Handling Missing Data in R

Akif Mustafa

2024-09-19

Introduction

What are Missing Values?

In real-world datasets, missing values are a common occurrence. They can result from various factors such as data entry errors, equipment malfunctions, or respondents not providing complete information in surveys. Missing data can create bias in analyses, reduce statistical power, and lead to invalid results if not handled properly.

Types of Missing Data

Missing data can occur for various reasons, and understanding the mechanism behind the missing data is crucial for choosing an appropriate handling method. There are three main types of missing data mechanisms:

1. **Missing Completely at Random (MCAR)**

In this case, the probability of a value being missing is independent of both observed and unobserved data. There is no systematic reason for the missing data, meaning the missingness is entirely random. Data is MCAR when the missing values are unrelated to any other variable in the dataset. For example, if a respondent in a survey forgets to answer a question due to distraction, the data could be classified as MCAR.

- **Effect on Analysis**: When data is MCAR, the missingness does not introduce any bias, and methods such as listwise deletion (omitting missing data) can be applied without serious consequences.

2. **Missing at Random (MAR)**

In MAR, the probability of a value being missing depends on observed data but not on the missing data itself. For instance, the likelihood of missing data in a medical study might depend on a participant's age or gender, but not on their health status, which is not observed. While the data is missing, its occurrence can be explained using other known variables.

- **Effect on Analysis**: If the missing data is MAR, more sophisticated imputation methods like regression imputation, kNN, or multiple imputation should be used. These methods use the observed data to make reasonable assumptions about the missing values.

3. **Missing Not at Random (MNAR)**

Data is MNAR when the probability of missingness is related to the unobserved data itself. This type of missingness is not random, and its presence often indicates a systematic pattern. For example, in a clinical trial, if patients with severe symptoms are more likely to drop out of the study, the missing data on these patients is not random and is directly related to their health status.

- **Effect on Analysis**: MNAR is the most challenging type of missing data to handle. Ignoring the mechanism behind the missing data can lead to biased results. Advanced modeling techniques and sensitivity analysis may be required to appropriately deal with MNAR data.

Project Objective

In this project, we will explore several methods for handling missing data, while keeping in mind the different types of missingness. Although our dataset will simulate missing data for learning purposes, understanding these types helps guide our choice of imputation methods in real-world scenarios.

Why is Handling Missing Data Important?

Handling missing values is crucial because they can significantly impact the accuracy of statistical models and data-driven decision-making. When we ignore or mishandle missing data, it can lead to biased estimates and faulty conclusions. Properly managing missing data helps ensure the validity of analyses and improves the overall quality of insights derived from the data.

Approaches to Handling Missing Data

There are various methods to deal with missing data, each with its advantages and limitations. This project will explore several common approaches to handle missing data, such as:
- **Omitting missing values**: Removing rows with missing data (not generally recommended).
- **Mean/median imputation**: Replacing missing values with the mean or median of the available data.
- **Linear regression imputation**: Predicting missing values using a regression model based on the relationships between other variables.
- **k-Nearest Neighbors (kNN) imputation**: Using the nearest neighbors to estimate missing values.
- **Multiple imputation**: A more robust method that generates multiple datasets with imputed values and combines results to improve estimates.

Project Objective

In this project, we will:
1. Introduce missing values to the dataset for learning purposes.
2. Visualize the missing data and assess its patterns.
3. Apply different imputation techniques to handle missing values.
4. Compare the effectiveness of these techniques through visualization and analysis.

Install and Load Relevant Packages

In order to handle missing data, we will need a few R packages. You can install them if you haven’t already, and then load them into your R session.

```
# Install necessary packages
install.packages("mice")
install.packages("VIM")
install.packages("Amelia")
install.packages("tidyverse")

# Load necessary libraries
library(mice)

library(Amelia)
```

Loading and Viewing the `birthwt` Data from the MASS Library

First, we will load the `birthwt` dataset from the `MASS` library and check for missing values.

```
library(MASS)
# Load and view birthwt data from MASS library
data("birthwt")
nrow(birthwt)  # Number of rows in the data
```

```
## [1] 189
```

```
sum(is.na(birthwt$bwt))  # Check how many missing observations are in the 'bwt' column
```

```
## [1] 0
```

Introducing Random Missing Values for Learning Purposes To demonstrate different methods for handling missing data, we will introduce some random missing values in the bwt (birthweight) column.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3
```

```
## Warning: package 'tidyr' was built under R version 4.3.3
```

```
## Warning: package 'purrr' was built under R version 4.3.3
```

```
## Warning: package 'lubridate' was built under R version 4.3.3
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.1
## ✓ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ✖ dplyr::select() masks MASS::select()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to be
come errors
```

```
library(VIM)
```

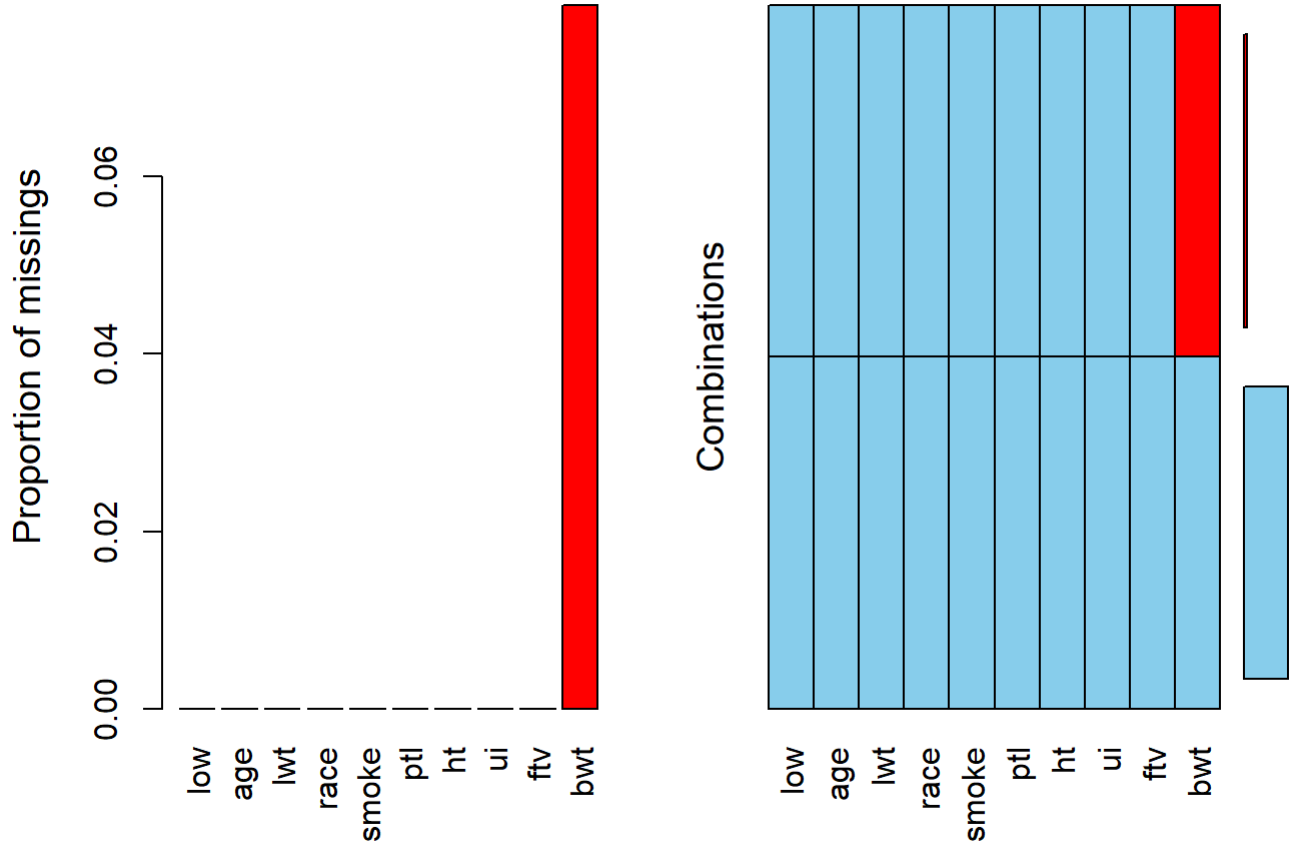
```
## Warning: package 'VIM' was built under R version 4.3.3
```

```
## Loading required package: colorspace
## Loading required package: grid
## VIM is ready to use.
##
## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues
##
## Attaching package: 'VIM'
##
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
set.seed(555)
miss_row = sample(nrow(birthwt), 15, replace = FALSE)

data1 = birthwt %>%
  mutate(bwt = replace(bwt, miss_row, NA))

# Check how many missing values are now present in the 'bwt' column
sum(is.na(data1$bwt))  # Now bwt has 15 missing values
```

```
## [1] 15
```

```
#Visualizing Missing Values
#We will use the aggr() function from the VIM package to visualize missing data in the datase
t.
aggr(data1)
```



Now, we will explore different techniques to handle missing data in the `bwt` column of our dataset.

Solution 1: Removing Missing Values (Not Recommended)

One of the simplest approaches is to remove rows with missing data. However, this method is not recommended, as it may lead to loss of valuable information.

```
# Removing Missing Values (Not Recommended)
data1 %>%
  drop_na(bwt) %>%
  head()  # Display the first few rows
```

```
##    low age lwt race smoke ptl ht ui ftv  bwt
## 85   0  19 182    2     0   0  0  1   0 2523
## 86   0  33 155    3     0   0  0  0   3 2551
## 87   0  20 105    1     1   0  0  0   1 2557
## 88   0  21 108    1     1   0  0  1   2 2594
## 89   0  18 107    1     1   0  0  1   0 2600
## 91   0  21 124    3     0   0  0  0   0 2622
```

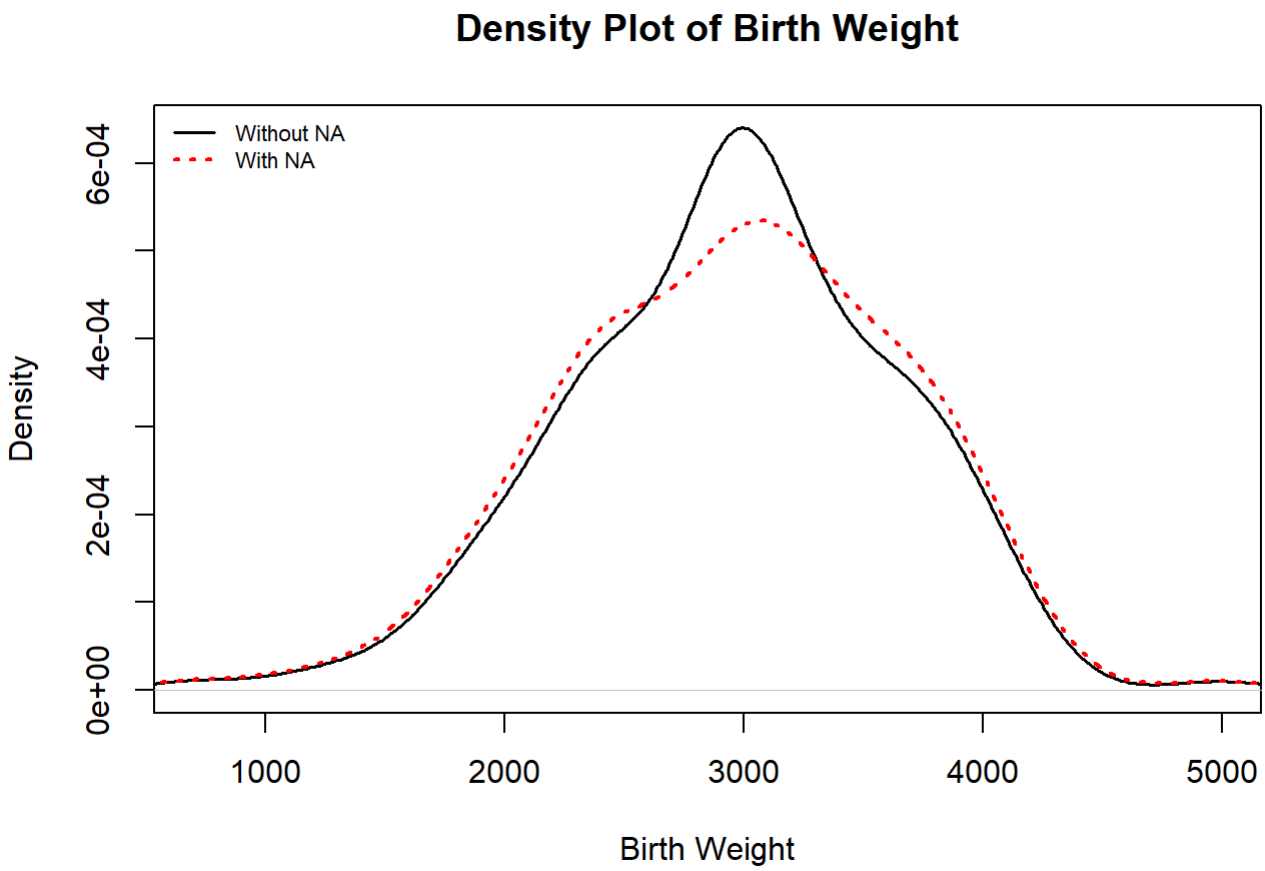
We did not modify the original dataset, as removing missing values often results in loss of data and can bias the results.

Solution 2: Replacing Missing Values with Mean/Median

A common method to handle missing data is to replace missing values with the mean or median of the available data. Let’s start by imputing the mean.

Replacing Missing Values with Mean

```
# Replace missing values in 'bwt' with the mean
m = mean(data1$bwt, na.rm = TRUE)

data2 = data1 %>%
  mutate(bwt = ifelse(is.na(bwt), m, bwt))

#We can visualize the distribution of birth weights before and after replacing missing values
using a density plot.

# Plot density with and without missing values
plot(density(data2$bwt),
     main = "Density Plot of Birth Weight",
     xlab = "Birth Weight",
     ylab = "Density",
     col = "black",
     lwd = 1.5,
     xlim = c(min(data2$bwt, na.rm = TRUE), max(data2$bwt, na.rm = TRUE)),
     ylim = c(0, max(c(density(data2$bwt)$y, density(data1$bwt, na.rm = TRUE)$y))))

# Add the second density plot (before imputation)
lines(density(data1$bwt, na.rm = TRUE),
      col = "red",
      lty = 3,
      lwd = 2)

# Add a legend
legend("topleft",
       c("Without NA", "With NA"),
       col = c("black", "red"),
       lty = c(1, 3),
       lwd = c(1.5, 2),
       bty = "n",
       cex = 0.7,
       pt.cex = 0.7)
```



Replacing Missing Values with Median

In cases where the data is skewed, it might be better to replace missing values with the median instead of the mean.

```
med = median(data1$bwt, na.rm = TRUE)

dataMed = data1 %>%
  mutate(bwt = replace(bwt, is.na(bwt), med))

# Check the new dataset
head(dataMed)
```

```
##    low age lwt race smoke ptl ht ui ftv  bwt
## 85   0  19 182    2     0   0  0  1   0 2523
## 86   0  33 155    3     0   0  0  0   3 2551
## 87   0  20 105    1     1   0  0  0   1 2557
## 88   0  21 108    1     1   0  0  1   2 2594
## 89   0  18 107    1     1   0  0  1   0 2600
## 91   0  21 124    3     0   0  0  0   0 2622
```

Solution 3: Linear Regression Imputation

Another approach to handling missing data is using linear regression to predict missing values based on other variables. In this method, we will use a regression model to predict the missing values in the `bwt` column using other available data.

Step 1: Running a Linear Regression Model

We will first convert the `race` variable into a factor and run a regression model excluding insignificant variables. After selecting the significant predictors, we will use the final model to predict missing values.

```
# Using linear regression for predicting missing values
data1$race = as.factor(data1$race)  # Convert 'race' to a factor variable

# Run initial regression (excluding the 'low' variable)
m1 <- lm(bwt ~ . -low, data = data1, na.action = na.omit)
summary(m1)
```

```
##
## Call:
## lm(formula = bwt ~ . - low, data = data1, na.action = na.omit)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1533.04  -437.43    36.73   471.31  1690.65
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2918.453    312.741   9.332  < 2e-16 ***
## age           -2.219      9.690  -0.229  0.81912
## lwt            3.850      1.760   2.187  0.03014 *
## race2       -439.718    151.716  -2.898  0.00427 **
## race3       -315.041    117.334  -2.685  0.00800 **
## smoke       -349.060    109.765  -3.180  0.00176 **
## ptl          -82.639    101.264  -0.816  0.41564
## ht          -636.029    202.094  -3.147  0.00196 **
## ui          -389.175    147.866  -2.632  0.00930 **
## ftv            7.177     46.675   0.154  0.87799
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 631.3 on 164 degrees of freedom
##   (15 observations deleted due to missingness)
## Multiple R-squared:  0.2219, Adjusted R-squared:  0.1792
## F-statistic: 5.198 on 9 and 164 DF,  p-value: 3.271e-06
```

```
# Remove insignificant variables from the model
m2 <- lm(bwt ~ lwt + race + smoke + ht + ui, data = data1, na.action = na.omit)
summary(m2)
```

```
##
## Call:
## lm(formula = bwt ~ lwt + race + smoke + ht + ui, data = data1,
##     na.action = na.omit)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1540.38  -432.72    56.18   447.56  1652.04
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2851.459    244.867  11.645  < 2e-16 ***
## lwt            3.955      1.682   2.352 0.019831 *
## race2       -434.420    146.317  -2.969 0.003427 **
## race3       -316.376    114.346  -2.767 0.006299 **
## smoke       -365.661    106.135  -3.445 0.000722 ***
## ht          -642.519    199.889  -3.214 0.001569 **
## ui          -414.771    142.254  -2.916 0.004036 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 627.1 on 167 degrees of freedom
##   (15 observations deleted due to missingness)
## Multiple R-squared:  0.2182, Adjusted R-squared:  0.1901
## F-statistic: 7.766 on 6 and 167 DF,  p-value: 2.227e-07
```

After running the regression, we will use this model to predict the missing values in the bwt column.

Step 2: Predicting Missing Values

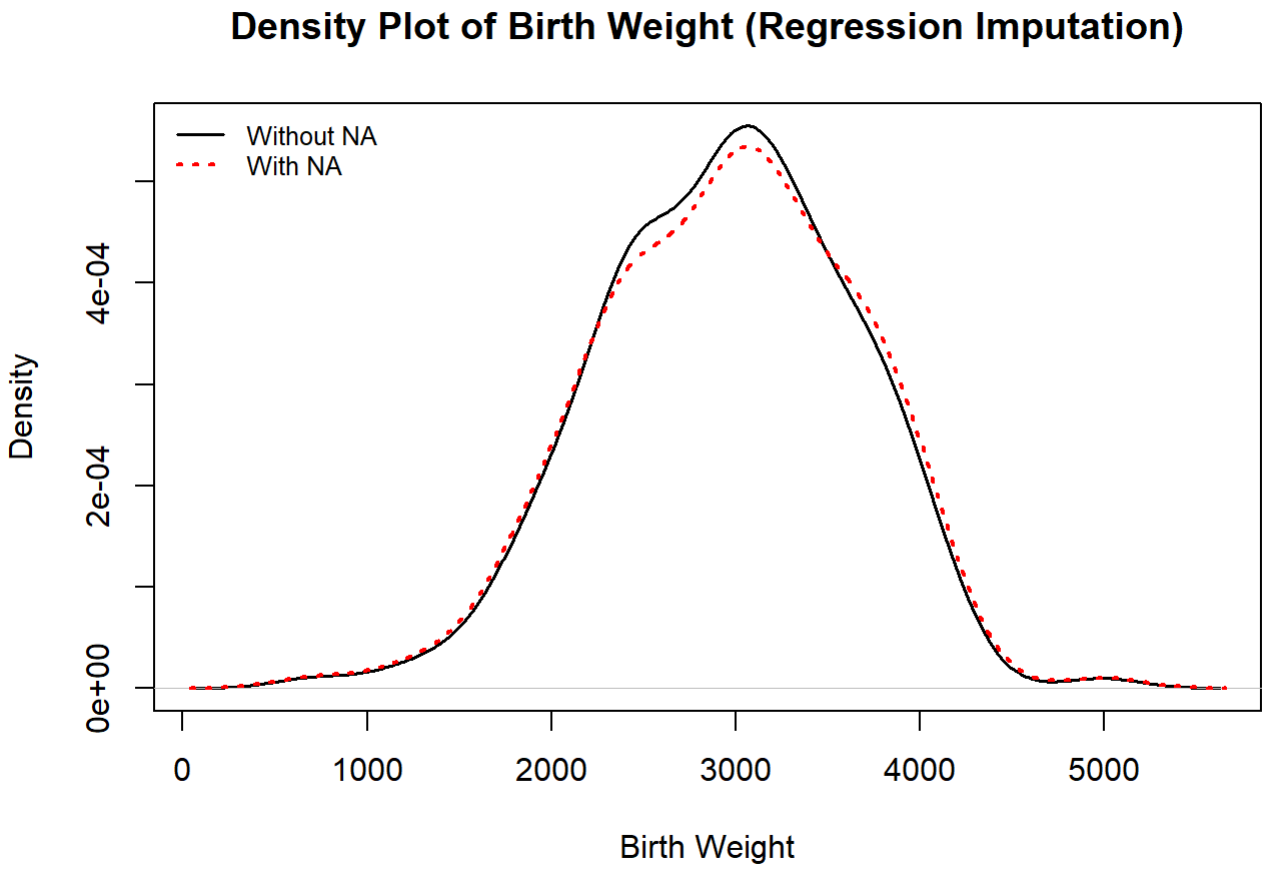
We will create a subset of the data that only contains rows with missing values in the bwt column, then use the regression model to predict those missing values.

```
# Identify rows with missing values in 'bwt'
na_rows <- which(is.na(data1$bwt))

# Create a dataset excluding the 'bwt' column for the rows with NA values
data_pred = data1[na_rows, -which(names(data1) == "bwt")]

# Predict missing 'bwt' values using the linear regression model
predicted_bwt = predict(m2, data_pred)

# Replace missing values in 'bwt' with the predicted values
data3 = data1 %>%
  mutate(bwt = replace(bwt, na_rows, predicted_bwt))
```

Step 3: Visualizing the Results

We can now compare the density plot of the bwt values after using linear regression imputation with the original dataset that contains missing values.

```
# Density plot after linear regression imputation
plot(density(data3$bwt),
     main = "Density Plot of Birth Weight (Regression Imputation)",
     xlab = "Birth Weight",
     ylab = "Density",
     col = "black",
     lwd = 1.5)

# Add the original density plot (with missing values)
lines(density(data1$bwt, na.rm = TRUE),
      col = "red",
      lty = 3,
      lwd = 2)

# Add a legend
legend("topleft",
       c("Without NA", "With NA"),
       col = c("black", "red"),
       lty = c(1, 3),
       lwd = c(1.5, 2),
       bty = "n",
       cex = 0.8,
       pt.cex = 0.8)
```



Solution 4: k-Nearest Neighbor (kNN) Imputation

The k-Nearest Neighbor (kNN) method is a non-parametric approach for imputing missing values. It estimates missing values based on the values of the nearest neighbors in the dataset. We will use the `VIM` package for this imputation method.

We will use the `kNN` function from the `VIM` package to impute the missing values in the `bwt` column. The parameter `k` specifies the number of nearest neighbors to consider.

```
# Imputation using kNN method
# We will use the VIM package for this
library(VIM)
data4 <- kNN(data1, k = 5, variable = "bwt")

# View the updated dataset with imputed values
head(data4)
```

```
##   low age lwt race smoke ptl ht ui ftv  bwt bwt_imp
## 1   0  19 182    2     0   0  0  1   0 2523   FALSE
## 2   0  33 155    3     0   0  0  0   3 2551   FALSE
## 3   0  20 105    1     1   0  0  0   1 2557   FALSE
## 4   0  21 108    1     1   0  0  1   2 2594   FALSE
## 5   0  18 107    1     1   0  0  1   0 2600   FALSE
## 6   0  21 124    3     0   0  0  0   0 2622   FALSE
```

Solution 5: Multiple Imputation Using MICE

Multiple imputation is a robust technique that involves generating several different imputed datasets and then combining the results to improve the accuracy of estimates. We will use the `mice` package to perform multiple imputation with the Predictive Mean Matching (PMM) method.

Step 1: Performing Multiple Imputation

We will use the `mice` function to perform multiple imputation on the dataset. The `m` parameter specifies the number of imputed datasets to generate.

```
# Load the mice package
library(mice)
```

```
## Warning: package 'mice' was built under R version 4.3.3
```

```
## Warning in check_dep_version(): ABI version mismatch:
## lme4 was built with Matrix ABI version 1
## Current Matrix ABI version is 0
## Please re-install lme4 from source or restore original 'Matrix' package
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
# Perform multiple imputation using Predictive Mean Matching (PMM)
impu_vals = mice(data1, method = "pmm", m = 5)
```

```
##
##  iter imp variable
##   1   1  bwt
##   1   2  bwt
##   1   3  bwt
##   1   4  bwt
##   1   5  bwt
##   2   1  bwt
##   2   2  bwt
##   2   3  bwt
##   2   4  bwt
##   2   5  bwt
##   3   1  bwt
##   3   2  bwt
##   3   3  bwt
##   3   4  bwt
##   3   5  bwt
##   4   1  bwt
##   4   2  bwt
##   4   3  bwt
##   4   4  bwt
##   4   5  bwt
##   5   1  bwt
##   5   2  bwt
##   5   3  bwt
##   5   4  bwt
##   5   5  bwt
```

```
# This command creates 5 datasets with different imputations
# You can choose any suitable one or take the mean of imputed values across the five iteratio
ns

# Imputing from the first imputed dataset
data5 = complete(impu_vals, 1)
```

Solution 6: Multiple Imputation Using Amelia

Multiple imputation using the `Amelia` package employs Expectation Maximization (EM) and Expectation Maximization with Bootstrapping (EMB) algorithms to handle missing data. This method is effective for dealing with missing values by generating multiple imputed datasets and combining the results to improve estimates.

Step 1: Preparing the Data

We need to remove any factor variables, such as `race` , before applying the `Amelia` method.

```
# Load the Amelia package
library(Amelia)
```

```
## Warning: package 'Amelia' was built under R version 4.3.3
```

```
## Loading required package: Rcpp
```

```
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.8.2, built: 2024-04-10)
## ## Copyright (C) 2005-2024 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
# Remove factor variable 'race' from the dataset
data_wo_race = data1 %>%
  dplyr::select(-race)

# Perform multiple imputation using Amelia
imps = amelia(data_wo_race, m = 5)
```

```
## -- Imputation 1 --
##
##   1  2  3
##
## -- Imputation 2 --
##
##   1  2  3
##
## -- Imputation 3 --
##
##   1  2  3
##
## -- Imputation 4 --
##
##   1  2  3
##
## -- Imputation 5 --
##
##   1  2  3
```

Step 2: Handling Missing Values

Calculate the mean of the imputed values for each missing entry and replace the missing values in the original dataset with these mean values.

```
# Rows with missing values in the 'bwt' column
na_rows <- which(is.na(data1$bwt))

# Extract imputed values for the 'bwt' column from each imputed dataset
imputed_bwt_values <- sapply(imps$imputations, function(x) x$bwt[na_rows])

# Calculate the mean of the imputed values for each missing entry
mean_imputed_bwt <- rowMeans(imputed_bwt_values)

# Replace missing values in the original data with the mean imputed values
data6 = data1
data6$bwt[na_rows] <- mean_imputed_bwt
#imputation complete
```

Plotting Results from Different Methods

To visualize and compare the effectiveness of different imputation methods, we plot the density of the `bwt` variable from the dataset with missing values imputed using various techniques. The density plots will help us understand how each method affects the distribution of the data.

Plotting Results from Different Methods

To visualize and compare the effectiveness of different imputation methods, we plot the density of the `bwt` variable from the dataset with missing values imputed using various techniques. The density plots will help us understand how each method affects the distribution of the data.

```
# Plot density for different imputation methods

# Compute density values for each dataset
density_original <- density(data1$bwt, na.rm = TRUE)
density_mean <- density(data2$bwt)
density_regression <- density(data3$bwt)
density_knn <- density(data4$bwt)
density_mice <- density(data5$bwt)
density_amelia <- density(data6$bwt)

# Plot density for the original data with missing values
plot(density_original,
     main = "Density Plot of Birthweight",
     xlab = "Birthweight",
     ylab = "Density",
     col = "black",
     lwd = 1.5,
     xlim = c(min(density_mean$x, na.rm = TRUE), max(density_mean$x, na.rm = TRUE)),
     ylim = c(0, max(c(density_mean$y, density_regression$y, density_knn$y, density_mice$y, d
ensity_amelia$y)))
)

# Add density lines for each imputation method
lines(density_mean,
      col = "red",
      lty = 2,
      lwd = 1.5
)

lines(density_regression,
      col = "blue",
      lty = 3,
      lwd = 1.5
)

lines(density_knn,
      col = "cyan",
      lty = 4,
      lwd = 1.5
)

lines(density_mice,
      col = "green",
      lty = 5,
      lwd = 1.5
)

lines(density_amelia,
      col = "magenta",
      lty = 6,
      lwd = 1.5
)

# Add a legend
legend("topleft",
```

```
       legend = c("Without NA", "Mean Replacement", "Regression", "kNN", "MICE", "Amelia"),
       col = c("black", "red", "blue", "cyan", "green", "magenta"),
       lty = c(1, 2, 3, 4, 5, 6),
       lwd = rep(1.5, 6),
       bty = "n",
       cex = 0.6,
       pt.cex = 0.6)
```