

MAJOR PROJECT REPORT ON
Missing Value Imputation For Multivariate Data Using Data Depth

*Submitted in partial fulfilment of the requirements
for the award of the degree of*

M.Sc. Statistics

Submitted By

**Aman Bashir Sheikh
(2022MSST003)**

Under the guidance of
Dr. Mahesh Barale



Department Of Statistics
School of Mathematics, Statistics
and Computational Sciences
Central University of Rajasthan, India

May 2024

Dr. Mahesh Barale
Assistant Professor
Department of Statistics

Email : mahesh.barale@curaj.ac.in
Website: <http://www.curaj.ac.in>



राजस्थान केंद्रीय विश्वविद्यालय
Central University of Rajasthan
(संसद के अधिनियम के तहत स्थापित केंद्रीय विश्वविद्यालय)
(A Central University by an Act of Parliament)
NH-8, Bandarsindri, 305801
Kishangarh (Ajmer), Rajasthan, INDIA
Phone (Office): +91-1463-238755/260200
Telefax: +91-1463-238722

Certificate

This is to certify that the work embodied in the accompanying project report entitled **Missing Value Imputation For Multivariate Data Using Data Depth** has been successfully carried by **Mr. Aman Bashir Sheikh**, a *IVth* Semester student of M.Sc. Statistics, of the Department of Statistics, Central University of Rajasthan, under my guidance and supervision. He worked for about ... weeks and her work carried out is satisfactory.

Place: Central University of Rajasthan
District-Ajmer, Rajasthan

Date:

Signature
Dr. Mahesh Barale
Assistant Professor
Department of Statistics



TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Mr. Aman Bashir Sheikh, Enroll. No 2022MSST003**, M.Sc. Statistics student of Department of Statistics has done project work **Missing Value Imputation For Multivariate Data Using Data Depth** under the guidance of Dr. Mahesh Barale, Assistant Professor at Department of Statistics, Central University of Rajasthan towards the partial fulfillment of the award of "Master of Science in Statistics" during the period Feb 2024 to May 2024.

Place: Central University of Rajasthan

District- Ajmer , Rajasthan

Date:

Signature
Dr. Deepesh Bhati
Head
Department of Statistics
central University of Rajasthan



Declaration

I, **Mr.Aman Bashir Sheikh**, hereby declare that the project work entitled **Missing Value Imputation For Multivariate Data Using Data Depth** submitted to Department of Statistics, Central University of Rajasthan as a partial fulfilment of requirements of IV-Semester examination, is a bonafide record of work under taken by me, under the supervision of Dr. Mahesh Barale, Assistant Professor at Department of Statistics, Central University of Rajasthan and it is not formed the basis for the award of any other Degree/Associateship/Fellowship by any University.

Place: Central University of Rajasthan

District- Ajmer , Rajasthan

Date:

Signature of Candidate
Mr.Aman Bashir Sheikh
Enrolment no. 2022MSST003



Acknowledgements

I would like to express my sincere gratitude to all those who have supported and guided me throughout the journey of completing this thesis. First and foremost, I am deeply thankful to my supervisor, **Dr. Mahesh Barale**, for their invaluable guidance, unwavering support, and insightful feedback. Their expertise and dedication played a pivotal role in shaping the direction of my research and enhancing the quality of this work. My heartfelt thanks go to my family and friends for their constant encouragement and understanding during this challenging phase of my academic journey. Their belief in me kept me motivated to overcome obstacles and strive for excellence. Lastly, I am indebted to the various resources, libraries, and research materials that have contributed to the depth and authenticity of this thesis.

Thank you all for being a part of this fulfilling endeavor.

Mr.Aman Bashir Sheikh
Enrollment No-2022MSST003

Contents

Contents	ii
1 Introduction	2
2 Data Depth	4
2.1 Data Depth	4
2.2 Depth In One Dimension	5
2.3 Depth In Two or more Dimensions	6
2.4 Depth Function	7
2.4.1 Mahalanobis Depth	8
2.4.2 Simplicial Depth	8
3 Imputation by depth Maximization	9
3.1 K-max method	9
3.1.1 Algorithm 1	10
3.2 Iterative Imputation	11
3.2.1 Algorithm 2	12
3.3 Class Based Imputation	13
3.3.1 Algorithm 3	13
4 Simulation Study	14
4.1 Data Description	14
4.2 Normalized Root Mean Squared Error	14
4.3 Performance study bupa dataset	16
4.3.1 bupa dataset at 5% missing value	16
4.3.2 Average NRMSE for Bupa data with 10% missing values	17
4.3.3 bupa dataset at 15% missing value	18
4.3.4 banknoten with 5% missing value	19
4.3.5 banknoten with 10% missing value	20
4.3.6 banknoten with 15% missing value	21
4.3.7 Normal distribution with 10% missing value	22
4.3.8 Time Comparision	23

5 Conclusion	24
Bibliography	26
5.1 Appendices	28
5.1.1 Used R-Packages	28
5.1.2 R-code	28
% footer content centered	

Chapter 1

Introduction

When data for one or more variables in a dataset are missing, it's referred to as missing data. A number of factors, including incorrect data entry, broken equipment, and survey non-response, can result in these missing values. A respondent may decline to comment on a question on the grounds of privacy concerns. Alternatively, the survey respondent is misinterpreting the question. The respondent might have responded, but that response would not have fit among the options that were given. In statistical analysis, missing values can present challenges as they may affect the accuracy and reliability of results. Missing data are questions without answers or variables without observation.

Multivariate data consist of measurements or observations on more than one characteristics or variables for each unit or individuals in the dataset. Missing values in datasets significantly impact analysis accuracy and interpretability. Consider already available datasets in r, viz., banknoten and liver disorder data (BUPA). suppose there is some level of missingness in banknote dimension data, its presence distorts classification accuracy and hinders feature importance assessment, affecting algorithm performance. Similarly, in liver disorder diagnosis data, missing values challenge diagnostic accuracy, feature importance identification, patient risk stratification, and treatment planning. Absence of critical variables compromises diagnostic precision and predictor identification, hindering risk stratification and treatment planning. Prudent handling of missing values, through imputation or model adjustments, is essential for reliable insights and informed decisions in both classification tasks and clinical practice. Effective addressing of missing values enhances analysis reliability and utility, ultimately benefiting patient care and outcomes.

In the literature, various methods for missing value imputation exist, including mean imputation ([Little and Rubin \(2019\)](#)) and other non parametric methods such as KNN imputation ([Troyanskaya et al. \(2001\)](#)) and Random Forest imputation ([Stekhoven and Bühlmann \(2012\)](#), [Pujiyanto et al. \(2019\)](#)). KNN imputation is a valuable technique for handling missing data estimating values based on the characteristics of neighboring data points ([Beretta and Santaniello \(2016\)](#)). In *Mean Imputation* the mean of the observed value for each variable is computed and the missing value for that variable are replaced by this mean ([Donders et al. \(2006\)](#)).

Nonparametric imputation by data depth: In this paper missing values are imputed in three simplest steps - 1) Use unconditional mean imputation to initialise missing values arbitrarily; 2) Impute missing values in one variable by the values predicted by the variable's regression model, using the

remaining variables as explanatory variables. 3) until convergence, iterate through the variables that have missing values ([Mozharovskyi et al. \(2019\)](#)).

class based missing value Imputation It has two underlying modes. While the second one uses the identified threshold for missing value imputation, the first one focuses on identifying the imputation threshold based on the distances between the class centre and their corresponding data samples ([Tsai et al. \(2018\)](#)).

In our proposed method, we address the issue of missing value imputation by using a depth-maximizing approach. We utilize conditional mean imputation to initialize the missing data. We will later use several imputation techniques to improve this initialization. Here, we iteratively replace the greatest depth corresponding to the mean-imputed matrix until the difference between the mean-imputed matrix and the imputed matrix is less than a predefined error threshold. We also present a novel method specifically tailored for categorical columns. This method iteratively imputes missing values while identifying categorical variables, with a focus on preserving category-wise integrity. By combining these methods, our methodology offers a comprehensive framework for accurate and dependable missing value imputation, which may increase the efficacy of data.

Chapter 2

Data Depth

2.1 Data Depth

If $D(\cdot/x)$ is bounded and non-negative, then it is a statistical depth function that maps \mathbb{R}^d to R. Following are the important properties of the depth function

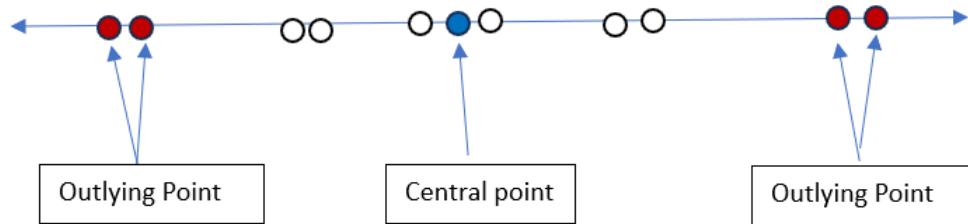
- **Affine invariant :** $D(y/Y) = D(Cy + d/CY + d)$ for any invertible $d \times d$ matrix C and any d belong to \mathbb{R}^d .
- **Maximal at the symmetry centre:**
 $D(c/x) = \sup_{x \in \mathbb{R}^d} D(x/X)$ for any X half-space symmetry around c.
- **Monotone With respect to the deepest point**
For any Y having c as a deepest point $D(y/Y) \geq D(\alpha c + (1 - \alpha)y/Y)$ for any $x \in \mathbb{R}^d$ [0,1].
- **Vanishing at infinity:**
 $\lim_{||x|| \rightarrow 0} D(y/Y) = 0$
- **Quasi concavity :**
Quasi concavity of $D(\cdot/Y)$, upper level sets (or depth trimmed regions) $D_a(y) = y \in \mathbb{R}^d : D(y/Y) \geq a$ to be convex.

Above are the important properties of the depth function

2.2 Depth In One Dimension

The measurement of a point's relative position within a given data cloud is called its data depth.

Figure 2.1: Depth in One Dimension



As a result, the points are arranged in relation to the data cloud from the centre outward.

The blue point (central point) has almost equal number of points on its left and right sides ,while the four red points (outlying point) have almost all the observation only on one side.

- Depth of the point

$$x = \min\{F_n(x), 1 - F_n(x)\} \quad (1)$$

Where F_n is the proportion of data points that are on the left of x (the empirical distribution function)
Note that the maximum possible value of the depth function is 1/2
where,

- F_n gives the point lies on the left of x
- $1 - F_n(x)$ gives the point lies on the right of x

For symmetric distribution the maximum depth is 1/2 .

Median be the point of maximum depth

Depth of point $x = \min\{F_n(x), 1 - F_n(x)\}$

Consider the distribution is symmetric around 0, then

$$D_p(0) = \min\{F_n(0), 1 - F_n(0)\}$$

$$D_p(0) = \frac{1}{2}$$

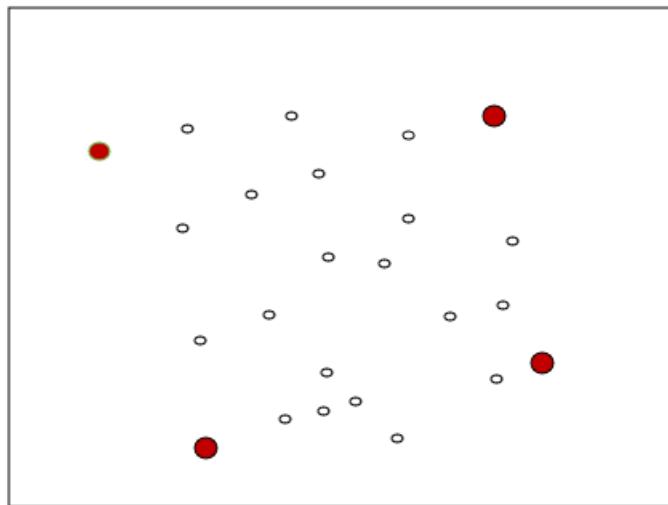
which is the maximum possible value of the depth function

If we take an example of Normal distribution with $\mu = 0 \& \sigma = 1$

then point Zero have the maximum value of depth

2.3 Depth In Two or more Dimensions

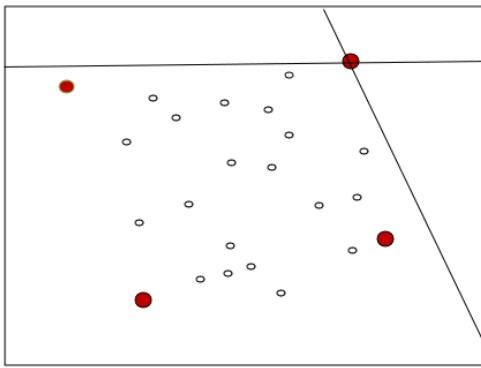
How do we extend these ideas in R^d for $d \geq 2$?.



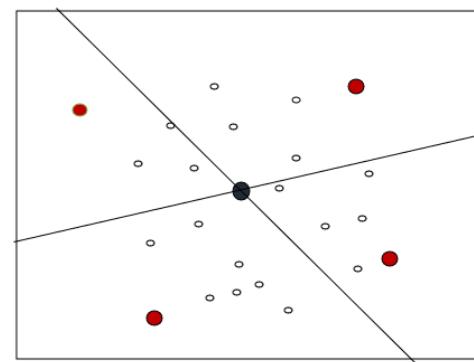
One method is to visualise a line through x (hyperplanes for $d \geq 3$) and measure the percentage of data points that are located on the line's two sides, or half spaces.

After taking into account all possible lines through x , we can determine the minimum of the proportion of data points in any half space of the line through x . This is known as the *Tukey Depth* ([Tukey \(1975\)](#)), or half-space depth. Any line passing through the blue point has nearly equal numbers of data points in both half spaces, whereas all of the data points are located on the half space of a single line passing through the red point.

Note: It depends on the line's orientation and rotation.



(a) fig:1



(b) fig:2

Figure 2.2: Data Depth in 2-Dimension

In relation to a given data cloud, the half space depth leads to the points in R^d to be arranged centre outward. The smallest number of data points lying in one of the sides of a line passing through x defines the location depth of a point in the bivariate case with respect to the two-dimensional data set S_n .

2.4 Depth Function

To expand the univariate notations of median, quantiles, ranks, signs, and order statistics to the context of multivariate data, depth functions were developed.

A depth function measures centrality, whereas a probability density function measures local probability weight. Multivariate outlyingness, quantile, sign, and rank functions are all strongly correlated with the contour of a multivariate depth function. When combined, these functions provide a potent methodology for non-parametric multivariate data description, outlier detection, data analysis, and inference. Examples of such applications include multivariate boxplots, location and scatter estimation, and symmetry testing.

There are several Depth function in literature such as

- Mahalanobis Depth ([Mahalanobis \(2018\)](#))
- Tukeys half-space depth ([Tukey \(1975\)](#))
- Simplicial Depth ([Liu \(1990\)](#))
- Spatial depth ([Vardi and Zhang \(2000\)](#))
- Projection Depth ([Zuo and Serfling \(2000\)](#)), etc.

2.4.1 Mahalanobis Depth

In relation to a p-dimensional data set S_n , the *Mahalanobis Depth* of a point $x \in S_n \subset R^p$ is defined as :

$$M_hD(x; s_n) = \frac{1}{[1 + (x - \bar{x})^T \Sigma^{-1} (x - \bar{x})]} \quad (2)$$

where the variance-covariance matrix of S_n and the mean vector are represented by *Barx* and *Sigma* ([Neto \(2008\)](#)).

2.4.2 Simplicial Depth

The number of closed simplex in S_n that contain x and have $p+1$ vertices is known as the simplicial depth of a point $x \in S_n \subset R^p$ with respect to the p -dimensional data set S_n .

The number of triangles with vertices in S_n and containing x is the simplicial depth of a point x in the bivariate case([Neto \(2008\)](#)).

Chapter 3

Imputation by depth Maximization

3.1 K-max method

Let X be a random vector in \mathbb{R}^d , represented as $X = (x_1, x_2, \dots, x_n)^T$, where each x_i is a sample point in \mathbb{X} . Consider X as a data matrix containing missing values. We distinguish between "incomplete rows" (incomplete-row (i)), which have missing entries, and "complete rows" (complete-row (i)), which contain all values.

The simplest method for imputing missing values involves:

- Initially, we classify the data into complete and incomplete rows based on the presence of missing entries in incomplete-row (i) .
- Next, we employ Mean Imputation to address missing values in the dataset X . This involves replacing missing entries with the column mean, creating a modified dataset denoted as X_{Mid} .
- We then construct a dummy matrix, X_{Dummy} , of the same dimension as X_{Mid} , where incomplete rows are replicated to maintain consistency.
- By assessing the depth of both X_{Dummy} and X_{Mid} , we identify the k points with the maximum depth.
- Finally, we replace missing values in the original dataset with corresponding values from X_{Mid} based on the maximum depth, effectively completing the imputation process.

3.1.1 Algorithm 1

Algorithm 1 missing_info_kmax

```

1:  $x \leftarrow$  Data with Missing value
2:  $mid \leftarrow$  Mean Imputed Matrix
3:  $depth\_fun \leftarrow$  Depth Function
4:  $k \leftarrow$  k point which having maximum depth
5: function MISSING_INFO_KMAX( $x, mid, depth\_fun, k$ )
6:    $x \leftarrow$  MATRIX( $x$ )
7:    $mid \leftarrow$  MATRIX( $mid$ )
8:    $row\_miss \leftarrow$  ROWS_WITH_MISSING_VALUES( $x$ )
9:    $location\_miss \leftarrow$  LOCATION_OF_MISSING_VALUES( $x$ )
10:   $location\_non\_miss \leftarrow$  LOCATION_OF_NON_MISSING_VALUES( $x$ )
11:   $incomplete\_rows \leftarrow$  ROWS_WITH_ANY_MISSING_VALUES( $x$ )
12:   $complete\_data \leftarrow$  COMPLETE_ROWS( $x$ )
13:   $incomplete\_data \leftarrow$  INCOMPLETE_ROWS( $x$ )
14:   $miss\_ind \leftarrow$  LOCATION_OF_MISSING_VALUES( $incomplete\_data$ )
15:   $imp\_data \leftarrow$  CREATE_EMPTY_MATRIX_WITH_SAME_DIMENSIONS_AS( $mid$ )
16:  for  $i \leftarrow 1$  to NUMBER_OF_ROWS( $incomplete\_data$ ) do
17:     $loc\_miss\_inc \leftarrow$  LOCATION_OF_MISSING_VALUES_IN_ROW( $incomplete\_data[i, ]$ )
18:     $dummy\_matrix \leftarrow$  CREATE_MATRIX_WITH_INCOMPLETE_ROW_REPLACED_BY_MID( $incomplete\_data[i, ], mid$ )
19:     $Est\_Meth \leftarrow$  "moment"
20:     $depth\_function \leftarrow$  CALCULATE_DEPTH_FUNCTION( $depth\_fun, dummy\_matrix, mid, Est\_Meth$ )
21:     $max\_depth\_indices \leftarrow$  TOP_K_INDICES( $depth\_function, k$ )
22:     $max\_depth\_points \leftarrow$  SELECT_POINTS_BY_INDICES( $mid, max\_depth\_indices$ )
23:    if  $k == 1$  then
24:       $imputed\_value \leftarrow$  VALUE_OF_MAX_DEPTH_POINT_FOR_MISSING_LOCATIONS( $loc\_miss\_inc, max\_depth\_points$ )
25:    else
26:       $value \leftarrow$  CALCULATE_COLUMN_MEANS( $max\_depth\_points$ )
27:       $imputed\_value \leftarrow$  VALUES_FOR_MISSING_LOCATIONS_IN_COLUMN_MEANS( $loc\_miss\_inc, value$ )
28:    end if
29:     $row\_miss\_i \leftarrow$  ROW_MISS_INDEX_FOR_ROW( $i$ )
30:    UPDATE_IMPUTED_DATA_AND_MID( $row\_miss\_i, loc\_miss\_inc, imputed\_value, imp\_data, mid$ )
31:  end for
32:   $imputed\_data \leftarrow imp\_data$ 
33:  return ( $imputed\_data, err$ )
34: end function

```

3.2 Iterative Imputation

The iterative imputation method is crafted to adeptly manage missing values within a dataset, denoted as matrix X . Through a series of iterations, it endeavors to steadily approximate missing entries until either reaching *convergence* or exhausting a predetermined *iteration limit*. Here's a concise breakdown of its core procedures

The method primarily comprises two steps: Data Pre-processing & Imputation, followed by Checking Convergence Criteria.

- **Data Pre-processing and Imputation:**

1. Identify incomplete rows containing missing values and classify the data as complete and incomplete.
2. Impute the mean at the location of missing values in the provided dataset and designate it as X_{Mid} .
3. For each incomplete row, construct the dummy matrix (X_{Dummy}) and compute the depth $D(X_{Dummy}, X_{Mid})$.
4. Identify k points with the highest depth corresponding to X_{Mid} and replace the missing value by the mean of those k values.
5. Update the imputed dataset with the newly estimated values (new imputed data).

- **Convergence Criteria:**

1. Monitor the maximum absolute difference between consecutive imputed datasets.
2. Terminate the iteration if the maximum difference falls below a predetermined threshold or if the maximum number of iterations is reached.

3.2.1 Algorithm 2

Algorithm 2 iterative_imputation

```

x ← Data with Missing value
2: mid ← Mean Imputed Matrix
   depth_fun ← Depth Function
4: k ← k point which having maximum depth
   num_iter ← Number of iteration
6: req_err ← Specified Errors
function ITERATIVE_IMPUTATION(x,depth_fun,k,num_iter,req_err)
8:   x ← MATRIX(x)
   location_miss ← LOCATION_OF_MISSING_VALUES(x)
10:  incomplete_row ← ROWS_WITH_ANY_MISSING_VALUES(x)
   incomplete_col ← COLUMNS_WITH_ANY_MISSING_VALUES(x)
12:  complete_data ← COMPLETE_ROWS(x)
   incomplete_data ← INCOMPLETE_ROWS(x)
14:  miss_ind ← LOCATION_OF_MISSING_VALUES(incomplete_data)
   mean_imp ← IMPUTE_MEAN(x,type = "columnwise")
16:  imp_old ← mean_imp
   iter ← 0
18:  dif ← ∞
while (iter < num_iter AND MAX(dif) > req_err) do
20:   imp_data ← CREATE_EMPTY_MATRIX_WITH_SAME_DIMENSIONS_AS(mean_imp)
   for i ← 1 TO NUMBER_OF_ROWS(incomplete_data) do
22:     loc_miss_inc ← LOCATION_OF_MISSING_VALUES_IN_ROW(incomplete_data[i,:])
     dummy_matrix ← CREATE_MATRIX_WITH_INCOMPLETE_ROW_REPLACED_BY_MEAN(incomplete_data[i,:],mean_imp)
24:     Est_Meth ← "moment"
     depth_function ← CALCULATE_DEPTH_FUNCTION(depth_fun,dummy_matrix,mean_imp,Est_Meth)
26:     max_depth_indices ← TOP_K_INDICES(depth_function,k)
     max_depth_points ← SELECT_POINTS_BY_INDICES(mean_imp,max_depth_indices)
28:     if k == 1 then
         imputed_val ← max_depth_points
      else
         imputed_val ← CALCULATE_COLUMN_MEANS(max_depth_points)
      end if
     imp_data[i,:] ← imputed_val
34:   end for
   missing_vals ← VALUES_AT_MISSING_INDICES(imp_data,miss_ind)
36:   imputed_data ← x
   imputed_data[location_miss] ← missing_vals
38:   dif ← ABSOLUTE_DIFFERENCE(imputed_data,mean_imp)
   mean_imp ← imputed_data
40:   iter ← iter + 1
end while
42:   return imputed_data
end function
  
```

3.3 Class Based Imputation

This method is specializes in imputing missing values in dataset which contain categorical variable.

Input Variables

- **data:** - Data Contain Missing Value(provided Data)
- **depth fun:** - Depth function guiding the imputation process.
- **k :** - Number of points with maximum depth considered.
- **num-iter :-** Maximum iterations allowed for imputation.
- **col-id:-** Column indices identifying categorical variables.

The function proceeded following these steps as:

- Iteratively, the function addresses each unique category within the categorical variable
- splits the dataset accordingly to the active category.
- With these steps, the function continued.removes the category variable from the segmented data and performs *iterative imputation*(Method 2) on it.
- went back into the original dataset with the imputed values added.

3.3.1 Algorithm 3

Algorithm 3 Impute_Class

```

function IMPUTE_COLUMNWISE(data,depth_fun,k,num_iter,col_id)
    imp_data      ← CREATE_MATRIX_WITH_ALL_ZEROS(rows           =
    NUMBER_OF_ROWS(data),cols = NUMBER_OF_COLUMNS(data))
3:   categories ← UNIQUE_VALUES_IN_COLUMNS(data,col_id)
    n.cat ← LENGTH(categories)
    for category in categories do
6:      category_data ← SELECT_ROWS_WHERE_COLUMN_EQUALS_CATEGORY(data,col_id[1],category)
      mod_data ← REMOVE_CATEGORY_COLUMNS(category_data,col_id)
      imp_class   ← iterative_imputation(mod_data,depth_fun,k,est_meth       =
      1,num_iter,req_err = 0.001)
9:      imp_data[SELECT_ROWS_WHERE_COLUMN_EQUALS_CATEGORY_INDEX(data,col_id[1],category),-col_imp_class
      imp_data[,col_id] ← SELECT_COLUMN_VALUES(data,col_id)
    end for
12:   return imp_data
end function

```

Chapter 4

Simulation Study

4.1 Data Description

For our performance study, we will be looking at two datasets from the *R package ddalpha*.

Name	Columns	Rows
Bupa	6	345
Banknoten	7	200

The **bupa** dataset consists of 345 records, each containing 6 attributes. These characteristics include age, gender, and drinking habits, in addition to blood test findings such as bilirubin, alkaline phosphate, and SGOT levels. Based on test findings and patient data, each record represents a patient and is intended to anticipate complications with the liver. On the other hand, the **banknoten** dataset, which is typically used for classification, includes 200 observations and 7 variables. It gives banknote dimensions in detail, including length, breadth, width, and diagonal length. Two extra columns provide more dimensional data. The difference between real and counterfeit banknotes occurs in the seventh column.

- Produced missing value by (*MCAR*)
- Missing percentage 5, 10, and 15
- Depth Function : *Mahalanobis*

4.2 Normalized Root Mean Squared Error

Normalized Root Mean Squared Error (NRMSE) is a measure of the differences between predicted and observed values in a dataset, normalized by the range, mean, or standard deviation of the observed values. It provides a dimensionless quantity that facilitates comparison across different datasets or models.

The formula for NRMSE is:

$$NRMSE = \sqrt{\frac{mean(X_{original} - X_{imputed})^2}{var(X_{original})}}$$

Where,

- $X_{imputed}$ represents the predicted matrix,
- $X_{original}$ represents the observed Data,
- n is the number of observations. (*without missing value*)

Why Normalize RMSE?

Normalizing RMSE allows for:

- **Comparison Across Different Scales:** Since NRMSE is dimensionless, it can be used to compare the performance of models across different datasets with varying scales.
- **Interpreting Error Relative to Data Scale:** It helps in understanding how significant the error is relative to the magnitude of the observed data. For instance, an RMSE of 5 might be acceptable for a dataset with values ranging from 0 to 100, but not for one ranging from 0 to 10.

Interpretation

- **NRMSE = 0:** Perfect model with no error.
- **$0 < NRMSE < 1$:** Error is less than the range/mean of the observed data.
- **$NRMSE > 1$:** Error exceeds the range/mean of the observed data, indicating a relatively poor model.

By providing a normalized metric, NRMSE offers a standardized way to assess model performance, facilitating easier comparison and interpretation of prediction errors.

4.3 Performance study bupa dataset

4.3.1 bupa dataset at 5% missing value

First we were taking the result on the bupa dataset at different percentage of missing data under *MCAR*.

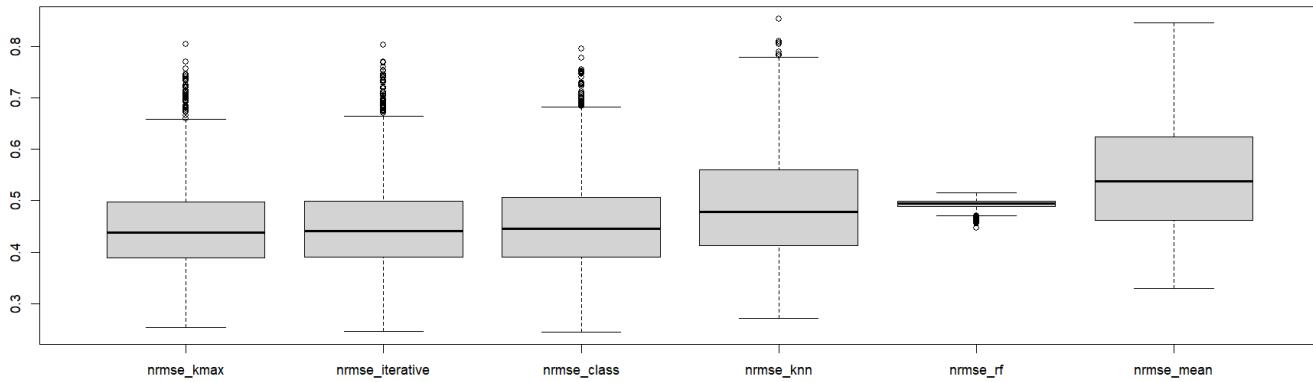


Figure 4.1: **bupa with 5% missing value**

Table 4.1: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.453497	0.455851	0.458311	0.495776	0.493108	0.548099

At 5% missing values under MCAR, the "K_max" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **0.453497**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.548099**.

4.3.2 Average NRMSE for Bupa data with 10% missing values

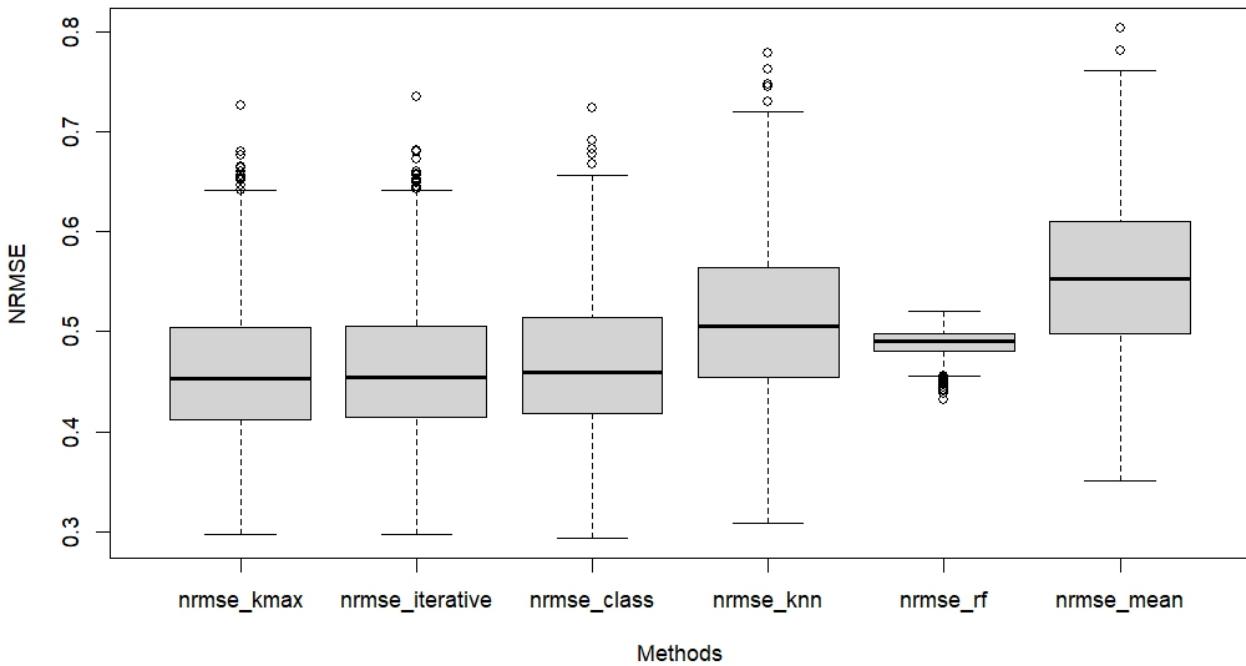


Figure 4.2: bupa with 10% missing value

Table 4.2: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.464624	0.46668	0.47095	0.51278	0.487748	0.555629

At 10% missing values under MCAR, the "K_max" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **0.464624**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.555629**

4.3.3 bupa dataset at 15% missing value

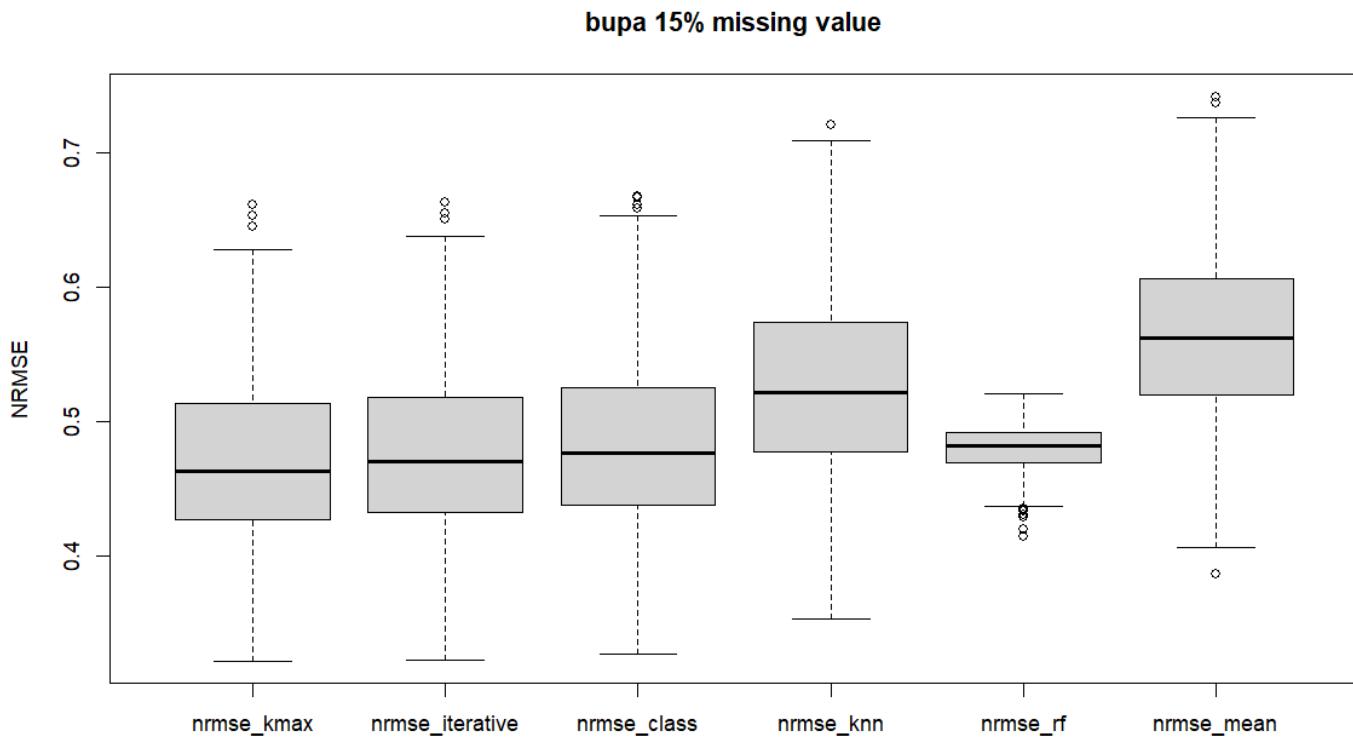


Figure 4.3: **bupa with 15% missing value**

Table 4.3: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.473875	0.477528	0.483791	0.526566	0.480133	0.56305

At 15% missing values under MCAR, the "K_max" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **0.473875**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.56305**

4.3.4 banknoten with 5% missing value

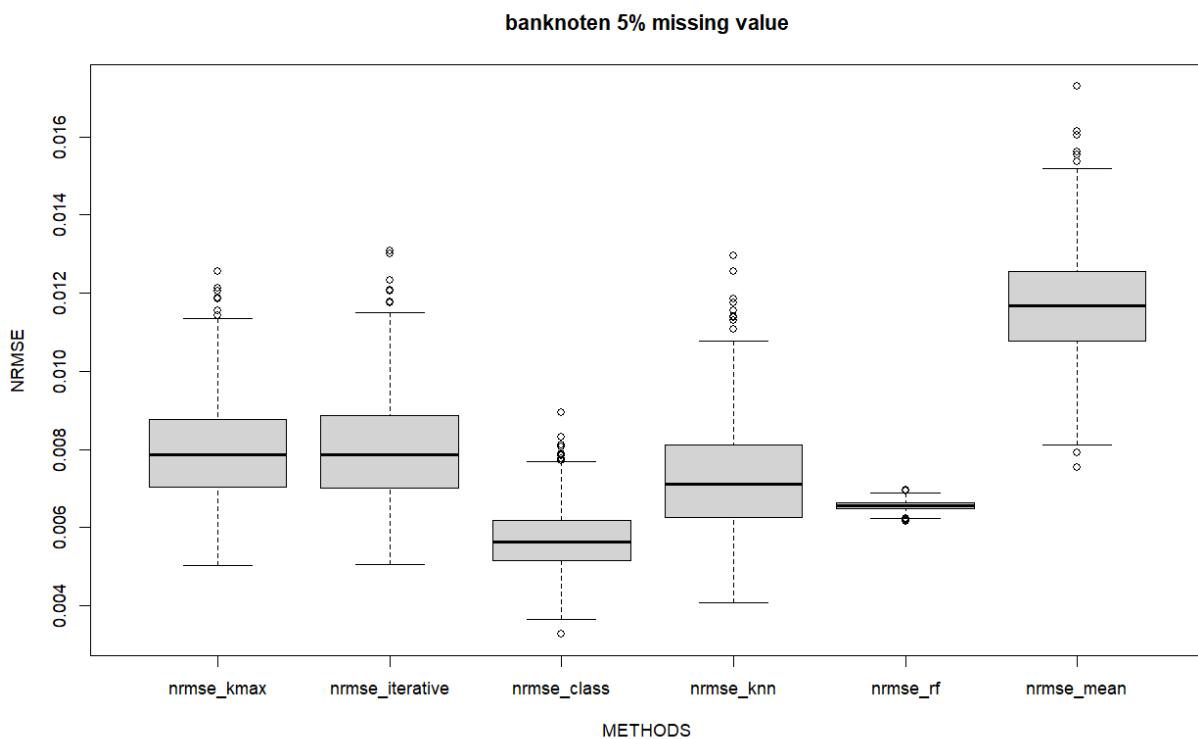


Figure 4.4: banknoten with 5% missing value

Table 4.4: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.007964	0.007971	0.005688	0.007226	0.006557	0.011695

The "IMPUTE CLASS" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **0.005688**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.011695**.

4.3.5 banknoten with 10% missing value

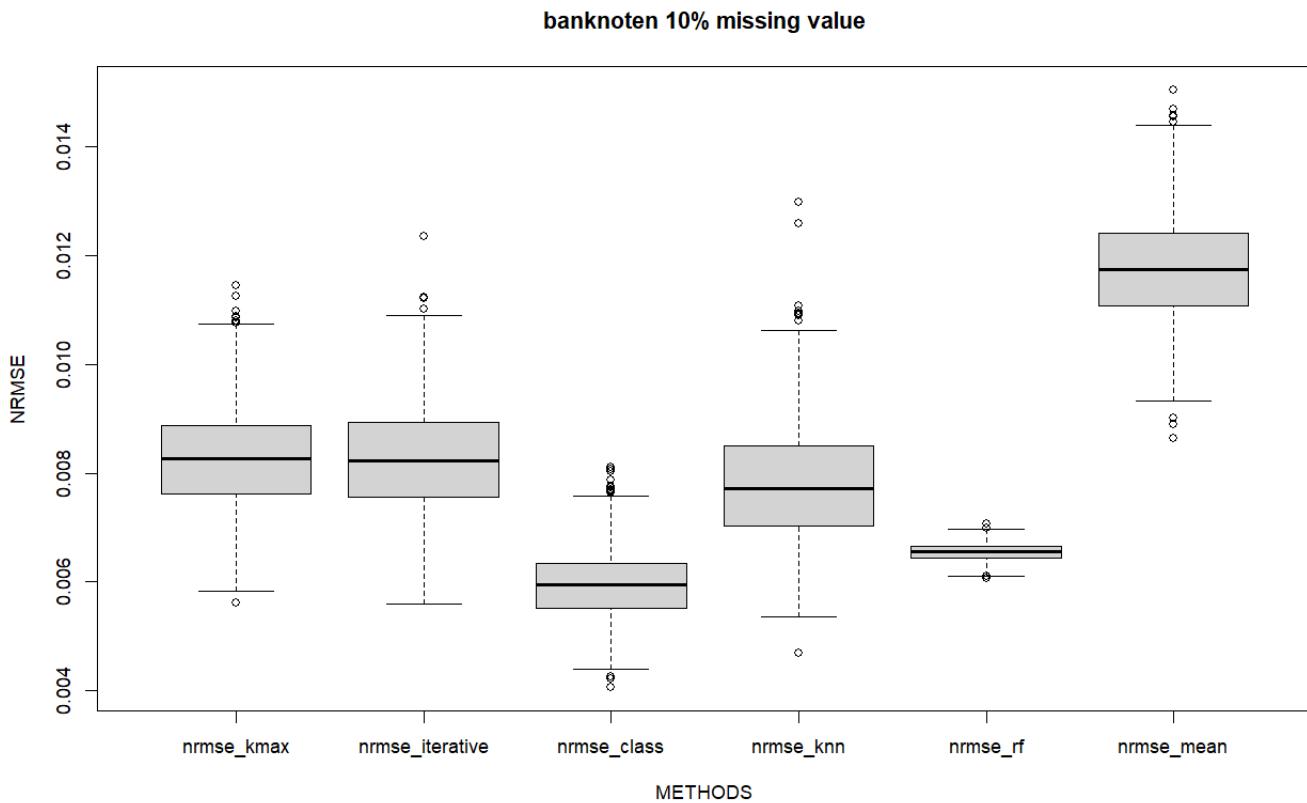


Figure 4.5: banknoten with 10% missing value

Table 4.5: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.008281	0.008267	0.005953	0.007795	0.006549	0.01173

The "IMPUTE CLASS" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **0.005953**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.01173**.

4.3.6 banknoten with 15% missing value

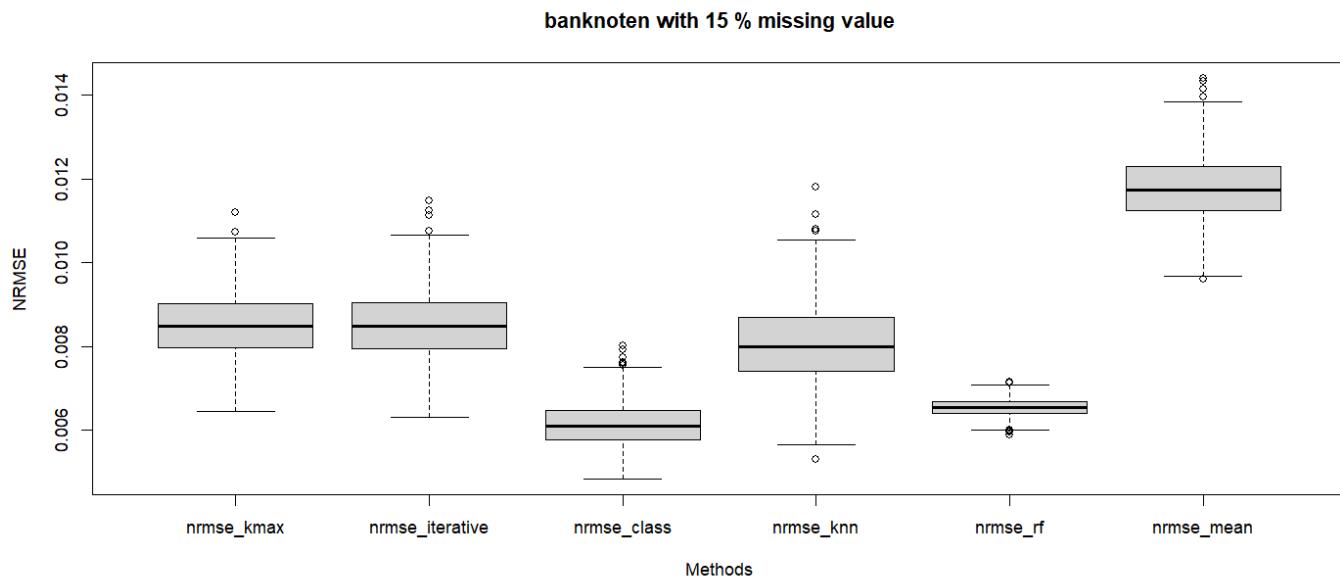


Figure 4.6: **banknoten with 15% missing value**

Table 4.6: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.008281	0.008267	0.005953	0.007795	0.006549	0.01173

The "IMPUTE CLASS" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **0.005953**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.01173**.

4.3.7 Normal distribution with 10% missing value

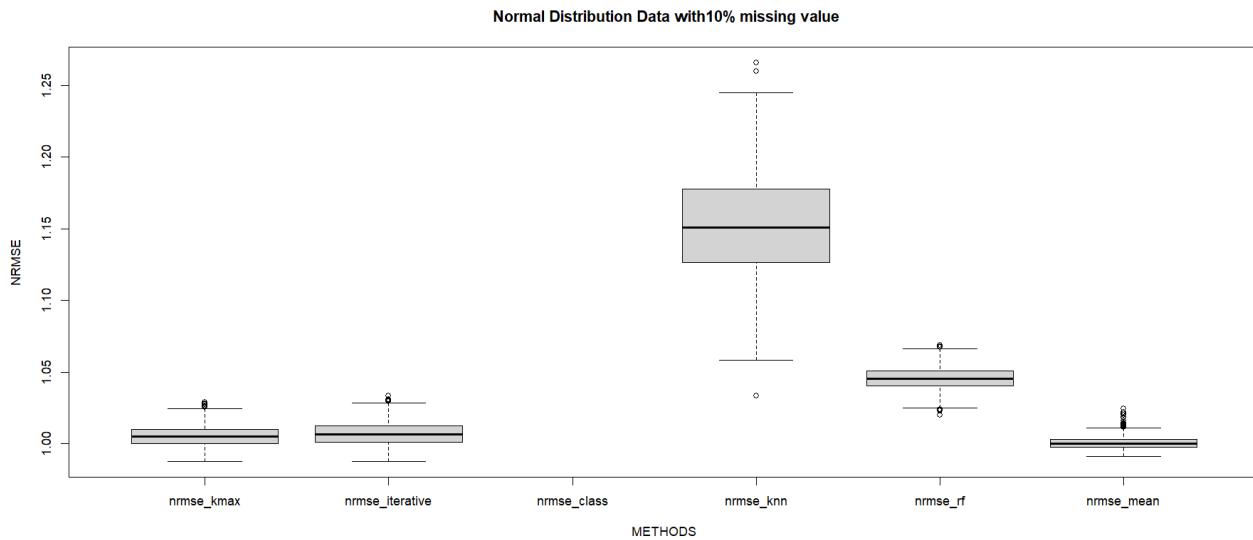


Figure 4.7: Normal distribution with 10% missing value

Here we are considering the data generated from multivariate normal distribution with parameter with parameter $\mu = (0, 0, 0, 0, 0)$ & $\Sigma = \text{diag}(1, 1, 1, 1, 1)$

Table 4.7: AVERAGE NRMSE

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	1.00663	1.006946	—	1.15425	1.04546081	1.00942

The "k-max" approach appears to be the most accurate of the methods mentioned for the given data, as seen by its lowest NRMSE score of **1.00663**. The least accurate approach for this dataset is the "MEAN IMPUTATION" method, which has the greatest NRMSE value **0.01173**

4.3.8 Time Comparison

We saw that there are not many variations in the average NRMSE (Normalized Root Mean Squared Error) of our created approach and Random Forest, among other methods, from Figures 4.1–4.7. In light of this, we also examine each method's efficiency with respect to computational time, which is the second factor in our analysis.

We determine the average time required for imputation by each approach in order to evaluate computational efficiency. We compare the imputation time required by different approaches in an effort to learn more about each method's usefulness than just its predictive power. This assessment is essential to find the approach that generates correct data and works well under reasonable time limitations. Here we are considering the data generated from multivariate normal distribution with parameter with parameter $\mu = (0, 0, 0, 0, 0)$ & $\Sigma = \text{diag}(1, 1, 1, 1, 1)$

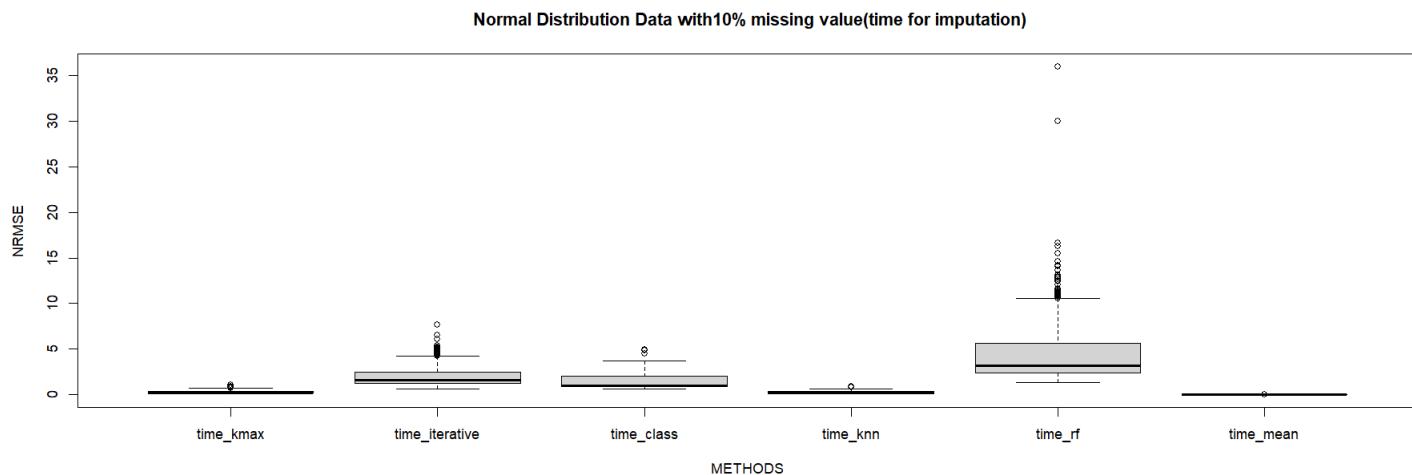


Figure 4.8: Normal distribution with 10% missing value

Table 4.8: AVERAGE Time for Imputation

METHOD	K-MAX	ITERATIVE IMPUTATION	IMPUTE CLASS	KNN	RANDOM FOREST	MEAN IMPUTATION
NRMSE	0.265664	1.97226	1.3320110	0.22018676	4.307334	0.0037

Chapter 5

Conclusion

The results presented in the previous chapter provide a comprehensive overview of the imputation techniques and their performances.

Firstly, considering the results on the BUPA dataset, it is evident that *K-Max Method* yields the lowest Normalized Root Mean Square Error (NRMSE). This demonstrates that *K-Max Method* is the most effective method among both our proposed methods and the existing imputation techniques. Its superior performance indicates its robustness and reliability in handling missing data in the BUPA dataset. The effectiveness of *K-Max Method* can be attributed to its ability to leverage the underlying patterns in the data more efficiently, thereby minimizing the error.

Additionally, for the Banknotan dataset, which includes class labels, a different approach is required. Given the structured nature of this dataset, the *Class Based Imputation* method proves to be the most effective for imputing missing values. The *Class Based Imputation* approach is specifically designed to handle datasets with classes, making it particularly well-suited for the Banknotan data. This method ensures that the imputation process respects the underlying class structure, leading to more accurate and meaningful data reconstruction. The effectiveness of the *Class Based Imputation* technique resides in its ability to take into account inter-class relationships and variations, which are essential for preserving the integrity of datasets containing information that is categorical or class-specific.

In summary, the analysis shows that *K-Max Method* is the best overall imputation technique for general datasets like BUPA, while *Class Based Imputation* is optimal for datasets with class structures, such as the Banknotan data. These findings emphasize the importance of selecting imputation methods that align with the specific characteristics of the dataset to achieve the best results.

Our proposed methods have demonstrated superior performance compared to widely used techniques like mean imputation, kNN imputation, and random forest imputation. They not only exhibit higher accuracy, evidenced by lower RMSE values across different levels of missing data, but also perform more efficiently in terms of computation time.

K-Max Method significantly outperforms traditional methods on the BUPA dataset, while *Class Based Imputation* effectively handles datasets with class structures, preserving class-specific information. The reduced computational time required for our methods further highlights their practicality for large-scale applications.

In conclusion, our proposed methods offer both better accuracy and significant time savings, making them advantageous over traditional imputation techniques. This dual benefit underscores their potential for adoption in various data-driven fields, enhancing the quality and efficiency of data analysis.

Bibliography

- Beretta, L. and Santaniello, A. (2016). Nearest neighbor imputation algorithms: a critical evaluation. *BMC medical informatics and decision making*, 16:197–208.
- Donders, A. R. T., Van Der Heijden, G. J., Stijnen, T., and Moons, K. G. (2006). A gentle introduction to imputation of missing values. *Journal of clinical epidemiology*, 59(10):1087–1091.
- Little, R. J. and Rubin, D. B. (2019). *Statistical analysis with missing data*, volume 793. John Wiley & Sons.
- Liu, R. Y. (1990). On a notion of data depth based on random simplices. *The Annals of Statistics*, pages 405–414.
- Mahalanobis, P. C. (2018). On the generalized distance in statistics. *Sankhyā: The Indian Journal of Statistics, Series A* (2008-), 80:S1–S7.
- Mozharovskyi, P., Josse, J., and Husson, F. (2019). Nonparametric imputation by data depth. *Journal of the American Statistical Association*.
- Neto, M. R. (2008). The concept of depth in statistics. *Tech. rep.*
- Pujianto, U., Wibawa, A. P., Akbar, M. I., et al. (2019). K-nearest neighbor (k-nn) based missing data imputation. In *2019 5th International Conference on Science in Information Technology (ICSITech)*, pages 83–88. IEEE.
- Stekhoven, D. J. and Bühlmann, P. (2012). Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118.
- 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.-F., Li, M.-L., and Lin, W.-C. (2018). A class center based approach for missing value imputation. *Knowledge-Based Systems*, 151:124–135.
- Tukey, J. W. (1975). Mathematics and the picturing of data. In *Proceedings of the International Congress of Mathematicians, Vancouver, 1975*, volume 2, pages 523–531.

Vardi, Y. and Zhang, C.-H. (2000). The multivariate 1 1-median and associated data depth. *Proceedings of the National Academy of Sciences*, 97(4):1423–1426.

Zuo, Y. and Serfling, R. (2000). General notions of statistical depth function. *Annals of statistics*, pages 461–482.

5.1 Appendices

5.1.1 Used R-Packages

- ddalpha
- missForest
- VIM
- MASS
- imputeR
- writexl

5.1.2 R-code

K-Max Method

```

1 missing_info_kmax = function(x,mid,depth_fun, k) {
2   x = as.matrix(x)
3   mid=as.matrix(mid)
4   row_miss = which(apply(is.na(x), 1, any))
5   location_miss = which(is.na(x), arr.ind=TRUE)
6   location_non_miss = which(!is.na(x), arr.ind=TRUE)
7
8   incomplete_rows = apply(is.na(x), 1, any)
9   complete_data = x[!incomplete_rows,,drop=FALSE]
10  incomplete_data = x[incomplete_rows,,drop=FALSE]
11
12  miss_ind = which(is.na(incomplete_data), arr.ind=TRUE)
13
14  #imp_data = matrix(0, nrow=nrow(mid), ncol=ncol(mid))
15  imp_data=d1
16  for (i in 1:nrow(incomplete_data)) {
17
18    loc_miss_inc = which(is.na(incomplete_data[i,]) == 1)
19    dummy_matrix = matrix(rep(as.vector(incomplete_data[i,])), nrow=
20      nrow(mid), byrow=TRUE, ncol=ncol(mid))
21    dummy_matrix[, loc_miss_inc] = mid[, loc_miss_inc]
22
23    Est_Meth = "moment"
24
25
26    depth_function = switch(depth_fun,
27      depth.Mahalanobis(dummy_matrix, mid, mah.
28        estimate=Est_Meth),
29      depth.spatial(dummy_matrix,mid, mah.
30        estimate=Est_Meth),
31
32
33
34
35
36
37
38
39
39
40
41
42
43
44
45
46
47
48
49
49
50
51
52
53
54
55
56
57
58
59
59
60
61
62
63
64
65
66
67
68
69
69
70
71
72
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
89
90
91
92
93
94
95
96
97
98
99
99
100
101
102
103
104
105
106
107
108
109
109
110
111
112
113
114
115
116
117
118
119
119
120
121
122
123
124
125
126
127
128
129
129
130
131
132
133
134
135
136
137
138
139
139
140
141
142
143
144
145
146
147
148
149
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
169
170
171
172
173
174
175
176
177
178
179
179
180
181
182
183
184
185
186
187
187
188
189
189
190
191
192
193
194
195
196
197
197
198
199
199
200
201
202
203
204
205
206
207
207
208
209
209
210
211
212
213
214
215
215
216
217
217
218
219
219
220
221
222
223
223
224
225
225
226
227
227
228
229
229
230
231
232
232
233
233
234
234
235
235
236
236
237
237
238
238
239
239
240
240
241
241
242
242
243
243
244
244
245
245
246
246
247
247
248
248
249
249
250
250
251
251
252
252
253
253
254
254
255
255
256
256
257
257
258
258
259
259
260
260
261
261
262
262
263
263
264
264
265
265
266
266
267
267
268
268
269
269
270
270
271
271
272
272
273
273
274
274
275
275
276
276
277
277
278
278
279
279
280
280
281
281
282
282
283
283
284
284
285
285
286
286
287
287
288
288
289
289
290
290
291
291
292
292
293
293
294
294
295
295
296
296
297
297
298
298
299
299
300
300
301
301
302
302
303
303
304
304
305
305
306
306
307
307
308
308
309
309
310
310
311
311
312
312
313
313
314
314
315
315
316
316
317
317
318
318
319
319
320
320
321
321
322
322
323
323
324
324
325
325
326
326
327
327
328
328
329
329
330
330
331
331
332
332
333
333
334
334
335
335
336
336
337
337
338
338
339
339
340
340
341
341
342
342
343
343
344
344
345
345
346
346
347
347
348
348
349
349
350
350
351
351
352
352
353
353
354
354
355
355
356
356
357
357
358
358
359
359
360
360
361
361
362
362
363
363
364
364
365
365
366
366
367
367
368
368
369
369
370
370
371
371
372
372
373
373
374
374
375
375
376
376
377
377
378
378
379
379
380
380
381
381
382
382
383
383
384
384
385
385
386
386
387
387
388
388
389
389
390
390
391
391
392
392
393
393
394
394
395
395
396
396
397
397
398
398
399
399
400
400
401
401
402
402
403
403
404
404
405
405
406
406
407
407
408
408
409
409
410
410
411
411
412
412
413
413
414
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
```

```

29         depth.projection(dummy_matrix, mid),
30         depth.halfspace(dummy_matrix, mid),
31         depth.simplicial(dummy_matrix, mid))
32
33 max_depth_indices = order(depth_function, decreasing=TRUE)[1:k]
34 max_depth_points = mid[max_depth_indices,]
35
36 if (k==1) {
37   imputed_value = max_depth_points[loc_miss_inc]
38 } else {
39   value = colMeans(max_depth_points)
40   imputed_value=value[loc_miss_inc]
41 }
42 #i=
43 row_miss_i=row_miss[i]
44 imp_data[row_miss_i,loc_miss_inc] = imputed_value
45 mid[row_miss_i,loc_miss_inc]=imputed_value
46 }
47 #missing_vals = imp_data[miss_ind]
48 #imputed_data = x
49 #imputed_data[location_miss] = missing_vals
50
51
52 return("imputed_data"=imputed_data)
53 }
```

Listing 5.1: R code for the K-Max Method

Iterative_Imputation

```

38     max_depth_indices =order(depth_function, decreasing = TRUE)[1:k
39         ]
40     max_depth_points= mean_imp[max_depth_indices, ]
41
42     if (k==1) {
43         imputed_val =max_depth_points
44     } else {
45         imputed_val=colMeans(max_depth_points)
46     }
47
48     imp_data[i, ] = imputed_val
49 }
50
51 missing_vals= imp_data[miss_ind]
52 imputed_data=x
53 imputed_data[location_miss]= missing_vals
54
55 dif = abs(imputed_data - mean_imp)
56
57 mean_imp= imputed_data
58 print(paste("Iteration:", iter, "Max difference:", max(dif)))
59 iter = iter + 1
60
61 }
62
63 return(imputed_data)
64 }
```

Listing 5.2: R code for the Iterative_Imputation

Impute_Class

```

1 Impute_class=function(data,depth_fun,k,num_iter){
2   #===== identify the column id of the categorical variable
3   col_id = which(sapply(data, function(x) {
4     is.factor(x) || is.character(x) || (is.numeric(x) && length(
5       unique(x)) < 5)
6   }))
7   print(col_id)
8   #=====
9   imp_data=matrix(0,nrow=nrow(data),ncol=ncol(data))
10  #=====
11  categories=unique(data[, col_id])
12  n.cat=length(unlist(unique(data[, col_id]))) # no of categories
13
14
15  for(j in 1:n.cat){
16    category_data=data[data[, col_id[1]] == categories[j], ]    #
17      categories wise data
18    mod_data=category_data[,-col_id]
19    imp_class=iterative_imputation(mod_data,depth_fun=depth_fun,k=k,
20      num_iter= num_iter,req_err = 0.001)
21    imp_data[data[, col_id[1]] == categories[j], -col_id]=imp_class
22    imp_data[,col_id]=data[,col_id]
23  }
24
25  return(imp_data)
}

```

Listing 5.3: R code for the Impute_Class