0.3746	0.1852	0.3372	0.1803	**0.1598**

We empirically show the convergence of the nearest neighbor-type algorithms by plotting the RMSE against the iterations for the various algorithms in the case where 10% of the data is missing completely at random. The number of iterations plotted in Figure 3.2 is the maximum iterations taken by all of the algorithms to converge.

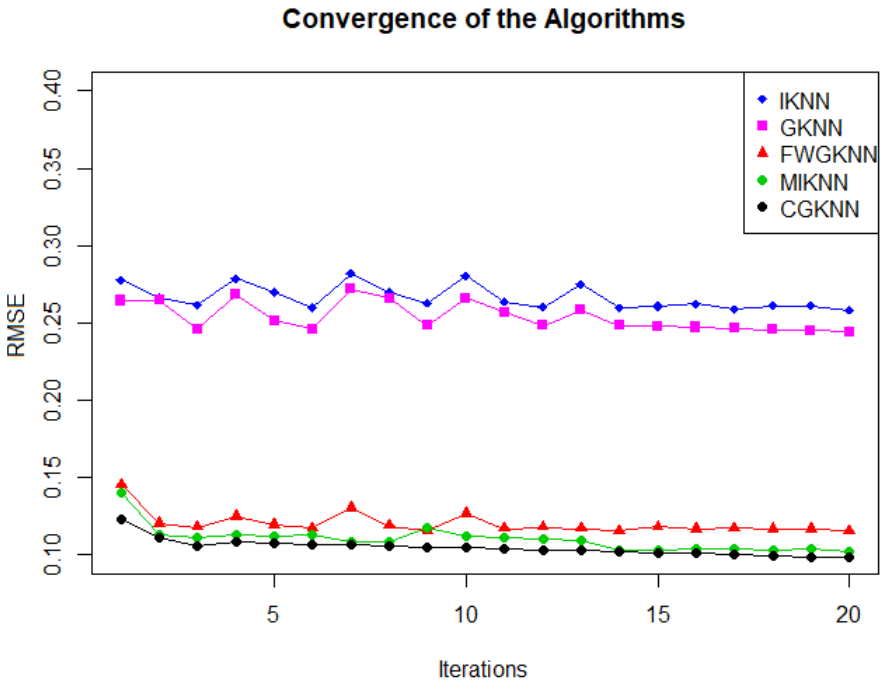

Figure 2.2: Convergence of RMSE for Nearest Neighbors Algorithms at 10% MCAR

We also calculated the classification accuracy using the Naive Bayes method on the non-imputed and imputed datasets with 10% and 20% missing data, with the help of 10-fold cross validation process, using 80% of the data as training data. The resulting improvement in accuracy for both the cases is highest for our CGKNN Algorithm, as shown in Table 2.2.

Table 2.2: Classification Accuracy (%) for the Toy Dataset

Missing Rate	10%	20%
No Imputation	90.44	85.29
MICE	91.91	86.76
MissForest	92.65	91.18
IKNN	89.71	87.82
MI-KNN	92.11	90.24
GKNN	90.20	89.20
FWGKNN	92.19	90.84
CGKNN	**92.92**	**91.93**

2.3.2.2 Missing at Random (MAR) Example

For this section, we illustrate how our method performs with respect to the six other techniques. We simulate our data from the multivariate normal distribution and then artificially introduce missingness in the data, at random (MAR), by letting the probability of missingness depend on the observed values. We take the number of classes $m = 4$, the number of attributes $p = 5$ and generate $n = 100$ observations for each class. Specifically,

$$X_i^{(k)} \sim N(\mu^{(k)}, \Sigma^{(k)}), i = 1, 2, ..., 100, k = 1, ..., 4,$$

where k stands for the $k-$th class, $\mu^{(k)} \sim U[-1, 1]^5 \; \forall \; k$ and $\Sigma^{(k)}$'s are randomly generated $5 * 5$ positive definite matrices using partial correlations (Joe 2006). This simulation procedure ensures us that we do not have the same mean and variance for

two different classes during simulation. Also, the missingness is induced using a logistic model on the missingness matrix D. In real life, we often encounter covariates which are demographic in nature and thus non-missing. For this example, we assume $X_{i1}^{(k)}, X_{i2}^{(k)}$ and $X_{i3}^{(k)}$ to be non-missing and the missingness of $X_{i4}^{(k)}$ and $X_{i5}^{(k)}$ to be dependent on these demographic, non-missing variables, for each class k. Recall the $n * p$ missing matrix D, which we modify to a layered 3D matrix $D^{(k)}, k = 1, .., 4$ with $n * p * m$ entries. We assume $D_{i1}^{(k)}, D_{i2}^{(k)}, D_{i3}^{(k)}$ to be all 1 and

$$D_{i4}^{(k)} \sim Ber(expit(p_{11} + p_{21} * X_{i1}^{(k)} + p_{31} * X_{i2}^{(k)} + p_{41} * X_{i3}^{(k)})), \tag{2.41}$$

$$D_{i5}^{(k)} \sim Ber(expit(p_{12} + p_{22} * X_{i1}^{(k)} + p_{32} * X_{i2}^{(k)} + p_{42} * X_{i3}^{(k)})) \tag{2.42}$$

where $expit(x) = \frac{e^x}{1+e^x}$ and $p'_{ij}s$ are vectors of size $p = 5$ chosen by us.

We provide a detailed analysis of how all the 6 algorithms performed in this simulation setting with $n = 100,\ p = 5$ and $m = 4$ classes in Table 2.3. Note that we used predictive mean matching as the imputation model for MICE. The plot for empirical convergence of the nearest neighbors algorithms are given in figure 2.3 when there is 20% missing data.

Table 2.3: RMSE Upon Convergence for the Toy MAR Dataset

Missing Rate	MICE	MissForest	IKNN	MI-KNN	GKNN	FWGKNN	CGKNN
10%	0.1301	0.0902	0.1407	0.1071	0.1299	0.1229	**0.0887**
20%	0.2084	0.1177	0.1750	0.1423	0.1508	0.1196	**0.1075**

We also calculated the classification accuracy using the Naive Bayes method on the non-imputed and imputed datasets with 10% and 20% missing data, with the help of 10-fold cross validation process. The resulting improvement in accuracy for both the cases is highest for our CGKNN Algorithm, as shown in Table 2.4.

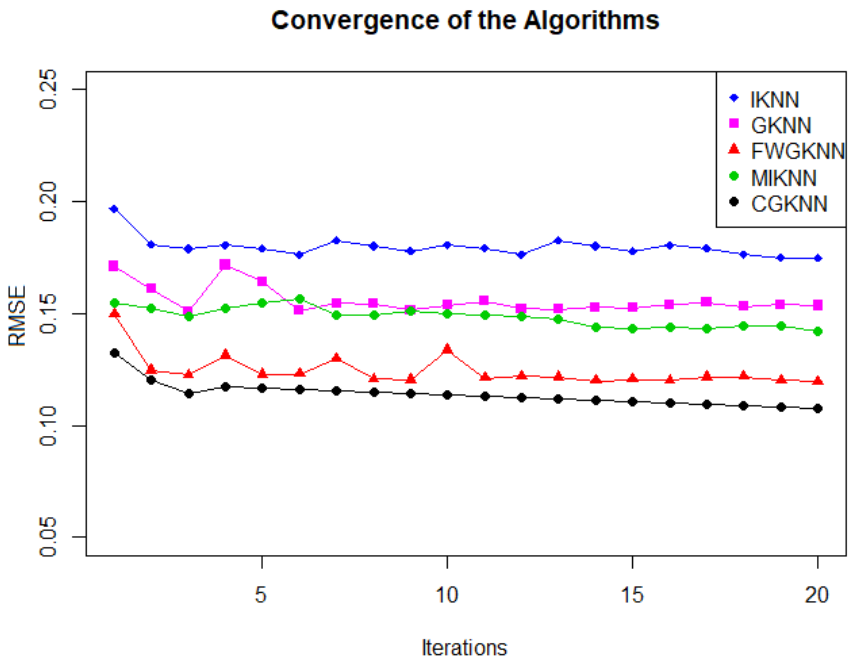

Figure 2.3: Convergence of RMSE for Nearest Neighbors Algorithms at 20% MAR

Table 2.4: Classification Accuracy (%) for MAR Dataset

Missing Rate	10%	20%
No Imputation	75.13	72.50
MICE	79.09	77.32
MissForest	83.75	80.11
IKNN	85.48	83.91
MI-KNN	88.26	86.37
GKNN	86.38	84.12
FWGKNN	89.01	87.30
CGKNN	**90.29**	**87.31**

2.4 Applications to UCI Machine Learning Repository Datasets

We evaluate the effectiveness of our imputation algorithm on 3 datasets obtained from UCI Machine Learning Repository (Newman et al. 2008), the Iris (Fisher's Iris Dataset), Voting and Hepatitis datasets, having respectively, characteristics mentioned in Table 2.5.

Table 2.5: Characteristics of the UCI Datasets Used for Data Analysis

Dataset	Instances	Features	Classes	Feature type	% Missing Rate
Iris	150	4	3	Continuous	0
Voting	435	15	2	Categorical	4.14
Hepatitis	155	19	2	Mixed (both)	5.39

We represent the Mutual Information (MI) of each feature in these datasets in the graphs shown in Fig. 2.4 - 2.6, and use it as the weights for our CGKNN algorithm.

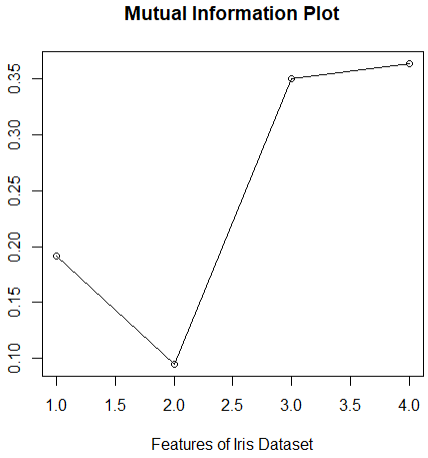

Figure 2.4: MI for the Sepal and Petal Lengths and Widths of Iris Dataset

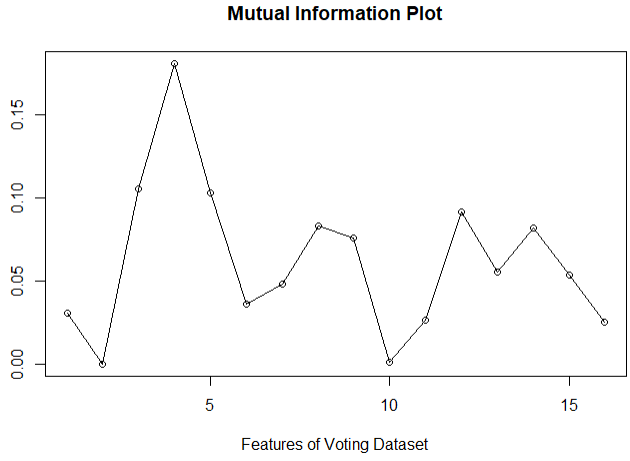

Figure 2.5: MI for Voting Dataset

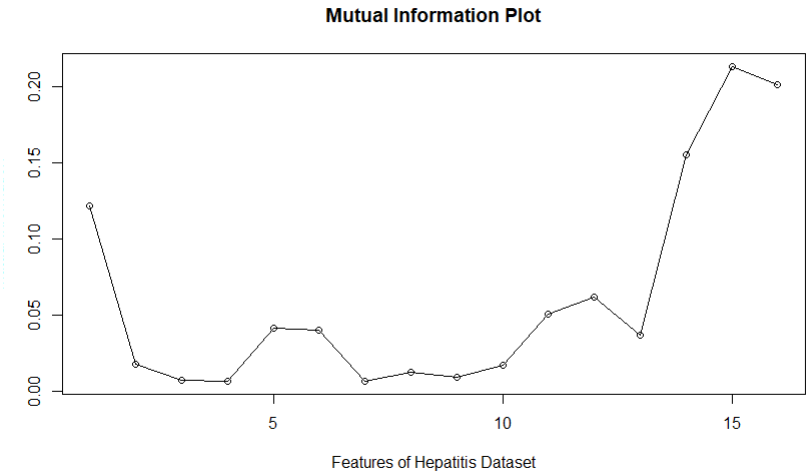

Figure 2.6: MI for Hepatitis Dataset

We then introduce 3 different rates of artificial missingness at random (MAR) - 5%, 10% and 20%. Then we run each of the imputation algorithms and calculate the RMSE of imputation after each algorithm converges. For MICE, we used predictive mean matching for continuous variables and polytomous logistic regression for categorical variables. Looking at Table 2.6 - Table 2.8 note that in almost all cases, our algorithm CGKNN performs better than the other algorithms, usually at higher percentages of missing values. MICE performs the worst in most cases, followed by MissForest, probably because they do not take into account any sort of feature relevance.

Table 2.6: Comparison of RMSE of Iris Dataset

Missing Rate	MICE	MissForest	IKNN	MI-KNN	GKNN	FWGKNN	CGKNN
5%	0.0729	0.0619	0.0588	**0.0503**	0.0534	0.0506	0.0509
10%	0.1205	0.1302	0.1107	0.1025	0.1038	0.0995	**0.0950**
20%	0.1427	0.1420	0.1241	0.1146	0.1246	0.1106	**0.1001**

We use a Naive Bayes classifier on the Iris dataset with 5% - 20% missingness and see that our CGKNN algorithm outperforms the closest approach FWGKNN and also GKNN, when used as an imputation approach before the classifier. The CGKNN

Table 2.7: Comparison of RMSE of Voting Dataset

Missing Rate	MICE	MissForest	IKNN	MI-KNN	GKNN	FWGKNN	CGKNN
5%	0.0928	0.0919	0.0874	0.0791	0.0820	**0.0770**	0.0779
10%	0.1029	0.1002	0.0949	0.0870	0.0929	0.0868	**0.0827**
20%	0.1521	0.1601	0.1574	0.1446	0.1099	0.1088	**0.1049**

Table 2.8: Comparison of RMSE of Hepatitis Dataset

Missing Rate	MICE	MissForest	IKNN	MI-KNN	GKNN	FWGKNN	CGKNN
5%	0.0913	0.0890	0.0785	**0.0711**	0.0792	0.0739	0.0714
10%	0.1029	0.1002	0.1107	0.0870	0.1038	0.0994	**0.0921**
20%	0.1967	0.1858	0.1592	0.0839	0.0980	0.0898	**0.0823**

algorithm also converges quite fast with respect to classification accuracy as shown in Fig. 2.7.

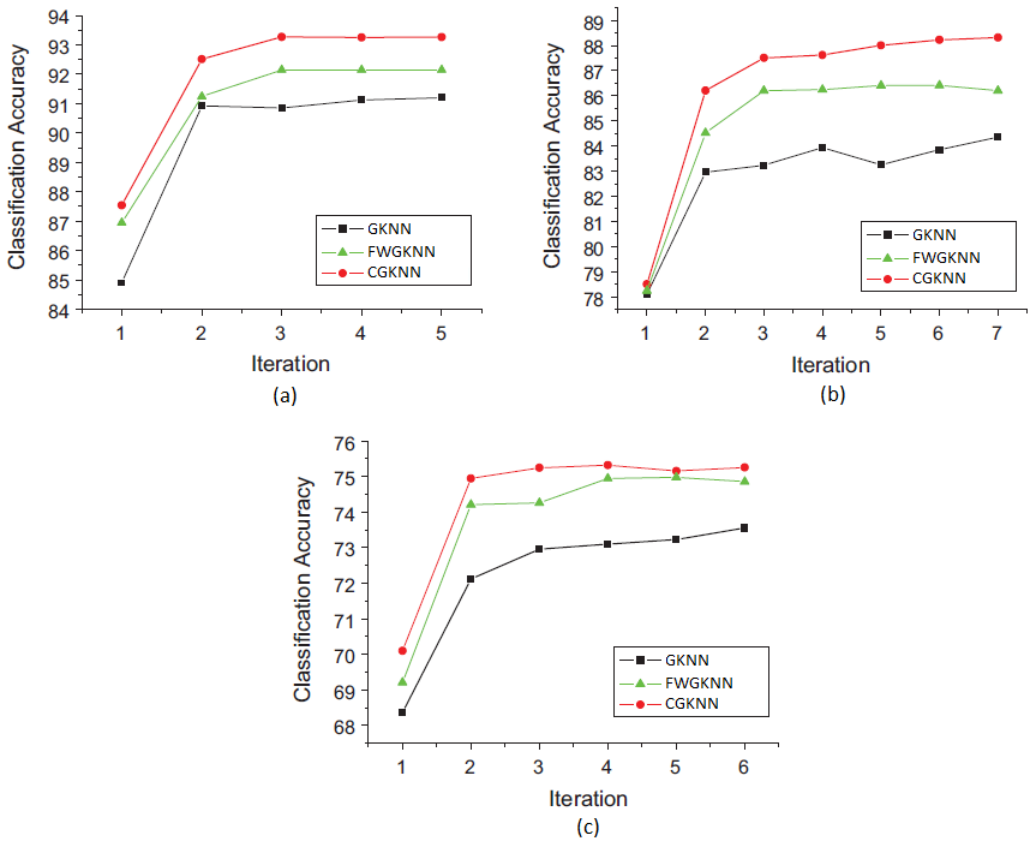

Figure 2.7: Classification Accuracy for the Iris Dataset at (a) 5% (b) 10% and (c) 20% Rates of Missingness After Using NN Imputation

2.5 Discussion

Missing data is a classical drawback for most classification algorithms. However, most of the missing data imputation techniques have been developed without taking into account the class information, which is always present for a supervised machine learning problem. k-Nearest Neighbors is a good technique for imputation of missing data and has shown to perform well against many other imputation procedures. We have proposed a method which not only takes into account the class information, but also uses a better metric to calculate the nearest neighbors in KNN imputation. Our Class-weighted Grey k-Nearest Neighbor (CGKNN) approach has same time complexity as the previous algorithms and even better than some KNN imputation algorithms like Grey-Based k-Nearest Neighbor (GBNN) and Iterative k-Nearest Neighbor (IKNN) imputation. We have shown that it outperforms all the other algorithms in simulated settings, as well as high rates of missingness in actual (non-simulated) datasets as far as imputation is concerned. We also show that it improves the accuracy of classification better than other imputation procedures. We do not make any assumptions regarding the variables of the feature matrix and thus, for any classification problem, our method can be used to impute missing data in the feature matrix.

However, an open problem is the selection of k in our nearest neighbors approach and we have chosen it through cross-validation and this method takes time. The reason why k is difficult to predict is because we do not have anything to validate the true value of k in our datasets. A potential future research could be to select the value of k in a smart, effective manner without involving cross-validation. Our algorithm has not been theoretically proven to converge, although it has been shown empirically. Finding the rate of convergence of our CGKNN algorithm is a good

theoretical problem to consider.

Another potentially interesting future research topic would be to extend this idea to regression problem where the outcome Y is continuous instead of categorical. The imputation of the data matrix X could be done with the help of information from Y since they are assumed to be related in a regression setting. We could also look into better methods of measuring the relationship between the features and class variable than mutual information (MI) and then use them as weights for the Grey distance. Another potential future research paper is to develop an algorithm which imputes and classifies simultaneously, thus yielding a better classification in a single step instead of imputation and classification at two different stages. This idea has already been worked on in Learning Vector Quantization (LVQ) (Villmann et al. 2006) but can be vastly improved.

The most difficult challenge, however, to find imputation techniques when the data is Not Missing at Random (NMAR). It is difficult to model this setting without making strong assumptions, and much development is still possible in that area. The main difficulty is to tackle the problem without assuming anything that may cause a bias - and that is not possible. Hopefully, new ideas will crop up in the future which will make NMAR problem easier to handle.

CHAPTER 3: RANDOM FOREST IMPUTATION OF MISSING COVARIATES FOR LONGITUDINAL DATA MODELS

3.1 Introduction

Longitudinal studies are prone to huge losses or missingness in terms of the number of participants due to follow up. Particularly in older populations, the reason for this loss to follow up is death. The other reasons may be serious like loss of mobility, shifting to a different part of the country, loss of memory or may be trivial like non-compliance. For all reasons other than death, the participants may be called up to partially recover the data, without a proper physical visit. An example is the Atherosclerosis Risk in Communities (ARIC) Study Cohort where there are 15,792 participants who took part in the 1st visit back in 1987-1989 across 4 states in the USA (ARIC investigators 1989) and only 6,538 people returned for the 5th visit in 2011-2013, which represents just 41.7 % of the original number of participants. For these 6,538 participants, diabetes was an outcome which was universally measured and a lot of other covariates were taken like blood pressure, glucose levels, etc. But these covariates had some missingness due to non-response (during the visit), storage problems or fault of the data collector. For the remaining 9,254 participants, those who were alive were called up to partially collect data on covariates for which physical visit was not needed. Diabetes status was ascertained for some of these 9,254 patients lost to follow-up, along with a few covariate values. However, most covariate values could not be recorded.

We are interested in dealing with longitudinal data where the outcome of interest is measured at each timepoint, but covariate values are missing at various timepoints

for some of the participants due to lack of a physical visit or other data recording problems. As shown in many studies (Enders 2010), ignoring observations with missing values and going ahead with the inference problem to be addressed leads to a loss of power and information. Some imputation problems are only for continuous data (Aittokallio 2009) whereas many parametric imputation techniques have a high computational complexity (Liao et al. 2014). Most of the parametric techniques have a bias associated with it if the model is misspecified. To ease the restrictions, fully conditional specification of the covariates had been used, but it is not easy to specify for complicated interactions (Bartlett et al. 2015).

Due to the complexities of dealing with imputation parametrically, non-parametric techniques of imputation has been primarily developed in recent years, for longitudinal studies. Machine learning methods in particular have been explored among non-parametric methods. Not much research has been done into imputing missing data using random forests for a longitudinal study model. Most of the erstwhile research on missing data imputation for longitudinal models has been parametric or very heuristic in nature like maximum likelihood based parametric methods and EM based algorithms (Ibrahim and Molenberghs 2009), hot and cold deck imputation (Spratt et al. 2010) and predictive mean matching (PMM) and a combination of logistic at item level and multivariate normal regression at class levels (Nooraee et al. 2018). An application of missing data imputation for longitudinal datasets in the ARIC study was done for the lung function capacities of black and white people of the cohort over visits 1 and 5 (Mirabelli et al. 2016), and the method for multiple missing data imputation used was inverse probability weighting (IPW) on the condition-of-being-alive method that was used to estimate the lung function decline and to construct weights that depend on the probability of dropout among those participants who are still alive, while removing the possibility at risk for

dropout those who died before being examined at the next ARIC visit.

The first research done in machine learning-based imputation of longitudinal data was using Partial Logistic Artificial Neural Network (PLANN) regularised within the evidence-based framework with Automatic Relevance Determination (ARD), together known as PLANN-ARD (Fernandes et al. 2008). For the simplicity and robustness of non-parametric methods, machine learning methods have been often preferred in missing data imputation. A promising approach can be random forests, developed first by Tin Kam Ho and then worked on by Breiman (Breiman 2001) and patented on by Adele Cutler (3185828).

Random forests have a huge advantage over traditional machine learning algorithms because it is comparatively faster to train, addresses complicated non-linear interactions, handles mixed data type and can easily scale to high dimensions without over-fitting. Random forest as an imputation technique has been studied before. The performance of the imputation technique is given by the Out-of-Bag (OOB) error of the RF models used in imputation. The first stride in random forest imputation was using the proximity algorithm by the R-package randomForest developed from Breiman's original idea (Liaw et al. 2002). The next approach in this area was using "on-the-fly imputation" algorithm which involves growing a survival tree simultaneously while imputing the missing values in the dataset (Ishwaran and Lu 2008). The third approach involves unsupervised learning in the imputation problem, where each covariate is treated as a dependent variable which is the outcome of a random forest grown from the other covariates. This approach is called missForest (Stekhoven and Bühlmann 2011) and is quite popular since it has been shown to outperform MICE and IKNN (Iterative k-Nearest Neighbors) algorithms.

We propose to use the various types of random forest imputation techniques by

using random splits while growing the tree to increase the computational speed of the methods (Tang and Ishwaran 2017). We apply various random forest imputation methods to simulated longitudinal datasets with datapoints missing both completely at random (MCAR) and at random (MAR). We compare the performance with traditional non-parametric imputation techniques like MICE (Buuren and Oudshoorn 1999) and iterative kNN (IKNN) (Troyanskaya et al. 2001).

The remainder of this chapter is organized as follows. In Section 2, we formally state the problem of missingness in longitudinal data and then propose the various imputation techniques using random forests. We shall show the variant of random forest imputation which we propose for longitudinal data. In Section 3, we test our proposed imputation methods, including their randomized versions, against MICE and Iterative kNN (IKNN) imputation in various simulation settings with data missing completely at random and at random. We check how our method performed against all the other methods. We conclude with a discussion of application to the ARIC dataset and scope for future work in Section 4.

3.2 Methodology

3.2.1 Formulation of the Problem

As stated before, we deal with longitudinal datasets where the response is observed or recorded for each timepoint either by a physical visit or phone call. Due to various reasons, the covariate values may be missing partially or wholly for a few participants at certain timepoints. Let $X = \{X_{ijk}\}_{i=1}^{n}$ be the $n \times t \times p$ data matrix where X_{ijk} represents the k-th covariate of the i-th subject/individual at time point j. We assume there are n individuals, p covariates and t time points. Similarly, $Y = \{Y_{ij}\}_{i=1}^{n}$ is the $n \times t$ response matrix where Y_{ij} represents the i-th subject

response at time point j. Y can be a continuous or categorical variable observed across t timepoints. There are n subjects for t time points. We consider the longitudinal data $\{(X_{ijk}, Y_{ij})\}_{i=1}^{n}$ with n subjects. We assume the following model to be satisfied by the data

$$E(Y_{ij}|X_{ij}) = f(X_{ij}), \quad i = \{1, 2, ..., n\}, j = \{1, 2, ..., t\}, \tag{3.1}$$

where $f(.)$ is a continuous or categorical real-valued function, $X_{ij} = \{X_{ijk}\}_{k=1}^{p}$ has missing entries and Y is completely observed. Now, there may be demographic covariates in X which do not change with time and are recorded for each of the n subjects in the initial visit. We still make t copies of these demographic variables so that the data matrix X has a dimension of $n \times t \times p$ and is easier for notation - but it will be treated differently during imputation as we will soon explain.

We denote the missing entries in the data matrix X by the indicator matrix $D = \{D_{ijk}\}_{i=1}^{n}$ with same dimensionality as X and

$$D_{ijk} = \begin{cases} 0, & X_{ijk} \text{ is missing} \\ 1, & \text{otherwise.} \end{cases} \tag{3.2}$$

We assume the data matrix contains mixed-type data. Without loss of generality, we can assume for each (i, j) pair, the subject $X_{ij} = \{X_{ijk}\}_{k=1}^{p}$ contains p_0 categorical features for $j \in \{1, 2, ..., p_0\}$ and p_1 continuous features for $j \in \{p_0 + 1, ..., p_0 + p_1\}$ such that $p_0 + p_1 = p$. Let the j-th categorical feature contain k_j different values and the j-th continuous variable representing the $(p_0 + j)$-th feature of X_i, indexed by $j \in \{1, ..., p_1\}$ take values from a continuous set $C_j \subset \mathbb{R}$. For each of the categorical features, we can map the k_j different values to the first k_j natural numbers, such that $X_{ij} \in \{1, ..., k_1\} \times ... \times \{1, ..., k_{p_0}\} \times C_1 \times ... \times C_{p_1} \subset \mathbb{R}^p$.

We assume that the data is missing at random (MAR), that is, the distribution of D does not depend on the missing values in X but rather the observed values,

$$P(D|X) = P(D|X_{obs}), \quad \text{for all } X_{mis} \tag{3.3}$$

We propose tree-based approaches to impute the $n \times t \times p$ data matrix X, since it will have complicated interactions between the p dimensional covariates $\{X_{ij}\}, i = 1, 2, ..., n, j = 1, 2, ..., t.$. Decision trees divide the feature matrix intuitively into classes and assigns a predicted value to each partition (Friedman et al. 2001). Classification and Regression Trees (CART) have been built in 1984 (Breiman et al. 1984) as a reproducible method to use decision trees to model the outcome variable. We then move on to random forest (RF), first introduced by Tin Kam Ho (Ho 1995), and later developed by Leo Breiman (Breiman 2001) and Adele Cutler who registered "Random Forests" as a trademark (3185828), which aggregates several trees that make decisions after some random selection at each internal tree node.

3.2.2 Classification and Regression Trees (CART)

In this section, we generally state how Classification and Regression Trees (CART) are used for longitudinal data setting. As stated previously, the time-varying response matrix Y has $n \times t$ entries and the time-varying as well as demographic covariates (replicated t times) are recorded in the data matrix X with dimension $n \times t \times p$. Both Y and the p two dimensional matrices X_{ij}, representing the entries of the p covariates can be either continuous or categorical, which means X is a mixed dataset. Let us consider the $k-$th feature of the dataset X filled with a series of binary splits, each of which can be written as $I\{X_{..k} \leq c\}$. When the split is made to form the tree, subjects in timepoints with $\{X_{..k} \leq c\}$ fall into one area of the split and

those with $\{X_{..k} > c\}$ fall into the other area made by the split. A splitting point is called an internal node. The terminal nodes are the final end points.

We can define the regions in the partitioned data matrix X as $R_m, m = 1, ..., M$, where M is the number of terminal nodes. Let the average outcome of the terminal node m be written as

$$\widehat{c}_m = \sum_{i=1}^{n} \sum_{j=1}^{t} \frac{Y_{ij} I\{X_{ij.} \in R_m\}}{\sum_{a=1}^{n} \sum_{b=1}^{t} I\{X_{ab.} \in R_m\}},$$

then we may estimate $f(X_{ij}) = E[Y_{ij}|X_{ij}]$ with

$$\widehat{f}(X_{ij}) = \sum_{m=1}^{M} \widehat{c}_m I\{X_{ij.} \in R_m\}.$$

So the predicted value of a new data point is equal to the average of observed outcomes across all training set data points, over all different timepoints, in the same terminal node as the new data point.

In the greedy approach to build a tree, all the p variables and their possible cut points are considered at each node, and we move forward with the cut point which gives a maximum gain in terms of a suitable criteria. To choose the variable and cut point for each node, minimization of the sum of squares is performed:

$$\min_{k,c} \left\{ \min_{a} \sum_{i,j:X_{ij.} \in R_l(k,c)} (Y_{ij} - a)^2 + \min_{b} \sum_{i,j:X_{ij.} \in R_g(k,c)} (Y_{ij} - b)^2 \right\},$$

where $R_l(k, c) = \{X | X_{..k} \leq c\}$ and $R_g(k, c) = \{X | X_{..k} > c\}$. Note that a and b are estimated by the average outcomes in R_l and R_g, respectively. For missing data problems, when $X_{..k}$ is not observed, it is left out of the calculation of split points. Further insights into how the trees are grown for various imputation techniques are

given in section 3.2.4. The stopping criteria for tree growth is a minimum (terminal) node size. In other words, a terminal node must have at least the minimum node size. When all terminal nodes can no longer be split, growing the tree is completed, and we call this the maximal tree T_0.

To decrease over-fitting, cost-complexity pruning is then used to cut back the maximal tree. Define T to be a subtree of T_0, and $|T|$ to be the total number of nodes in the tree T. A sequence of subtrees is constructed by sequentially collapsing non-terminal nodes that lead to the smallest value of

$$\frac{1}{M}\sum_{m=1}^{M}\sum_{i,j:X_{ij.}\in R_m}(Y_{ij}-\widehat{c}_m)^2.$$

The tree within this sequence that minimizes the cost complexity criterion,

$$C_\alpha(T)=\sum_{m=1}^{M}\sum_{i,j:X_{ij.}\in R_m}(Y_{ij}-\widehat{c}_m)^2+\alpha M,$$

is then selected as the final tree. α may be chosen by minimizing the cross-validated sum of squares.

The above concept of regression may also be extended to classification (since the name is Classification and Regression Trees) in longitudinal studies where the splitting criteria are based on node impurity measures such as misclassification error or Gini index. The predicted class of a data point in a classification problem with K distinct classes is

$$\underset{k\in 1:K}{\arg\max}\ \{\widehat{p}_{mk}\},$$

where $\widehat{p}_{mk}$ is the proportion of observations in terminal node m with the class k. In this entire section describing CART, only single trees are used which may have high bias - the solution to which is using a combination of trees called random forest. For

growing the tree in a general CART setting, only complete cases are chosen. For missing data cases, the trees are grown after pre-imputation or out-of-bag imputation as described in Section 3.2.4.

3.2.3 Random Forests

To improve the instability of CART and produce better predictions, Breiman later developed random forests (RF) (Breiman 2001) after it was introduced by Tin Kam Ho (Ho 1995), as a combination of bagging with CART. In RF, B bootstrapped samples are drawn from the data, and a tree is grown for each sample. Growing a tree in RF differs from growing a tree from CART in that not all p variables are considered in a greedy approach at each node. Instead, $q \leq p$ variables are randomly selected and evaluated as the potential split variable. Once the forest has been grown, we may then retrieve an estimate from each tree: $\widehat{f}^b(X_{ij})$, for $b = 1, \ldots, B$, and a bootstrapped estimate is $\widehat{f}(X_{ij}) = \frac{1}{B}\sum_{b=1}^{B} \widehat{f}^b(X_{ij})$. For classification, a majority vote is taken across the trees for the predicted class of a new data point.

Additionally, trees in RF are not pruned. Because RF incorporates bootstrapping and the variables are selected at random, over-fitting is mitigated, and pruning is not necessary. The random variable selection reduces the correlation between each of the trees, thereby reducing the variance in the average across trees. This isn't to say RF avoids overfitting. Let p^* be the number of relevant variables such that $p^* << p$. Especially with small q, it becomes unlikely that the relevant variables are sampled at each node. However, with q large enough, RF still performs quite well. It is typically recommended to use $q = \lfloor p/3 \rfloor$.

The default parameter for the number of trees to grow in the R package randomForest (Breiman 2001) is 500. However, the growing of trees can be stopped

once the out of bag (OOB) error reaches a desired level of convergence. This error is defined by averaging the prediction error for each subject among all trees for which the subject was not apart of the bootstrapped sample. The OOB error is similar to the error estimated from n-fold cross-validation, making random forest a convenient method in that it is able self-evaluate as trees are grown.

Variable importance (VI) measures can be calculated one of two ways with RF. The first uses the Gini index: the improvement in the split-criterion provided by splitting on the jth variable is cumulatively summed across all trees. Alternatively, OOB prediction errors are compared before and after randomly permuting values for the jth variable. The increase in prediction error is averaged across all trees, and that average is considered as the variable importance. This feature of RF allows us to observe which variables drove construction of the forest the most.

Similar to CART, RF can still make bad splits near the start of the tree, leading to an overall poor fit after later splits. The bagging mechanism of RF reduces this error by averaging across many de-correlated trees. By using the tree building criteria described in Section 3.2.2, we can grow RF in longitudinal data and use that for building a predictive model to estimate (3.1).

3.2.4 Random Forest Imputation

We now look into this section as how we can extend the idea of building a random forest for longitudinal datasets to imputation of the covariates (both time-varying and demographic) in the dataset X using random forest models built with the help of response matrix Y which is fully observed. A basic imputation algorithm for any longitudinal dataset is called "Strawman imputation" which is the initial imputation of any dataset with missing values. It is a quick way to impute by

replacing continuous covariate values with the mean among observed values (over all the timepoints) and categorical covariate values with mode among all observed values at all timpoints, with ties broken at random. A basic study of these random forest algorithms in cross-sectional data has been performed (Tang and Ishwaran 2017). We look into three basic imputation algorithms using RFs in longitudinal data, where all the algorithms are iterated until they converge:

(A) Proximity Imputation (RF_{prx}): We preimpute the data, grow the forest and impute the missing values using a proximity matrix described later.

(B) On-the-fly Imputation (RF_{otf}): We simultaneously impute data while growing the forest and then impute again at the end when the forest is built.

(C) Unsupervised or missForest Imputation (RF_{unsv}): We preimpute the data and grow a forest for each covariate with a missing value, using the other covariates as explanatory variables. We use each of these grown forests to predict the missing values of the covariates.

3.2.4.1 Proximity Imputation ($R_{prx}/R_{prx.R}$)

In proximity imputation (Liaw et al. 2002), we first preimpute the longitudinal study dataset X using Strawman imputation and Y is complete (as per our study setting). We fit a random forest model on this imputed data $\{X_{ijk}\}_{k=1}^{p}$, using the response Y according to the method described in the previous sections. From the random forest model, we get a proximity matrix which has dimensions $nt \times nt$ for each of the $n \times t$ replicates of a covariate. The $(i, j)-$th entry of this matrix records how many times observation i was present in the same terminal node of a tree as observation j in the bootstrap samples considered for building B trees in the random forest. Using the proximity entries of each observation with the other observations as weights, when an observation is missing, the weighted mean of that feature over all

the (originally) non-missing observations are taken if the feature is continuous and the weighted mode is taken if the feature is categorical. This newly imputed dataset is used to grow a new RF on the longitudinal data and a new proximity matrix is calculated. This procedure is iterated until convergence. For the demographic covariates, instead of considering all the non-missing observations in the $nt \times nt$ entries, we can just look at the average of the non-missing observations in the first $n \times n$ entries because the other entries are just a replicate of this sub-matrix. We refer to this proximity imputation as R_{prx}.

A modification of this method is random splitting, which is different from non-random or deterministic splitting where all possible split points are checked for the splitting variables, say X_{split}. In random splitting, a maximum of $m_{split}(\geq 0)$ random split points are checked for each of the variables in X_{split} and the best split is chosen based on the splitting criteria. Each tree of the random forest model is grown in this way then the proximity matrix is calculated for imputation, just like R_{prx}. This is different from deterministic splitting since the split points are chosen randomly in the covariate space. The advantage of random splitting is that it is much faster than deterministic splitting. We denoted this randomized split version of proximity imputation as $R_{prx.R}$.

3.2.4.2 On-the-fly Imputation (OTFI, $R_{otf}/R_{otf.R}$)

The disadvantage of R_{prx} is that the prediction would be biased since we consider the non-missing observations only, while building the tree. A remedy for this would be to impute simultaneously while building the tree. On-the-fly imputation (Ishwaran and Lu 2008) was developed in Survival Forests and it can be extended to longitudinal data. The steps to simultaneously impute and grow the tree are called random forest on-the-fly imputation or R_{otf}. We start by considering only

non-missing observations in X to calculate the split statistic at various split points, while growing a tree. Let us say $X_{..k}$ is the covariate chosen for splitting to create two nodes. Now each of the $n \times t$ observations in data matrix X must be put in one of the nodes. For a particular observation, say ab-th observation, if X_{abk} is missing, then it is imputed temporarily by a random value of the non-missing observations of $X_{..k}$, which are in the bag (among the bootstrap samples). This imputed value is used to decide which side of the split the observation X_{abk} will be placed. When all the $n \times t$ observations of X are split, the imputed data are reset to missing. This process is repeated until the tree is fully grown in the bootstrap sample. Now we have a fully grown tree whose missing values have been temporarily imputed during growth. After the trees are created in the random forest, the originally missing values at all the terminal nodes are reset to missing. They are imputed by the mean or mode (if their corresponding feature is continuous or categorical) of the terminal node data from the out-of-bag observations whose values for $X_{..k}$ are non-missing.

Note that while growing the tree, the imputed value in X_{abk} is only used to decide which node the observation $X_{ab.}$ will be placed in and not to calculate the split statistic. All split statistics are calculated from the non-missing values only. This entire process is iterated until the imputed matrices of two simultaneous iterations are close with respect to Root Mean Square Error (RMSE). Just like the previous algorithm, we use the random splitting method to calculate the best split out of m_{split} random split points of the splitting variables X_{split}. This faster version of the out-of-bag imputation algorithm is called $R_{otf.R}$.

3.2.4.3 Unsupervised or missForest Imputation ($R_{unsv}/R_{unsv.R}$)

We notice that both R_{prx} and R_{otf} require the response matrix Y to grow the random forest and thus they are supervised methods to build the random forest. The

unsupervised random forest imputation technique is based on the missForest imputation technique (Stekhoven and Bühlmann 2011). Here we take each of the p features of the data matrix X at t time points with n entries, and perform Strawman's imputation to get an initially imputed matrix, where the imputation is done for each feature at each time point (so, $t \times p$ means or modes are computed). For a time-varying f-th feature at a time point o, say $X_{.of}$, let us say a few observations are missing out of the n subjects. We grow a random forest model with $X_{.of}$ as the response and all other features in the same time point o, $\{X_{.ok}\}_{k \neq f}$, and the same $f-$th feature at all other timepoints, $\{X_{.jf}\}_{j \neq o}$ as the independent explanatory variables. So, in essence, we choose $(t-1)+(p-1)$ columns as explanatory variables to grow a random forest and impute the missing values of $X_{.of}$ from the prediction of the random forest model thus grown.

For the demographic covariates, say $d-$th feature $X_{..d}$, we just need to impute at the initial timepoint. So, we impute using the other covariates at timepoint 1 as the explanatory variable, $\{X_{.1k}\}_{k \neq d}$. These $(p-1)$ columns are selected as explanatory variables to grow a random forest with $\{X_{.1d}\}$ as the response. The missing values are predicted from the random forest model thus grown. After all the time-varying and demographic covariates are imputed, we get an imputed matrix, which is again used iteratively to grow $t \times p$ different random forests, until convergence. This process is called unsupervised random forest imputation RF_{unsv} since the response variable Y is not required. A similar random split version as the previous algorithms, where each time the random forest models are grown using m_{split} random split points for each of the $t \times p$ random forests. This version is much faster and termed as randomized unsupervised random forest imputation $RF_{unsv.R}$.

3.3 Simulation Studies

In this section we explore the performance of our 3 random forest based algorithms, RF_{unsv}, RF_{otf}, and RF_{prx} and their randomized versions against MICE and IkNN imputation.

- **MICE (Multiple Imputation using Chained Equations):** The MICE algorithm developed by Van Buuren and Oudshoorn (Buuren and Oudshoorn 1999) uses multiple imputation assuming Fully Conditional Specification (FCS) of the variables of X. This means that the conditional distribution of each variable given other variables is known. We assume an imputation model for each variable, based on the other variables. We use predictive mean matching in general for continuous variables and various forms of logistic regression for categorical variables. For multiple unordered valued categorical variables, we use polytomous logistic regression. We use proportional odds model for ordered categorical variables.

- **Iterative k-Nearest Neighbor imputation (IKNN):** In this method, mean or mode imputation is done as preliminary imputed values for the corresponding type of variable. This is followed by calculating k closest instances to the instance with a missing value, through Euclidean distance as distance metric. The original missing values are imputed using weighted mean or mode of the attribute values of the k nearest neighbors, using reciprocal of the squared distance from the original observation as weights for continuous case and a function of the scaled similarity metric as weights for mode in the categorical case.

We measure the performance by accuracy of prediction using Root Mean Square Error (RMSE) as follows:

$$RMSE = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(e_i - \tilde{e}_i)^2}, \tag{3.4}$$

where e_i is the true value, $\tilde{e}_i$ is the imputed value of the missing data, and m denotes the number of missing values. We consider the following longitudinal models for our simulation experiments:

1. Fixed Effects Model: $Y_{ij} = X'_{ij}\beta + \epsilon_{ij}$, where Y_{ij} represents the i-th subject at time point j. X'_{ij} and β are p-dimensional, non-stochastic variables, $\epsilon'_{ij}s$ are normal with mean 0 and variance structure such that they are independent when j is fixed but dependent when i is fixed and time-point j is varied. Y being the sum of a stochastic and a combination of non-stochastic variables, has the same distribution as ϵ.

2. Mixed Effects Model: $Y_{ij} = X'_{ij}\beta + X'_{ij}b + \epsilon_{ij}$, where b is the random effect in the mixed effects model which is normally distributed with mean 0 and variance Σ (a $p \times p$ matrix).

We take the number of time-points $t = 4$, the number of time-varying covariates/attributes $p = 5$ and generate $n = 100$ subjects for each time-point. Specifically we generate a p dimensional vector,

$$X_{ij} \sim N(\mu_j, \Sigma_j), i = 1, 2, ..., 100, j = 1, ..., 4,$$

where j stands for the $j-$th time-point, $\mu_j \sim U[-1, 1]^5 \; \forall \; j$ and Σ_j's are randomly generated $5 * 5$ positive definite matrices using partial correlations (Joe 2006). This

simulation procedure ensures us that we do not have the same mean and variance for two different time-points during simulation. We also generate the variance Σ of b from the same randomly generated $5 * 5$ positive definite matrices using partial correlations (Joe 2006). For each simulation setting, β was fixed and generated from $U[-1, 1]^5$. The correlation structure of ϵ_{ij} was chosen as compound symmetric where $Cov(\epsilon_{ij}, \epsilon_{ik}) = \sigma^2[I\{j = k\} + \rho.I\{j \neq k\}]$.

3.3.1 Missing Completely at Random (MCAR)

For this section, we introduce missingness completely at random, in the datasets generated, via a missingness matrix D described before. The 0's of the matrix D are generated by Uniform random variables, and the parameters for the two models are taken as:

- Fixed effects model covariance terms: $\sigma^2 = 1$ and $\rho = 0.2$,
- Mixed effects model covariance terms: $\sigma^2 = 0.5$ and $\rho = 0.1$.

We record the final RMSE of each algorithm, after it converges, with 10% and 20% missingness in the data. Note that we used predictive mean matching in MICE and for each example, $n = 100,\ p = 5$ and $t = 4$ time points. Table 3.1 gives the RMSE of each algorithm for the fixed effects model and Table 3.2 gives the RMSE for the mixed effects model. We observe from the RMSE of each imputation method that the randomized versions of each random forest imputation algorithm performs nearly as good as the original method - while being computationally much faster. The On-the-fly imputation algorithm (RF_{otf}) generally performs the best in MCAR simulations and a close competitor is the unsupervised random forest imputation based on missForest (RF_{unsv}).

Table 3.1: RMSE Upon Convergence for MCAR Fixed Effects Model

Missing %	IKNN	MICE	RF_{prx}	$RF_{prx.R}$	RF_{otf}	$RF_{otf.R}$	RF_{unsv}	$RF_{unsv.R}$
10%	0.2201	0.1602	0.1421	0.1604	0.1032	0.1313	**0.1031**	0.1278
20%	0.3534	0.2837	0.1813	0.2058	**0.1501**	0.1599	0.1632	0.1734

Table 3.2: RMSE Upon Convergence for MCAR Mixed Effects Model

Missing %	IKNN	MICE	RF_{prx}	$RF_{prx.R}$	RF_{otf}	$RF_{otf.R}$	RF_{unsv}	$RF_{unsv.R}$
10%	0.2212	0.2408	0.1820	0.2203	**0.1691**	0.1902	0.1799	0.1812
20%	0.3009	0.3136	0.2341	0.2599	**0.1822**	0.2013	0.1832	0.1920

We can empirically plot the convergence of RMSE of the algorithm, as shown below in Figure 4.7. We can see that the On-the-fly random forest imputation performs the best, and is closely followed by the unsupervised random forest imputation, which uses missForest.

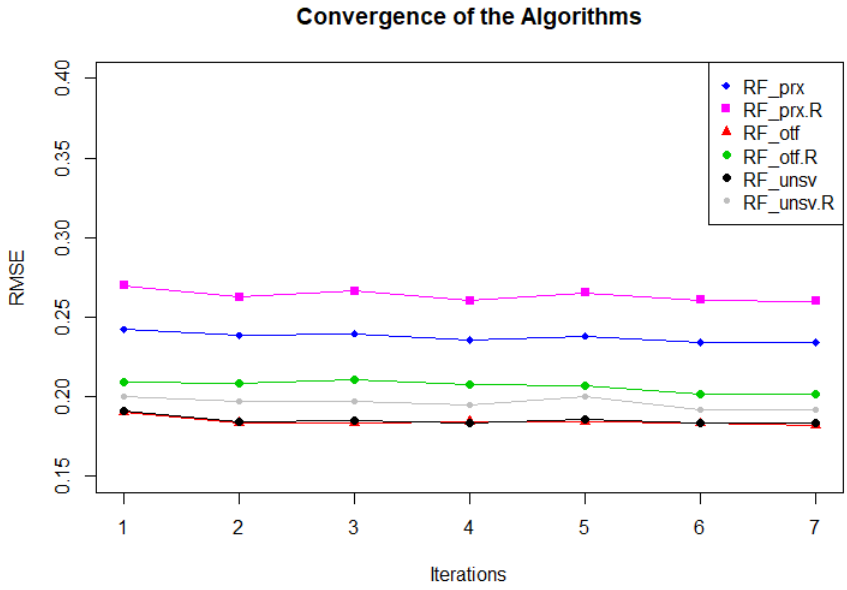

Figure 3.1: RMSE for all the RF Algorithms at 20% MCAR

3.3.2 Missing at Random (MAR)

For this section, we illustrate how our methods performs with respect to the other techniques. We simulate our longitudinal data from the multivariate normal distribution according to the same parameters taken at the MCAR examples, and then artificially introduce missingness in the data, at random (MAR), by letting the probability of missingness depend on the observed values. Also, the missingness is induced using a logistic model on the missingness matrix D. In real life, we often encounter time-independent covariates which are demographic in nature and often non-missing. For this example, we assume X_{ij1}, X_{ij2} and X_{ij3} to be non-missing and the missingness of X_{ij4} and X_{ij5} to be dependent on these demographic, non-missing variables, for each class k. Recall the $n \times t \times p$ missing matrix D, where we assume $D_{ij1}, D_{ij2}, D_{ij3}$ to be all 1 and

$$D_{ij4} \sim Ber(expit(p_{11} + p_{21} * X_{ij1} + p_{31} * X_{ij2} + p_{41} * X_{ij3})), \tag{3.5}$$

$$D_{ij5} \sim Ber(expit(p_{12} + p_{22} * X_{ij1} + p_{32} * X_{ji2} + p_{42} * X_{ij3})) \tag{3.6}$$

where $expit(x) = \frac{e^x}{1+e^x}$ and $p'_{ij}s$ are vectors of size $p = 5$ chosen by us.

For the MAR model, we take the fixed and mixed effects model described in the section before. We provide a detailed analysis of how all the 5 algorithms performed in this simulation setting with $n = 100,\ p = 5$ and $t = 4$ time points in Table 3.3 and 3,4. Note that we used predictive mean matching as the imputation model for MICE. We can see that in MAR model, the unsupervised random forest imputation technique based on missForest (RF_{unsv}) performs the best. Also, just like the MCAR model, the randomized versions of each imputation algorithm performs quite good with respect to the original random forest imputation technique.

Table 3.3: RMSE Upon Convergence for MAR Fixed Effects Model

Missing %	IKNN	MICE	RF_{prx}	$RF_{prx.R}$	RF_{otf}	$RF_{otf.R}$	RF_{unsv}	$RF_{unsv.R}$
10%	0.2992	0.2507	0.1385	0.1392	0.1108	0.1472	**0.1003**	0.1102
20%	0.3895	0.3499	0.1820	0.1956	0.1477	0.1607	**0.1225**	0.1304

Table 3.4: RMSE Upon Convergence for MAR Mixed Effects Model

Missing %	IKNN	MICE	RF_{prx}	$RF_{prx.R}$	RF_{otf}	$RF_{otf.R}$	RF_{unsv}	$RF_{unsv.R}$
10%	0.3021	0.3056	0.2092	0.2218	0.1747	0.2232	**0.1491**	0.1561
20%	0.4255	0.4012	0.2577	0.2619	0.1984	0.2107	**0.1723**	0.1815

We can empirically plot the convergence of RMSE of the algorithm, as shown below in Figure 3.2. We can see that the randomized unsupervised random forest imputation technique ($RF_{unsv.R}$) performs quite well with respect to the best technique, RF_{unsv}, and it is computationally much faster.

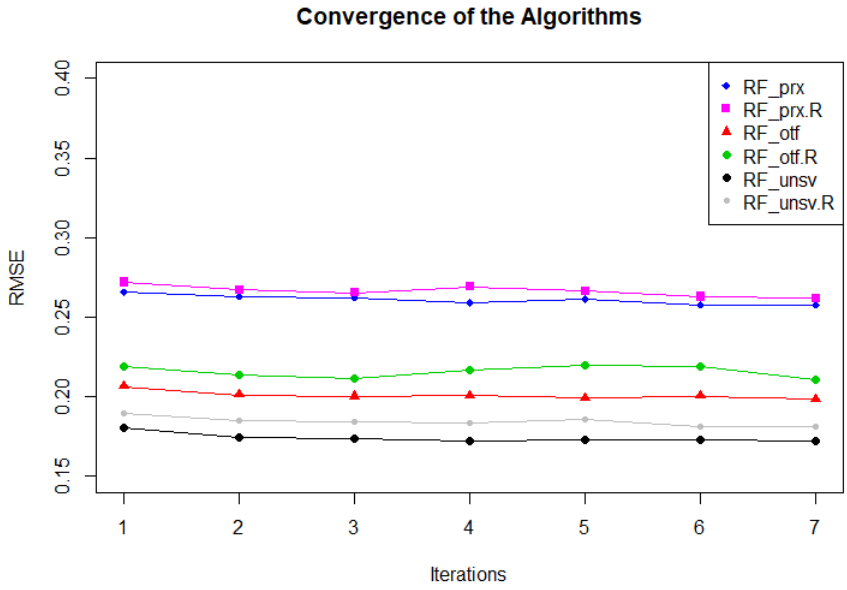

Figure 3.2: RMSE for the 5 Algorithms at 20% MAR

3.4 Discussion

We notice that for all the longitudinal models we considered in our simulation studies, the randomized versions perform well with respect to the non-randomized versions but take less time to compute. Hence, the fastest and reasonably accurate algorithm is the randomized unsupervised random forest imputation, $R_{unsv.R}$, for both MCAR and MAR cases. The most accurate imputation algorithm for the MCAR missingness case is the on-the-fly imputation method, R_{otf} and for the MAR missingness case is the missForest based imputation method, R_{unsv}. This method is suitable for imputation in categorical outcomes as well.

We plan to write an ARIC manuscript proposal to use the each visits data for diabetes indicators of each participants as well as some of the relevant covariates which affect diabetes, like diet and exercise factors which are also a part of the ARIC questionnaire, and collected at the visits. These factors would provide the covariate values needed for the longitudinal data model with diabetes as the outcome. For individuals who have dropped out of the study but have not died, we can take information about their covariates from the initial visits as well as the telephone calls, if answered. The participants who did not respond to telephone interview would have to be dropped from our study since it does not fit our imputation model. The imputation of covariates would help provide a better prediction model for diabetes. Our main task going ahead is to develop a random forest procedure to impute the outcome of interest, diabetes, for dropouts without any data from telephonic interviews (classic drop-out case). A good idea may be to use demographic, time-independent variables like sex, race, age, etc. taken at the beginning of the study - so that we could ascertain the diabetes status of a lot of non-compliant participants, and calculate an unbiased prevalence of diabetes in the ARIC study cohort.

CHAPTER 4: NATURAL LANGUAGE PROCESSING FOR CLINICAL DIAGNOSTIC CODES

4.1 Introduction

The final topic of this manuscript deals with converting the ICD codes into 5 word texts and then using text mining methods to cluster them. International Classification of Diseases (ICD) codes are known as International Statistical Classification of Diseases and related health problems and it is a group of codes developed mainly for clinical diagnostic purposes - used for statistical and epidemiological purposes afterwards. The ICD is maintained by the World Health Organization (WHO), which is the directing and coordinating authority for health diagnostic codes for classifying diseases - including complaints, social situations, surroundings and wide variety of other things like external causes of injury or disease. These ICD codes are revised after every few years and is currently in its 10th revision, called ICD-10 codes, since 1994 (ICD classifications, 2014), with the 11-th classification soon to arrive - but not widely used yet.

The ICD codes are a group of 5 alphanumeric symbols which provides a standardized method for the classification of patient's signs and symptoms at any medical encounter. They are used for both inpatient and outpatient settings. We decide to look at these codes as text from a book. Each patient hospitalization is treated as a document and each ICD code is treated as a term or a word and this is how we treat this as a bag-of-words model. This concept has not been explored too much in the past and very few attempts have been made to cluster these codes (Erraguntla et al. 2012, Pereira et al. 2013). We try to use text mining methods on

these ICD codes, to get meaningful cluster of symptoms related to a particular disease. We denote the document-term matrix to be X which is a $n \times p$ matrix and it may be very sparse given that there are less patients than the 5-digit ICD codes. Here the n represents all possible ICD-9 codes and p represents the number of hospitalizations of the patients, if the hospitalizations are looked at separately. So we must choose a suitable text mining method to cluster the ICD codes in that case. One possibility is to truncate the ICD codes to 3-digit codes which reveal the symptoms broadly, as in heart or lung or kidney disease and thus it can be used to make the matrix less sparse. This approach has been used before to rank hospitals and validate models (Cerrito 2008).

In our problem, we deal with a special longitudinal dataset where the data has been collected for more than 25 years across 4 states to find an exhaustive list of factors affecting heart diseases. The Atherosclerosis Risk in Communities (ARIC) study is a prospective cohort study designed to evaluate the causes of atherosclerosis and its clinical effects in a general population based sample of adults. Men and women, aged 45-64 years, were recruited and enrolled from four U.S. communities: Forsyth County, North Carolina; Jackson, Mississippi; suburbs of Minneapolis, Minnesota; and Washington County, Maryland. Follow-up examinations occurred in 1990 - 1992 (Visit 2 with n = 14,348, 93% of those still alive; when participants were 48-67 years of age), 1993-1995 (Visit 3), 1996-1998 (Visit 4) and 2011-2013 (Visit 5 with n = 6,538, 65% of those still alive, when participants were 65-90 years of age).

In the 5th visit of this study, participants had been diagnosed with dementia, mild cognitive disorder or no cognitive disorder. Our aim is to characterize the participants affected by dementia and identify the reasons for hospitalizations that may have preceded visit 5 of the ARIC study. In the 5th visit, the participants were classified into groups of ‘dementia’, ‘mild cognitive disorder’ and ‘no cognitive

disorder' based on their cognitive status. We group the 'mild cognitive disorder' and 'no cognitive disorder' participants into the 'no dementia' group. We basically look into the hospitalizations of the participants whose cognitive status ascertainment was done in Visit 5 and also enrolled in Medicare FFS insurance, so that a list of ICD-9 codes would be available for each of their hospitalizations prior to their Visit 5. We look into 5 years of hospitalization of each of these eligible candidates, prior to the 5th visit in the ARIC study. ICD-9 codes have been used to classify hospitalized patients into dementia category before (Pippenger et al. 2001), but no clustering effort has been made into a group of hospitalizations over many patients.

The rest of the chapter is organized as follows: section 2 contains Methodology summarizing the methods used to cluster the ICD-9 codes of the patients hospitalization, section 3 contains the results of the clustering of hospitalizations and section 4 contains a discussion of the results and further work which can be done for clustering these ICD-9 codes.

4.2 Methodology

4.2.1 Principal Components Analysis (PCA)

This is an **unsupervised** clustering method which is used to cluster numbers as well as text (Pearson 1901). We convert a set of correlated variables to linearly uncorrelated variables called principal components by orthogonalization. If there the feature matrix is $n \times p$ then the maximum number of principal components is $\min(n-1, p)$. The principal components of a data matrix X are obtained by maximizing the variance of the matrix, such that the first principal component is obtained by $w_{(1)} = \arg\max\{\frac{w^T X^T X w}{w^T w}\}$ and after $(k-1)$ principal components are obtained, the k-th component is subsequently obtained by $\widehat{X_k} = X - \sum_{s=1}^{k-1} X w_{(s)} w_{(s)}^T$.

Through this process we obtain an eigenvalue decomposition of the data matrix X. We can easily check that in this process, any combination of principal components is orthogonal to any combination of a different set of principal components.

4.2.2 Non-negative Matrix Factorization (NMF) with Gram Schmidt Orthogonalization

As a preliminary analysis, we have tried non-negative matrix factorization (NMF) coupled with Gram-Schmidt orthogonalization to make the components orthogonal. In general, non-negative matrix factorization (Lee and Seung 2001) is similar to PCA except that we use it to find the patterns with the same direction of correlation. The idea is to break up a data matrix into K factors such that

$$X_{n\times p} = W_{n\times K} H_{K\times p} = \sum_{k=1}^{K} W_{:,k} H_{k,:}$$

where $K << p$ and $X \geq 0$ is a non-negative data matrix. $W \geq 0$ is a set of non-negative observation factors and $W_{:,k}$ is a mixture of observations that comprise the k-th factor. $H_{kj} \geq 0$ is a set of non-negative feature factors, often sparse (mixture factors). $H_{k,:} \geq 0$ is a mixture of features that comprise the k-th factor.

Non-negative matrix factorization is often used in topic modelling and text mining. Let X be a matrix of news articles (rows) by words (columns) whose entries are word counts in the articles. Then $X_{n\times p} = W_{n\times K} H_{K\times p} = \sum_{k=1}^{K} W_{:,k} H_{k,:}$ is a sum of K topics. $X_{ij} = W_{i,:}^T H_{:,j}^T = \sum_{k=1}^{K} W_{ik} H_{kj}$.

Here the outer product of k-th column of W ($W_{:,k}$) and k-th row of H ($H_{k,:}$) is the topic k, like gay marriage. $H_{k,:}$ are the non-zero words contributing to topic k, like marriage, gay, equal, etc. $W_{:,k}$ are the non-zero news articles belonging to topic k, like “NC allows officials to refuse to perform gay marriages" (NY Times).

In our application, X is a matrix of people's hospitalizations (rows) by ICD-9 codes of the the corresponding hospitalization (columns) whose entries are word (frequency) counts. Then $X_{n\times p} = W_{n\times K}H_{K\times p} = \sum_{k=1}^{K} W_{:,k}H_{k,:}$ is a sum of K hospitalization clusters. Here the outer product of k-th column of W ($W_{:,k}$) and k-th row of H ($H_{k,:}$) is the hospitalization cluster k, like hospitalization for leg fracture. $W_{:,k}$ are the non-zero hospitalizations belonging to hospitalization cluster k, like serious hospitalizations. $H_{k,:}$ are the non-zero words (ICD-9 codes) contributing to hospitalization k, like bone fracture, bleeding, etc.

We basically calculate the NMF of the non-negative document-term matrix X by minimizing the Frobenius norm,

$$\min_{W,H} ||X - WH||_F \;\; s.t. \;\; W \geq 0, H \geq 0$$

where the Frobenius norm is basically defined for a $n \times p$ matrix X with entries X_{ij} as

$$||X||_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{p} |X_{ij}|^2}$$

After factorization of the matrix X we calculate the importance of the ICD-9 codes in each of the K factors by taking the rows of H, i,e, $H_{1,:}, H_{2,:}, ..., H_{K,:}$.

We follow up NMF with Gram-Schmidt orthogonalization (Cheney and Kincaid 2009) which makes the row vectors of H orthogonal to each other. This ensures that the ICD codes in the clusters are unique. This is equivalent to imposing a restriction $HH' = I$ while calculating the NMF of X.

The ideal number of clusters K (rank) is chosen by finding the point of inflection of the RSS plot against a range of clusters in the NMF. It can also be chosen where the dispersion measure of the NMF decreases sharply. The number of ICD-9 codes for

each of the clusters is also chosen by the same point of inflection rule of the importance measure of the factors, which is basically plotting the values of the rows of H in increasing order. Basically the point of inflection is where the first derivative is maximized and so it can be approximated by finding the point which has the maximum second order difference.

After finding all the important ICD-9 code in the K clusters, plot a dendrogram to show how each of the ICD-9 codes are clustered with each other. A dendrogram is a bottom up tree structure which measures the hierarchical relationship between objects based on their distance. We make a dendrogram based on the ICD-9 code membership in each of the clusters before Gram-Schmidt Orthogonalization.

We take each of the $\sum_{i=1}^{n} m_i$ hospitalizations of n patients and treat them as independent observations. Each hospitalization has a group of 3 digit ICD-9 codes accompanying it and we treat these as words of a paragraph (in a document) since each code represents the diagnosis of a symptom. We group the hospitalizations by the number of days the hospitalization discharge occurred prior to visit 5. The reason for grouping is that a single hospitalization would have only a handful of the 999 possible 3-digit ICD-9 codes and the resulting document-term matrix would be extremely sparse - like taking the term frequencies of a single paragraph in a document. So, we take a group of hospitalizations to form a document in the problem - much like a group of paragraphs make a document. For each 30 days of hospitalization, we treat all the hospitalizations as a single document and obtain the document-term matrix which is to be clustered.

4.3 Results

For our ARIC dataset looking into 5th visit participants with Medicare FFS Insurance, we have a total of 1951 hospitalizations in 5 years prior to diagnosis of dementia. Out of 1951, 1327 hospitalizations are for people classified as mild cognitive disorder or no cognitive disorder, which we mark as "non-dementia" category. The remaining 624 hospitalizations are of dementia patients. We also observe 197 different ICD-9 codes for non-dementia patients and 139 different codes for dementia patients. We group the 1327 and 624 hospitalizations based on the number of days of the discharge date falls before Visit 5. All hospitalizations in the 30 day periods before visit 5 is grouped into 1 document (recall the document-term matrix and over 5 years of hospitalization, 62 documents can be formed. So for dementia patients, the document-term matrix consists of 32 documents (n) and 624 terms or ICD-9 codes (p). For the non-dementia patients, the document-term matrix consists of $n = 32$ documents and $p = 1327$ terms or ICD-9 codes.

The diagnostics of the NMF for dementia patients are given in Figure 4.1 below. As we can see, the dispersion measure is minimum for $K = 4$ and we choose that as our number of clusters.

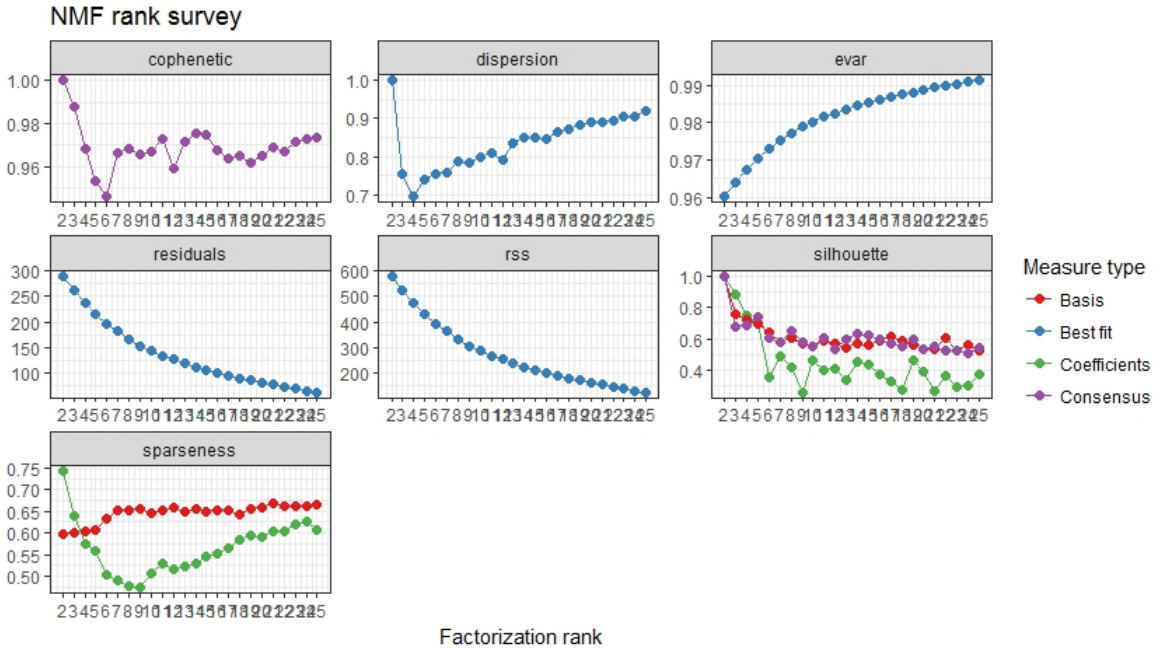

Figure 4.1: Dementia Patient Clustering Diagnostics

We analyze the dementia hospitalizations using NMF with Gram-Schmidt orthogonalization using $K = 4$. We select only those ICD-9 codes from each of the cluster whose importance lies above the aforementioned point of inflection of the curve. The importance plots of the ICD-9 codes for each of the 4 clusters are given below in figure 4.2.

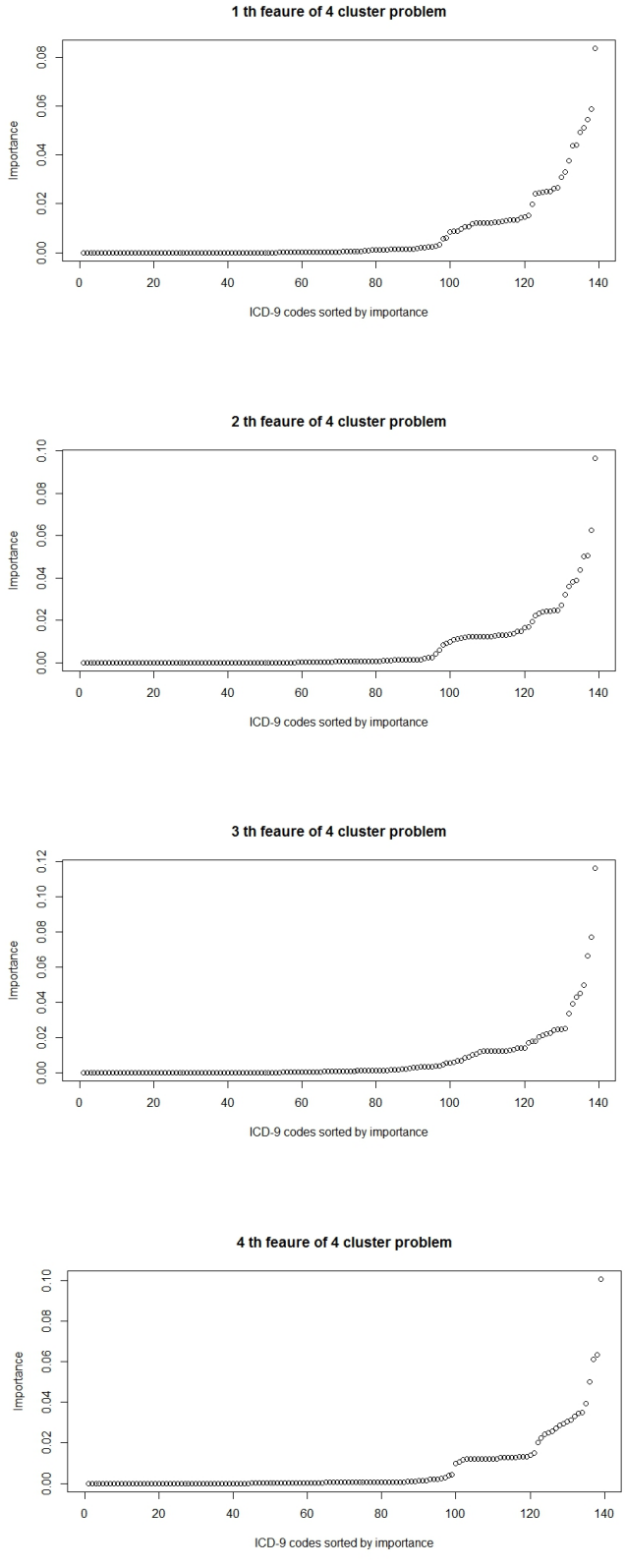

Figure 4.2: Importance Measures of the Clusters

Finally we make a dendrogram of the important ICD-9 codes for dementia as given in Figure 4.3 and it shows that the most prevalent symptoms relate to endocrinal, psychotic and genitourinary disorders.

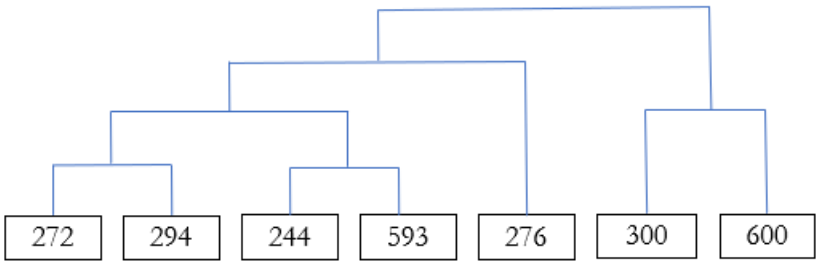

Figure 4.3: Dendrogram of Important ICD-9 Codes in Dementia Patients With Hospitalizations Binned in 30 Day Groupings

We can find the list of three-digit ICD-9 code summary below (Source: Wikipedia) in Figure 4.4.

- List of ICD-9 codes 001–139: infectious and parasitic diseases
- List of ICD-9 codes 140–239: neoplasms
- List of ICD-9 codes 240–279: endocrine, nutritional and metabolic diseases, and immunity disorders
- List of ICD-9 codes 280–289: diseases of the blood and blood-forming organs
- List of ICD-9 codes 290–319: mental disorders
- List of ICD-9 codes 320–389: diseases of the nervous system and sense organs
- List of ICD-9 codes 390–459: diseases of the circulatory system
- List of ICD-9 codes 460–519: diseases of the respiratory system
- List of ICD-9 codes 520–579: diseases of the digestive system
- List of ICD-9 codes 580–629: diseases of the genitourinary system
- List of ICD-9 codes 630–679: complications of pregnancy, childbirth, and the puerperium
- List of ICD-9 codes 680–709: diseases of the skin and subcutaneous tissue
- List of ICD-9 codes 710–739: diseases of the musculoskeletal system and connective tissue
- List of ICD-9 codes 740–759: congenital anomalies
- List of ICD-9 codes 760–779: certain conditions originating in the perinatal period
- List of ICD-9 codes 780–799: symptoms, signs, and ill-defined conditions
- List of ICD-9 codes 800–999: injury and poisoning

Figure 4.4: General Meaning of 3 Digit ICD-9 Codes

The diagnostics of the NMF for dementia patients are given in Figure 4.5 below. As we can see, the dispersion measure is minimum for $K = 4$ and we choose that as our number of clusters.

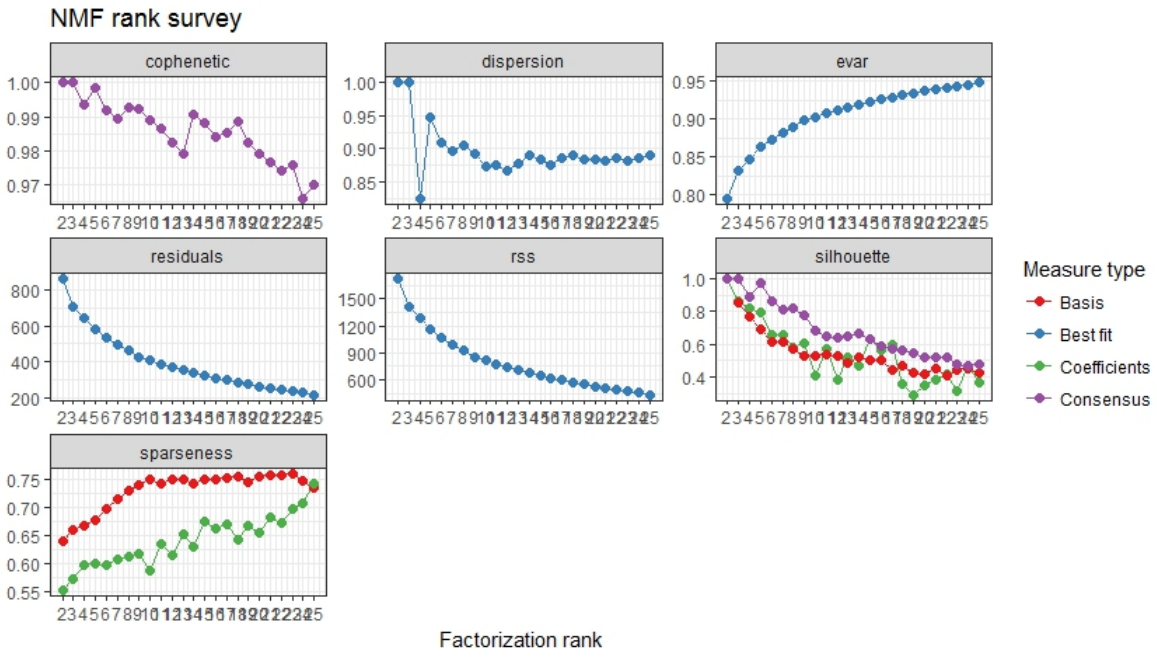

Figure 4.5: Non-dementia Patient Clustering Diagnostics

So, we analyze the non-dementia hospitalizations using NMF with Gram-Schmidt orthogonalization using $K = 4$. We select only those ICD-9 codes from each of the cluster whose importance lies above the aforementioned point of inflection of the curve. The importance plots of the ICD-9 codes for each of the 4 clusters are given below in figure 4.6.

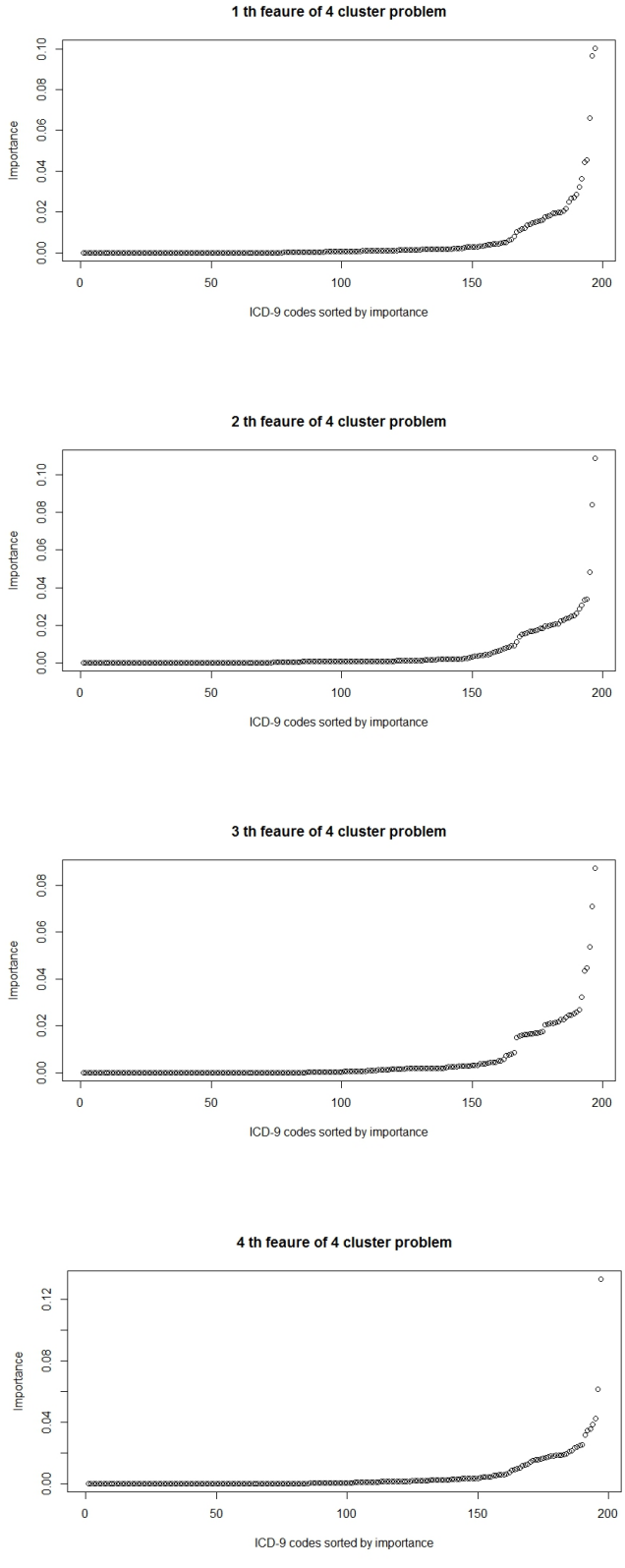

Figure 4.6: Importance Measures of the Clusters

Finally we make a dendrogram of the important ICD-9 codes for non-dementia patients are given in Figure 4.7 and we can see that those ICD-9 codes relate to esophageal, circulatory, skin and skeletal disorders. This is in contrast to the ICD-9 codes in the dementia patient clusters.

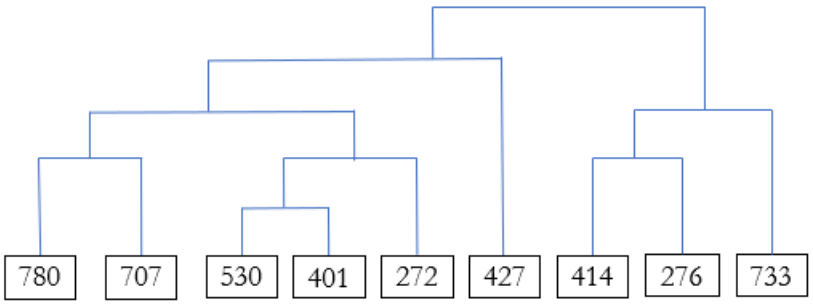

Figure 4.7: Dendrogram of Important ICD-9 Codes in Non-dementia Patients With Hospitalizations Binned in 30 Day Groupings

4.4 Discussions

We can find a greater frequency of codes for genitourinary and cerebral degeneration in participants with a positive dementia classification. Dementia affected participants had a greater frequency of ICD-9 codes for depression, heart failure, acute renal failure, and hormonal disorders, as compared to those with no cognitive impairment. In contrast, the ICD-9 codes for non-dementia patients were more general as it related to various other types of disorders throughout the body. Our goal, which was achieved, was to show ICD-9 code clusters for hospitalization of dementia patients differ from non-dementia patients.

The clustering methods used by us would be an improvement of the existing methods developed before in the University of Chicago which estimates Network Memberships by Principal Component Analysis (PCA) and Simplex Vertices Hunting, and also another topic estimation based on Singular Value Decomposition (SVD) (Jin

et al. 2017, Ke and Wang 2017). We hope to improve on the existing methods and develop predictive models for ARIC dataset hospitalizations, based on the ICD-9 codes.

REFERENCES

Aha, D. W., Kibler, D., and Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6(1):37–66.

Aittokallio, T. (2009). Dealing with missing values in large-scale studies: microarray data imputation and beyond. *Briefings in bioinformatics*, 11(2):253–264.

Alpaydin, E. (2009). *Introduction to Machine Learning*. MIT press.

ARIC investigators, T. (1989). The atherosclerosis risk in communities (aric) study, design and objectives. *American Journal of Epidemiology*, 129(4):687–702.

Bartlett, J. W., Seaman, S. R., White, I. R., Carpenter, J. R., and Initiative*, A. D. N. (2015). Multiple imputation of covariates by fully conditional specification: Accommodating the substantive model. *Statistical methods in medical research*, 24(4):462–487.

Batista, G. E. and Monard, M. C. (2002). A study of k-nearest neighbour as an imputation method. In *Second International Conference on Hybrid Intelligent Systems*, volume 87, pages 251–260.

Batista, G. E. and Monard, M. C. (2003). An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5-6):519–533.

Bertsimas, D., Pawlowski, C., and Zhuo, Y. D. (2017). From predictive methods to missing data imputation: An optimization approach. *Journal of Machine Learning Research*, 18:196–1.

Brás, L. P. and Menezes, J. C. (2007). Improving cluster-based missing value estimation of dna microarray data. *Biomolecular engineering*, 24(2).

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees*. Wadsworth, Belmont, California.

Buuren, S. v. (2007). Multiple imputation of discrete and continuous data by fully conditional specification. *Statistical Methods in Medical Research*, 16(3):219–242.

Buuren, S. v., Brand, J. P., Groothuis-Oudshoorn, C. G., and Rubin, D. B. (2006). Fully conditional specification in multivariate imputation. *Journal of Statistical Computation and Simulation*, 76(12):1049–1064.

Buuren, S. v. and Oudshoorn, K. (1999). *Flexible mutlivariate imputation by MICE*. TNO Prevention and Health.

Cerrito, P. B. (2008). Text mining to define a validated model of hospital rankings. In *Emerging Technologies of Text Mining: Techniques and Applications*, pages 268–296. IGI Global.

Cheney, W. and Kincaid, D. (2009). Linear algebra: Theory and applications. *The Australian Mathematical Society*, 110.

Choudhury, A. and Kosorok, M. R. (2020). Missing data imputation for classification problems. *arXiv preprint arXiv:2002.10709*.

Cios, K. J. and Moore, G. W. (2002). Uniqueness of medical data mining. *Artificial Intelligence in Medicine*, 26(1-2):1–24.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39:1–38.

Deng, J.-L. (1982). Control problems of grey systems. *Systems & Control Letters*, 1(5):288–294.

Duda, R. O., Hart, P. E., and Stork, D. G. (2012). *Pattern Classification*. John Wiley & Sons.

Dudani, S. A. (1976). The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, (4):325–327.

Enders, C. K. (2010). *Applied missing data analysis*. Guilford press.

Erraguntla, M., Gopal, B., Ramachandran, S., and Mayer, R. (2012). Inference of missing icd 9 codes using text mining and nearest neighbor techniques. In *2012 45th Hawaii International Conference on System Sciences*, pages 1060–1069. IEEE.

Farhangfar, A., Kurgan, L., and Dy, J. (2008). Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition*, 41(12):3692–3705.

Farhangfar, A., Kurgan, L. A., and Pedrycz, W. (2007). A novel framework for imputation of missing values in databases. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(5):692–709.

Fernandes, A. S., Jarman, I. H., Etchells, T. A., Fonseca, J. M., Biganzoli, E., Bajdik, C., and Lisboa, P. J. (2008). Missing data imputation in longitudinal cohort studies: Application of plann-ard in breast cancer survival. In *2008 Seventh International Conference on Machine Learning and Applications*, pages 644–649. IEEE.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

García-Laencina, P. J., Sancho-Gómez, J.-L., and Figueiras-Vidal, A. R. (2013). Classifying patterns with missing values using multi-task learning perceptrons. *Expert Systems with Applications*, 40(4):1333–1341.

García-Laencina, P. J., Sancho-Gómez, J.-L., Figueiras-Vidal, A. R., and Verleysen, M. (2009). K nearest neighbours with mutual information for simultaneous classification and missing data imputation. *Neurocomputing*, 72(7-9):1483 – 1493.

Gelman, A. and Hill, J. (2006). *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge university press.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning*. Springer Series in Statistics. Springer, New York, second edition. Data mining, inference, and prediction.

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.

Honaker, J., King, G., and Blackwell, M. (2011). Amelia II: A program for missing data. *Journal of Statistical Software*, 45(7):1–47.

Huang, C.-C. and Lee, H.-M. (2004). A grey-based nearest neighbor approach for missing attribute value prediction. *Applied Intelligence*, 20(3):239–252.

Huang, C.-C. and Lee, H.-M. (2006). An instance-based learning approach based on grey relational structure. *Applied Intelligence*, 25(3):243–251.

Ibrahim, J. G. and Molenberghs, G. (2009). Missing data methods in longitudinal studies: a review. *Test*, 18(1):1–43.

Ishwaran, H. and Lu, M. (2008). Random survival forests. *Wiley StatsRef: Statistics Reference Online*, pages 1–13.

Jin, J., Ke, Z. T., and Luo, S. (2017). Estimating network memberships by simplex vertex hunting. *arXiv preprint arXiv:1708.07852*.

Joe, H. (2006). Generating random correlation matrices based on partial correlations. *Journal of Multivariate Analysis*, 97(10):2177–2189.

Ke, Z. T. and Wang, M. (2017). A new svd approach to optimal topic estimation. *arXiv preprint arXiv:1704.07016*.

Kim, H., Golub, G. H., and Park, H. (2004). Missing value estimation for dna microarray gene expression data: local least squares imputation. *Bioinformatics*, 21(2):187–198.

Kullback, S. (1997). *Information theory and statistics*. Courier Corporation.

Kwak, N. and Choi, C.-H. (2002). Input feature selection by mutual information based on parzen window. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (12):1667–1671.

Lakshminarayan, K., Harp, S. A., and Sammad, T. (1999). Imputation of missing data in industrial databases. *Applied Intelligence*, 11(3):259–275.

Le Gruenwald, M. H. (2005). Estimating missing values in related sensor data streams. In *COMAD*.

Lee, D. D. and Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562.

Li, D., Deogun, J., Spaulding, W., and Shuart, B. (2004). Towards missing data imputation: a study of fuzzy k-means clustering method. In *International Conference on Rough Sets and Current Trends in Computing*, pages 573–579. Springer.

Li, K.-H. (1988). Imputation using markov chains. *Journal of Statistical Computation and Simulation*, 30(1):57–79.

Liao, S. G., Lin, Y., Kang, D. D., Chandra, D., Bon, J., Kaminski, N., Sciurba, F. C., and Tseng, G. C. (2014). Missing value imputation in high-dimensional phenomic data: imputable or not, and how? *BMC bioinformatics*, 15(1):346.

Liaw, A., Wiener, M., et al. (2002). Classification and regression by randomforest. *R news*, 2(3):18–22.

Little, R. J. and Rubin, D. B. (1987). *Statistical Analysis with Missing Data*. Wiley, New York.

Little, R. J. and Rubin, D. B. (2002). *Statistical Analysis with Missing Data*, volume 2. John Wiley & Sons.

Luengo, J., García, S., and Herrera, F. (2012). On the choice of the best imputation methods for missing values considering three groups of classification methods. *Knowledge and Information Systems*, 32(1):77–108.

Mirabelli, M. C., Preisser, J. S., Loehr, L. R., Agarwal, S. K., Barr, R. G., Couper, D. J., Hankinson, J. L., Hyun, N., Folsom, A. R., and London, S. J. (2016). Lung function decline over 25 years of follow-up among black and white adults in the aric study cohort. *Respiratory medicine*, 113:57–64.

Newman, D. J., Hettich, S. C., Blake, C. L., and Merz, C. J. (2008). UCI repository of machine learning databases.

Nooraee, N., Molenberghs, G., Ormel, J., and Van den Heuvel, E. R. (2018). Strategies for handling missing data in longitudinal studies with questionnaires. *Journal of Statistical Computation and Simulation*, 88(17):3415–3436.

Pan, R., Yang, T., Cao, J., Lu, K., and Zhang, Z. (2015). Missing data imputation by k nearest neighbours based on grey relational structure and mutual information. *Applied Intelligence*, 43(3):614–632.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pereira, L., Rijo, R., Silva, C., and Agostinho, M. (2013). Icd9-based text mining approach to children epilepsy classification. *Procedia Technology*, 9:1351–1360.

Pippenger, M., Holloway, R. G., and Vickrey, B. G. (2001). Neurologists' use of icd-9cm codes for dementia. *Neurology*, 56(9):1206–1209.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc.

Raghunathan, T. E., Lepkowski, J. M., Van Hoewyk, J., and Solenberger, P. (2001). A multivariate technique for multiply imputing missing values using a sequence of regression models. *Survey Methodology*, 27(1):85–96.

Rubin, D. B. (1977). Formalizing subjective notions about the effect of nonrespondents in sample surveys. *Journal of the American Statistical Association*, 72(359):538–543.

Rubin, D. B. and Schafer, J. L. (1990). Efficiently creating multiple imputations for incomplete multivariate normal data. In *Proceedings of the Statistical Computing Section of the American Statistical Association*, volume 83, page 88. American Statistical Association.

Schafer, J. L. (1997). *Analysis of incomplete multivariate data*. Chapman and Hall.

Sefidian, A. M. and Daneshpour, N. (2019). Missing value imputation using a novel grey based fuzzy c-means, mutual information based feature selection, and regression model. *Expert Systems with Applications*, 115:68–94.

Spratt, M., Carpenter, J., Sterne, J. A., Carlin, J. B., Heron, J., Henderson, J., and Tilling, K. (2010). Strategies for multiple imputation in longitudinal studies. *American journal of epidemiology*, 172(4):478–487.

Städler, N., Stekhoven, D. J., and Bühlmann, P. (2014). Pattern alternating maximization algorithm for missing data in high-dimensional problems. *The Journal of Machine Learning Research*, 15(1):1903–1928.

Stein, B. V. and Kowalczyk, W. (2016). An incremental algorithm for repairing training sets with missing values. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 175–186. Springer.

Stekhoven, D. J. and Bühlmann, P. (2011). Missforest - non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118.

Stone, M. (1974). Cross-validation and multinomial prediction. *Biometrika*, 61(3):509–515.

Tang, F. and Ishwaran, H. (2017). Random forest missing data algorithms. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 10(6):363–377.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. (2001). Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525.

Tsai, C.-J., Lee, C.-I., and Yang, W.-P. (2008). A discretization algorithm based on class-attribute contingency coefficient. *Information Sciences*, 178(3):714–731.

Villmann, T., Schleif, F.-M., and Hammer, B. (2006). Comparison of relevance learning vector quantization with other metric adaptive classification methods. *Neural Networks*, 19(5):610–622.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y., et al. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37.

Zhang, S. (2011). Shell-neighbor method and its application in missing data imputation. *Applied Intelligence*, 35(1):123–133.

Zhang, S. (2012). Nearest neighbor selection for iteratively knn imputation. *Journal of Systems and Software*, 85(11):2541–2552.