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 IV^{th} 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

% 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), Pujianto et al. (2019)). KNN imputation is a valuable technique for handling missing data estimating values based on the charecteristics 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 $D(./x)$ is bounded and non-negative, then it is a statistical depth function that maps R^d to R. Following are the important properties of the depth function

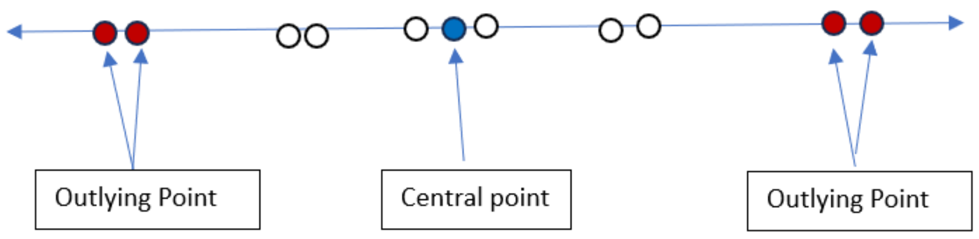
- **Affine invariant :** $D(y/Y) = D(Cy + d/CY + d)$ for any invertible $d \times d$ matrix C and any d belong to R^d.
- **Maximal at the symmetry centre:**
  $D(c/x) = \sup_{x \in \mathbb{R}^d} D(x/X)$ for any X half-space symmetry around c.
- **Monotone With respect to the deepest point**
  For any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

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

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

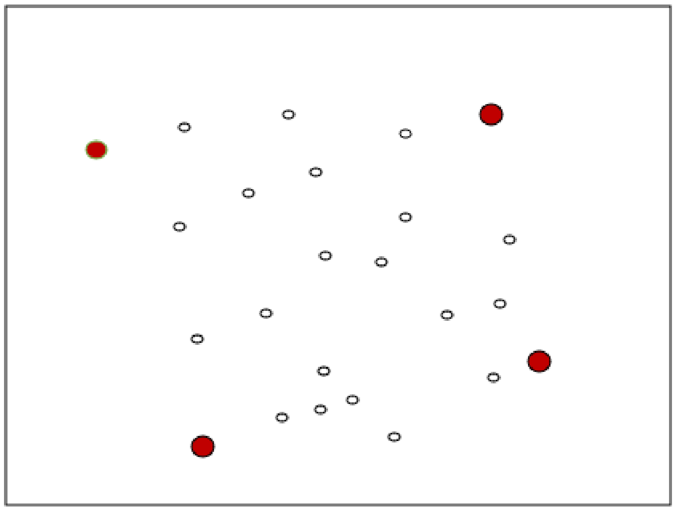
For symmetric distribution the maximum depth is 1/2 .

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

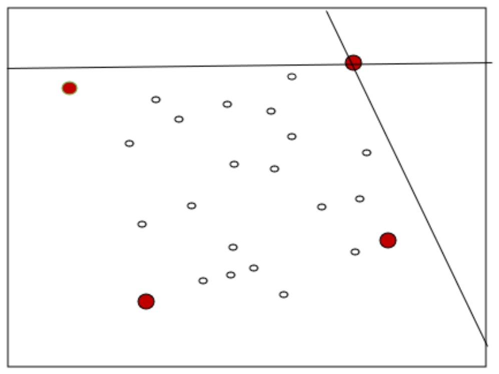
2.3 Depth In Two or more Dimensions

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



One method is to visualise a line through x (hyperplanes for $dgeq3$) 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

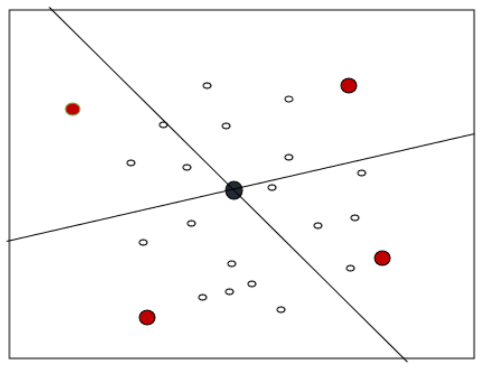

(b) fig:2

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 :

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

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 $X = (x_1, x_2, \ldots, x_n)^T$, 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

```
$x \leftarrow$ Data with Missing value
$mid \leftarrow$ Mean Imputed Matrix
$depth\_fun \leftarrow$ Depth Function
$k \leftarrow$ k point which having maximum depth
function MISSING_INFO_KMAX($x, mid, depth\_fun, k$)
    $x \leftarrow$ MATRIX($x$)
    $mid \leftarrow$ MATRIX($mid$)
    $row\_miss \leftarrow$ ROWS_WITH_MISSING_VALUES($x$)
    $location\_miss \leftarrow$ LOCATION_OF_MISSING_VALUES($x$)
    $location\_non\_miss \leftarrow$ LOCATION_OF_NON_MISSING_VALUES($x$)
    $incomplete\_rows \leftarrow$ ROWS_WITH_ANY_MISSING_VALUES($x$)
    $complete\_data \leftarrow$ COMPLETE_ROWS($x$)
    $incomplete\_data \leftarrow$ INCOMPLETE_ROWS($x$)
    $miss\_ind \leftarrow$ LOCATION_OF_MISSING_VALUES($incomplete\_data$)
    $imp\_data \leftarrow$ CREATE_EMPTY_MATRIX_WITH_SAME_DIMENSIONS_AS($mid$)
    for $i \leftarrow 1$ to NUMBER_OF_ROWS($incomplete\_data$) do
        $loc\_miss\_inc \leftarrow$ LOCATION_OF_MISSING_VALUES_IN_ROW($incomplete\_data[i,]$)
        $dummy\_matrix \leftarrow$ CREATE_MATRIX_WITH_INCOMPLETE_ROW_REPLACED_BY_MID($incomplete\_data[i,], mid$)
        $Est\_Meth \leftarrow$ "$moment$"
        $depth\_function \leftarrow$ CALCULATE_DEPTH_FUNCTION($depth\_fun, dummy\_matrix, mid, Est\_Meth$)
        $max\_depth\_indices \leftarrow$ TOP_K_INDICES($depth\_function, k$)
        $max\_depth\_points \leftarrow$ SELECT_POINTS_BY_INDICES($mid, max\_depth\_indices$)
        if $k == 1$ then
            $imputed\_value \leftarrow$ VALUE_OF_MAX_DEPTH_POINT_FOR_MISSING_LOCATIONS($loc\_miss\_inc, max\_depth\_points$)
        else
            $value \leftarrow$ CALCULATE_COLUMN_MEANS($max\_depth\_points$)
            $imputed\_value \leftarrow$ VALUES_FOR_MISSING_LOCATIONS_IN_COLUMN_MEANS($loc\_miss\_inc, value$)
        end if
        $row\_miss\_i \leftarrow$ ROW_MISS_INDEX_FOR_ROW($i$)
        UPDATE_IMPUTED_DATA_AND_MID($row\_miss\_i, loc\_miss\_inc, imputed\_value, imp\_data, mid$)
    end for
    $imputed\_data \leftarrow imp\_data$
    return ($imputed\_data, err$)
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 $D(X_{\text{Dummy}}, X_{\text{Mid}})$.
  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

```
    $x \leftarrow$ Data with Missing value
 2: $mid \leftarrow$ Mean Imputed Matrix
    $depth\_fun \leftarrow$ Depth Function
 4: $k \leftarrow$ k point which having maximum depth
    $num\_iter \leftarrow$ Number of iteration
 6: $req_err \leftarrow$ Specified Errors
    function ITERATIVE_IMPUTATION($x, depth\_fun, k, num\_iter, req\_err$)
 8:     $x \leftarrow$ MATRIX($x$)
        $location\_miss \leftarrow$ LOCATION_OF_MISSING_VALUES($x$)
10:     $incomplete\_row \leftarrow$ ROWS_WITH_ANY_MISSING_VALUES($x$)
        $incomplete\_col \leftarrow$ COLUMNS_WITH_ANY_MISSING_VALUES($x$)
12:     $complete\_data \leftarrow$ COMPLETE_ROWS($x$)
        $incomplete\_data \leftarrow$ INCOMPLETE_ROWS($x$)
14:     $miss\_ind \leftarrow$ LOCATION_OF_MISSING_VALUES($incomplete\_data$)
        $mean\_imp \leftarrow$ IMPUTE_MEAN($x, type =$ "$columnwise$")
16:     $imp\_old \leftarrow mean\_imp$
        $iter \leftarrow 0$
18:     $dif \leftarrow \infty$
        while ($iter < num\_iter$ AND MAX($dif$) $> req\_err$) do
20:         $imp\_data \leftarrow$ CREATE_EMPTY_MATRIX_WITH_SAME_DIMENSIONS_AS($mean\_imp$)
            for $i \leftarrow 1$ TO NUMBER_OF_ROWS($incomplete\_data$) do
22:             $loc\_miss\_inc \leftarrow$ LOCATION_OF_MISSING_VALUES_IN_ROW($incomplete\_data[i,]$)
                $dummy\_matrix \leftarrow$ CREATE_MATRIX_WITH_INCOMPLETE_ROW_REPLACED_BY_MEAN($incomplete\_data[i,], mean\_imp$)
24:             $Est\_Meth \leftarrow$ "$moment$"
                $depth\_function \leftarrow$ CALCULATE_DEPTH_FUNCTION($depth\_fun, dummy\_matrix, mean\_imp, Est\_Meth$)
26:             $max\_depth\_indices \leftarrow$ TOP_K_INDICES($depth\_function, k$)
                $max\_depth\_points \leftarrow$ SELECT_POINTS_BY_INDICES($mean\_imp, max\_depth\_indices$)
28:             if $k == 1$ then
                    $imputed\_val \leftarrow max\_depth\_points$
30:             else
                    $imputed\_val \leftarrow$ CALCULATE_COLUMN_MEANS($max\_depth\_points$)
32:             end if
                $imp\_data[i,] \leftarrow imputed\_val$
34:         end for
            $missing\_vals \leftarrow$ VALUES_AT_MISSING_INDICES($imp\_data, miss\_ind$)
36:         $imputed\_data \leftarrow x$
            $imputed\_data[location\_miss] \leftarrow missing\_vals$
38:         $dif \leftarrow$ ABSOLUTE_DIFFERENCE($imputed\_data, mean\_imp$)
            $mean\_imp \leftarrow imputed\_data$
40:         $iter \leftarrow iter + 1$
        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($data, depth\_fun, k, num\_iter, col\_id$)
    $imp\_data \leftarrow$ CREATE_MATRIX_WITH_ALL_ZEROS(rows = NUMBER_OF_ROWS($data$), cols = NUMBER_OF_COLUMNS($data$))
    $categories \leftarrow$ UNIQUE_VALUES_IN_COLUMNS($data, col\_id$)
    $n.cat \leftarrow$ LENGTH($categories$)
    for category in $categories$ do
        $category\_data \leftarrow$ SELECT_ROWS_WHERE_COLUMN_EQUALS_CATEGORY($data, col\_id[1]$, category)
        $mod\_data \leftarrow$ REMOVE_CATEGORY_COLUMNS($category\_data, col\_id$)
        $imp\_class \leftarrow$ iterative_imputation($mod\_data, depth\_fun, k, est\_meth = 1, num\_iter, req\_err = 0.001$)
        $imp\_data$[SELECT_ROWS_WHERE_COLUMN_EQUALS_CATEGORY_INDEX($data, col\_id[1]$, category), $-col\_$
imp_class
        $imp\_data[, col\_id] \leftarrow$ SELECT_COLUMN_VALUES($data, col\_id$)
    end for
    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*.

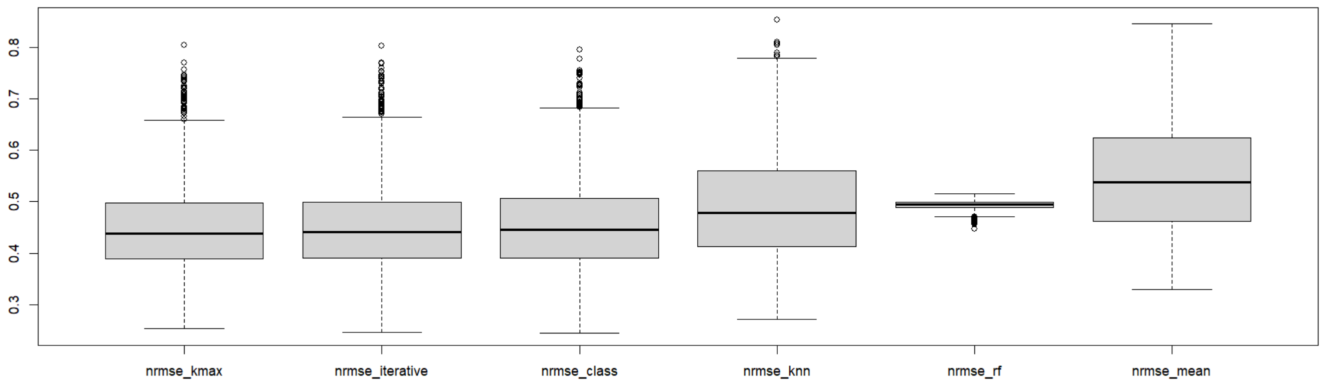


Figure 4.1: **bupa with** 5% **missing value**

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

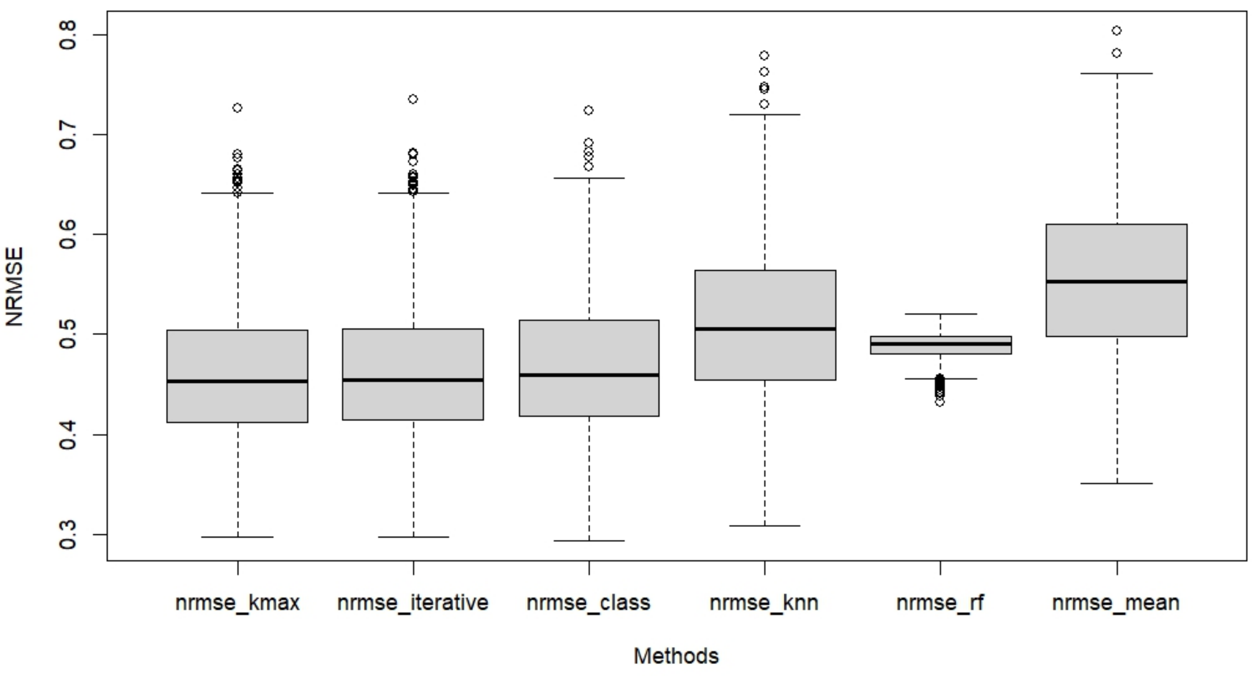


Figure 4.2: **bupa with** 10% **missing value**

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	0.55629

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

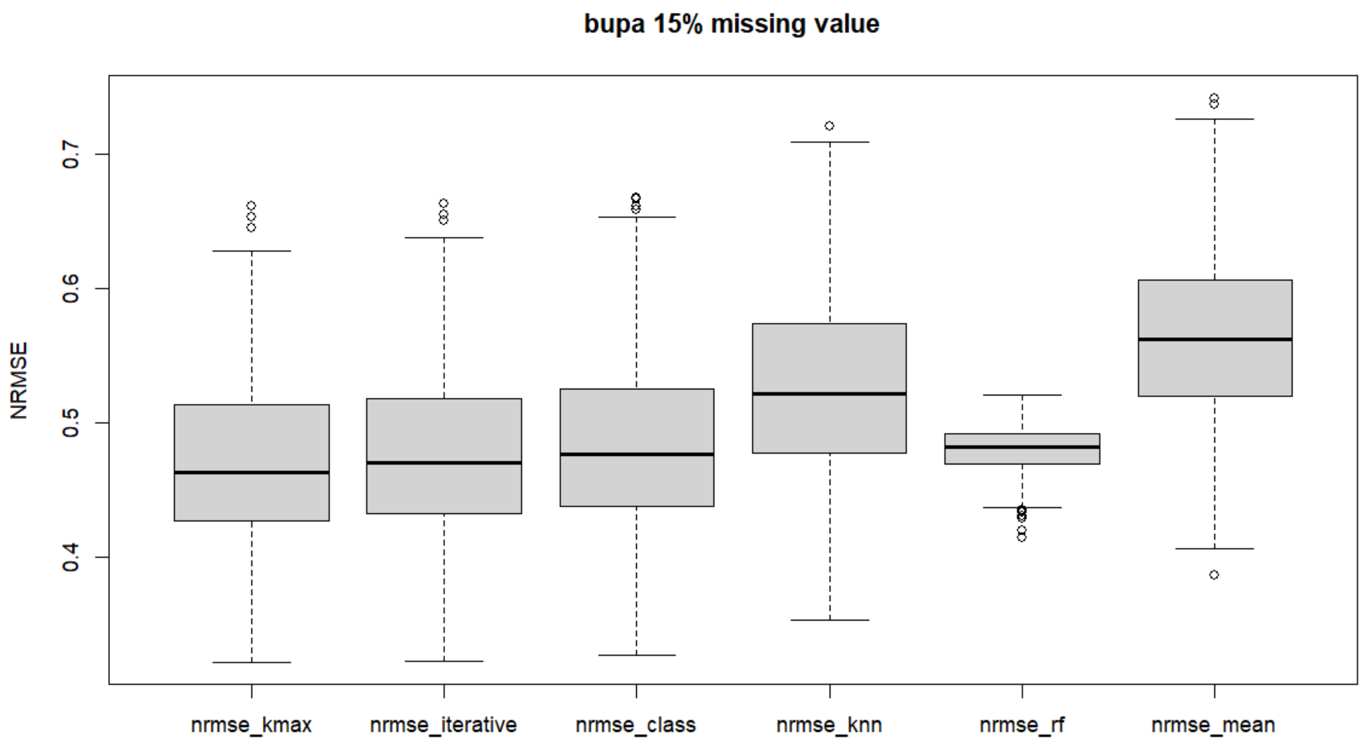


Figure 4.3: **bupa with** 15% **missing value**

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

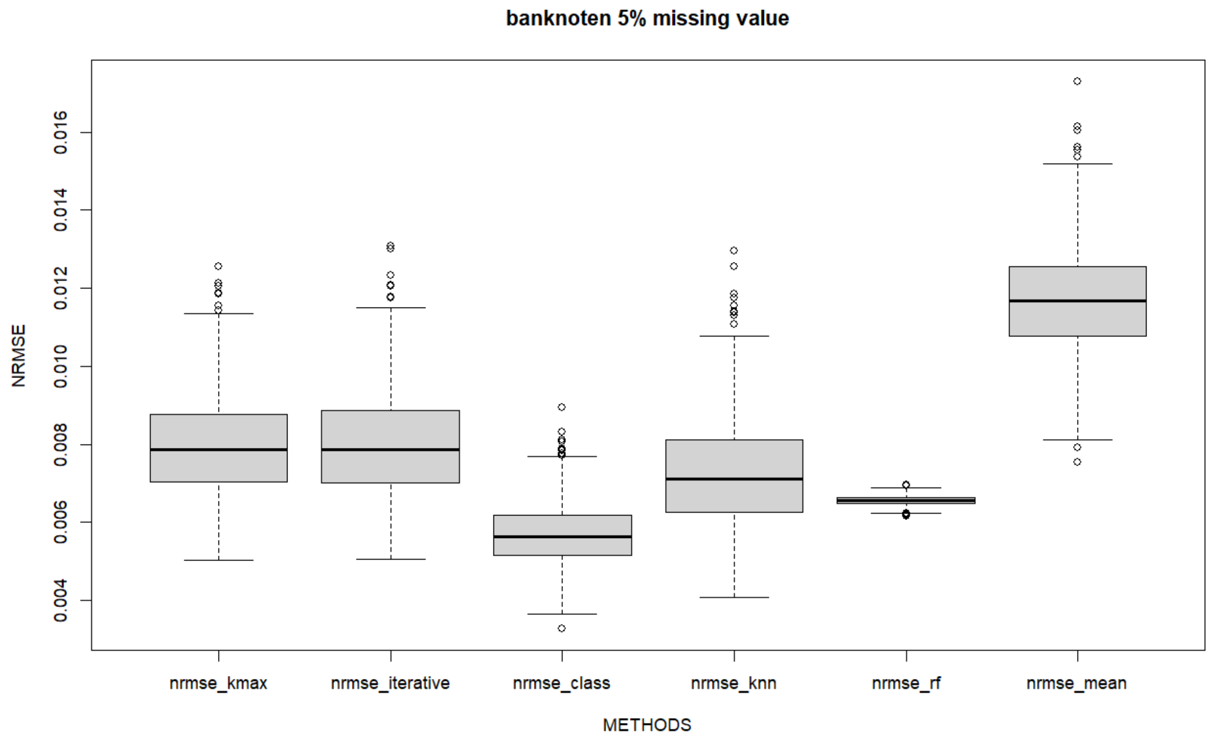


Figure 4.4: **banknoten with 5% missing value**

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

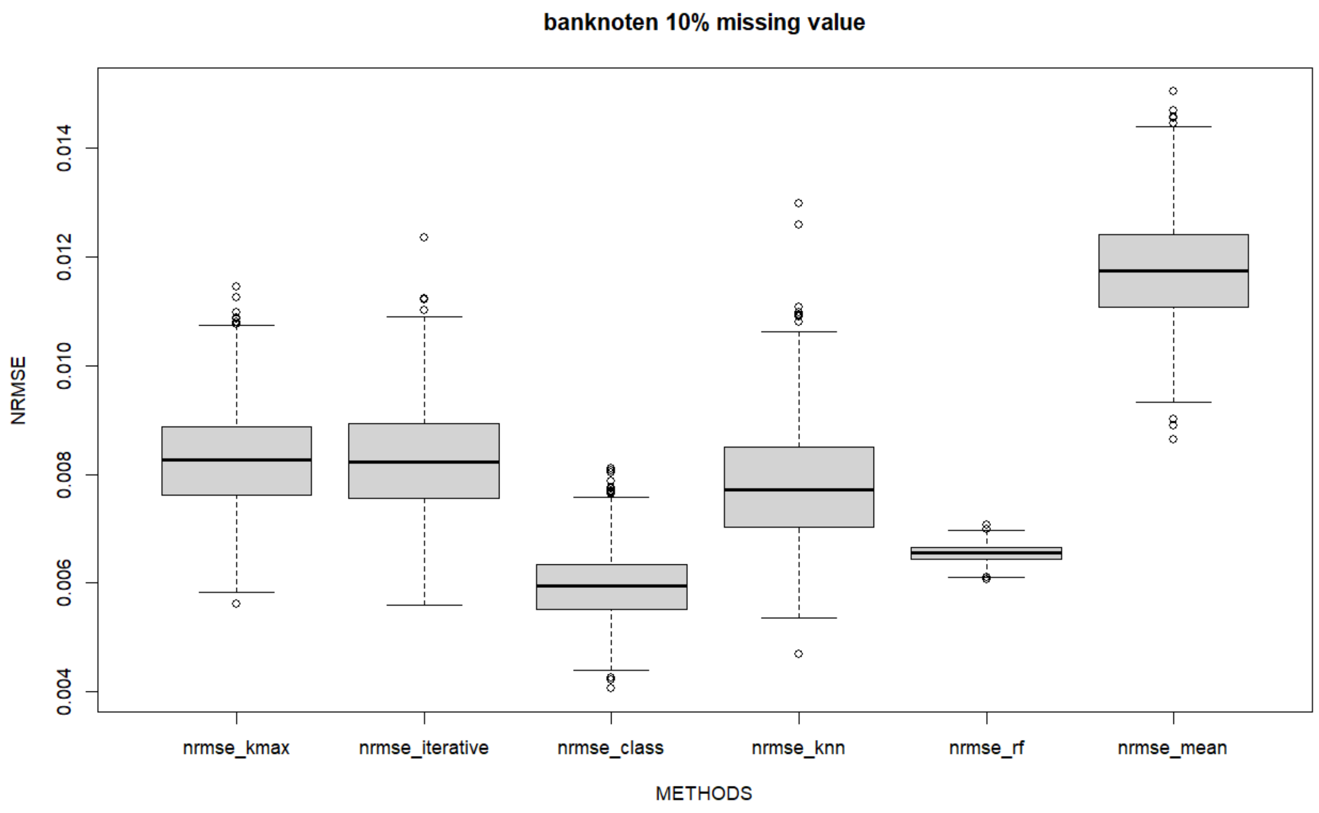


Figure 4.5: **banknoten with** 10% **missing value**

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

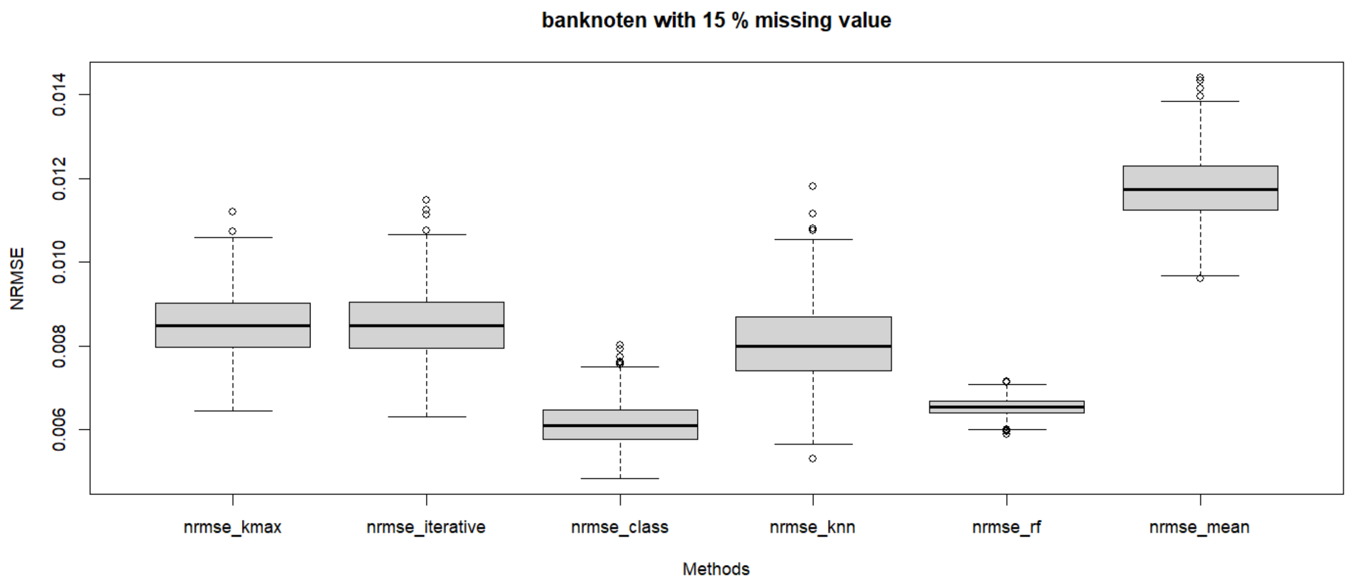


Figure 4.6: **banknoten with 15% missing value**

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

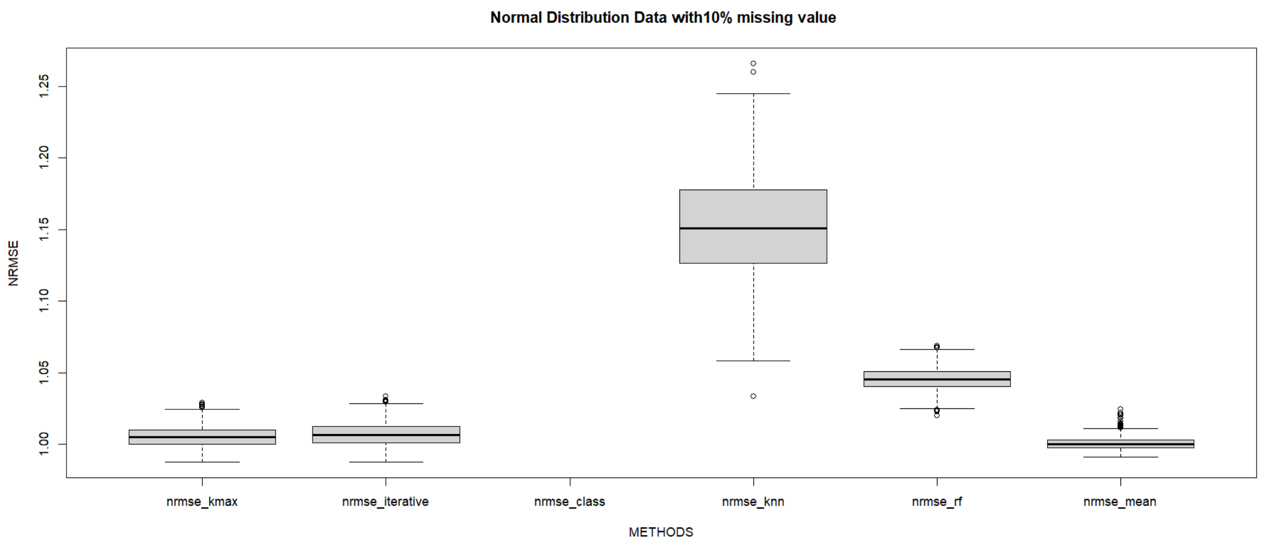


Figure 4.7: **Normal distribution with 10% missing value**

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

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 Comparision

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 $\mu = (0,0,0,0,0)$ & $\Sigma = diag(1,1,1,1,1)$

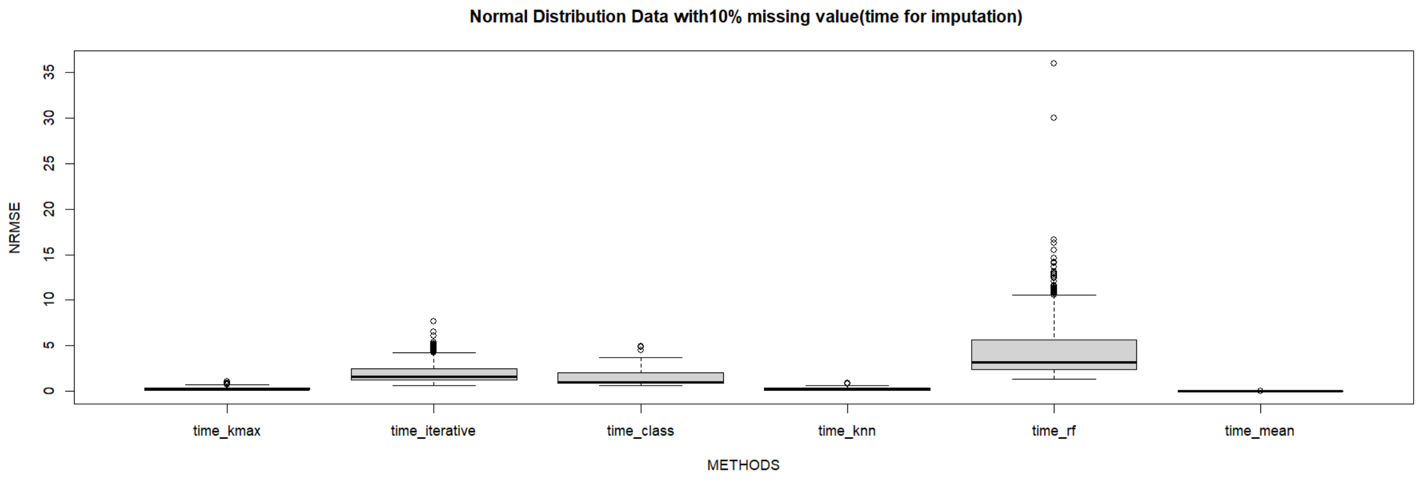


Figure 4.8: **Normal distribution with 10% missing value**

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 l 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

```r
missing_info_kmax = function(x,mid,depth_fun, k) {
  x = as.matrix(x)
  mid=as.matrix(mid)
  row_miss = which(apply(is.na(x), 1, any))
  location_miss = which(is.na(x), arr.ind=TRUE)
  location_non_miss = which(!is.na(x), arr.ind=TRUE)

  incomplete_rows = apply(is.na(x), 1, any)
  complete_data = x[!incomplete_rows,,drop=FALSE]
  incomplete_data = x[incomplete_rows,,drop=FALSE]

  miss_ind = which(is.na(incomplete_data), arr.ind=TRUE)

  #imp_data = matrix(0, nrow=nrow(mid), ncol=ncol(mid))
  imp_data=d1
  for (i in 1:nrow(incomplete_data)) {

    loc_miss_inc = which(is.na(incomplete_data[i,]) == 1)
    dummy_matrix = matrix(rep(as.vector(incomplete_data[i,])), nrow=
       nrow(mid), byrow=TRUE, ncol=ncol(mid))
    dummy_matrix[, loc_miss_inc] = mid[, loc_miss_inc]


      Est_Meth = "moment"


    depth_function = switch(depth_fun,
                            depth.Mahalanobis(dummy_matrix, mid, mah.
                               estimate=Est_Meth),
                            depth.spatial(dummy_matrix,mid, mah.
                               estimate=Est_Meth),
                            depth.projection(dummy_matrix, mid),
                            depth.halfspace(dummy_matrix, mid),
                            depth.simplicial(dummy_matrix, mid))

    max_depth_indices = order(depth_function, decreasing=TRUE)[1:k]
    max_depth_points = mid[max_depth_indices,]

    if (k==1) {
      imputed_value = max_depth_points[loc_miss_inc]
    } else {
      value = colMeans(max_depth_points)
      imputed_value=value[loc_miss_inc]
    }
    #i=
    row_miss_i=row_miss[i]
    imp_data[row_miss_i,loc_miss_inc ] = imputed_value
    mid[row_miss_i,loc_miss_inc]=imputed_value
  }
  #missing_vals = imp_data[miss_ind]
  #imputed_data = x
  #imputed_data[location_miss] = missing_vals


  return("imputed_data"=imputed_data)
}
```

Listing 5.1: R code for the K-Max Method

Iterative_Imputation

```

iterative_imputation = function(x, depth_fun, k, num_iter,req_err) {
  x = as.matrix(x)
  location_miss =which(is.na(x), arr.ind = TRUE)
  incomplete_row= apply(is.na(x), 1, any)
  incomplete_col = apply(is.na(x), 2, any)
  complete_data = x[!incomplete_row, , drop = FALSE]
  incomplete_data = x[incomplete_row, , drop = FALSE]

  miss_ind = which(is.na(incomplete_data), arr.ind = TRUE)

  #======mean imputed data / complete data
  mean_imp = missMethods::impute_mean(x, type = "columnwise")
  imp_old = mean_imp

  iter= 0
  dif = Inf

  while((iter < num_iter) & (max(dif) > req_err)) {
    imp_data = matrix(0, nrow = nrow(mean_imp), ncol = ncol(mean_imp)
       )

    for (i in 1:nrow(incomplete_data)) {
      loc_miss_inc = which(is.na(incomplete_data[i, ]) == 1)
      dummy_matrix = matrix(rep(as.vector(incomplete_data[i, ])),
         nrow = nrow(mean_imp),
                              byrow = TRUE, ncol = ncol(mean_imp))
      dummy_matrix[, loc_miss_inc] = mean_imp[, loc_miss_inc]


        Est_Meth = "moment"


      depth_function = switch(depth_fun,
                                depth.Mahalanobis = depth.Mahalanobis(
                                   dummy_matrix, mean_imp, mah.estimate
                                    = Est_Meth),
                                depth.spatial = depth.spatial(dummy_
                                   matrix, mean_imp, mah.estimate = Est
                                   _Meth),
                                depth.projection = depth.projection(
                                   dummy_matrix, mean_imp),
                                depth.halfspace = depth.halfspace(dummy
                                   _matrix, mean_imp),
                                depth.simplicial = depth.simplicial(
                                   dummy_matrix, mean_imp))
```

```

   max_depth_indices =order(depth_function, decreasing = TRUE)[1:k]
   max_depth_points= mean_imp[max_depth_indices, ]

   if (k==1) {
    imputed_val =max_depth_points
   } else {
    imputed_val=colMeans(max_depth_points)
   }

   imp_data[i, ] = imputed_val
  }

  missing_vals= imp_data[miss_ind]
  imputed_data=x
  imputed_data[location_miss]= missing_vals

  dif = abs(imputed_data - mean_imp)

  mean_imp= imputed_data
   print(paste("Iteration:", iter, "Max difference:", max(dif)))
  iter = iter + 1

 }

 return(imputed_data)
}
```

Listing 5.2: R code for the Iterative_Imputation

Impute_Class

```
Impute_class=function(data,depth_fun,k,num_iter){
  #======== identify the column id of the categorical variable
     ======================
  col_id = which(sapply(data, function(x) {
    is.factor(x) || is.character(x) || (is.numeric(x) && length(
       unique(x)) < 5)

  }))
  print(col_id)
  #======================================
  imp_data=matrix(0,nrow=nrow(data),ncol=ncol(data))
  #================
  categories=unique(data[, col_id])
  n.cat=length(unlist(unique(data[, col_id]))) # no of categories


  for(j in 1:n.cat){
    category_data=data[data[, col_id[1]] == categories[j], ]    #
       categories wise data
    mod_data=category_data[,-col_id]
    imp_class=iterative_imputation(mod_data,depth_fun=depth_fun,k=k,
       num_iter= num_iter,req_err = 0.001)
    imp_data[data[, col_id[1]] == categories[j], -col_id]=imp_class
    imp_data[,col_id]=data[,col_id]

  }
  return(imp_data)

}
```

Listing 5.3: R code for the Impute_Class