Exploratory Data Analysis (EDA) Using the Missingno Python Package

A Tool for Visualizing Missing Data

Instructor: Prof. Gopinath Panda

Group 16:

Name	Student ID
Dev Vyas	202201453
Dharmi Patel	202201467
Preksha Shah	202203004

1 Introduction

The `missingno` package in Python is a powerful tool for visualizing and understanding missing data within datasets. This document provides an overview of its functionalities and explains the different types of missing values with detailed practical examples.

2 Functionalities of Missingno

2.1 Matrix

Purpose: The `matrix` function provides a comprehensive visualization of missing data within a DataFrame. It displays a matrix where each cell represents a data point, with colors indicating the presence or absence of data.This function helps in identifying patterns of missing data by showing which rows and columns have missing values. It allows you to quickly see the proportion of missing data and how it is distributed across the dataset.

Syntax:

```
import missingno as msno

msno.matrix(df, **kwargs)
```

Arguments:

- `df`: The DataFrame containing missing values.
- `figsize`: Optional parameter to set the size of the figure (e.g., `(10, 6)`).
- `fontsize`: Optional parameter to set the font size for labels.
- `color`: Optional parameter to set the color of the missing data cells.

Example:

```
import pandas as pd
import missingno as msno

# Sample DataFrame with missing values
data = {
    'A': [1, 2, None, 4],
    'B': [None, 2, 3, 4],
    'C': [1, None, None, 4]
}
df = pd.DataFrame(data)

# Visualize missing values
msno.matrix(df)
```

This code snippet generates a matrix visualization of missing values in the DataFrame.

2.2 Bar

Purpose: The `bar` function creates a bar chart showing the count of missing values in each column. This visualization helps in understanding which columns have the most missing data.By providing a bar chart of missing values, this function highlights columns with substantial missing data, making it easier to prioritize data cleaning efforts.

Syntax:

```
msno.bar(df, **kwargs)
```

Arguments:

- `df`: The DataFrame with missing values.
- `figsize`: Optional parameter to set the size of the figure (e.g., `(10, 6)`).
- `color`: Optional parameter to set the color of the bars.
- `sort`: Optional parameter to sort bars by the number of missing values.

Example:

```
# Visualize missing values as a bar chart
msno.bar(df)
```

This code snippet generates a bar chart to visualize missing values in each column.

2.3 Heatmap

Purpose: The `heatmap` function displays a heatmap showing correlations between missing values in different columns. It helps identify relationships and patterns in the missingness.This function provides a visual correlation matrix of missing values, which can reveal if certain columns have similar patterns of missingness. It's useful for detecting if missing values in one column are related to missing values in another.

Syntax:

```
msno.heatmap(df, **kwargs)
```

Arguments:

- `df`: The DataFrame with missing values.
- `figsize`: Optional parameter to set the size of the figure (e.g., `(10, 8)`).
- `cmap`: Optional parameter to set the colormap for the heatmap.

Example:

```
# Visualize correlation of missing values
msno.heatmap(df)
```

This code snippet generates a heatmap to show correlations between missing values in different columns.

2.4 Dendrogram

Purpose: The `dendrogram` function clusters columns based on their missing value patterns. It helps in understanding how columns with missing values relate to each other.This function uses hierarchical clustering to create a dendrogram that groups columns with similar missing data patterns. This visualization is useful for identifying clusters of columns that share similar characteristics in terms of missing data.

Syntax:

```
msno.dendrogram(df, **kwargs)
```

Arguments:

- `df`: The DataFrame with missing values.
- `figsize`: Optional parameter to set the size of the figure (e.g., `(10, 8)`).

Example:

```
# Visualize columns clustering based on missing values
msno.dendrogram(df)
```

This code snippet generates a dendrogram to show the clustering of columns based on missing values.

3 Types of Missing Values

3.1 MCAR (Missing Completely At Random)

Definition: Missing data that is independent of both observed and unobserved data. The missingness occurs randomly without any pattern.

Example: Suppose you have a dataset of survey responses where some entries are missing due to random errors during data collection, such as a survey participant accidentally skipping questions. The missingness is unrelated to the responses themselves.

Handling MCAR: You can use techniques like listwise deletion or simple imputation (mean or median) as the missingness is random and does not introduce bias.

3.2 MAR (Missing At Random)

Definition: Missing data that depends on observed data but not on the missing data itself. The probability of missingness is related to other measured variables.

Example: Consider a dataset where income data is missing more frequently for people with lower levels of education. Although the probability of missing data is related to education level (an observed variable), it is not related to the missing income values themselves.

Handling MAR: Techniques such as multiple imputation or modeling the missing data can be used as the missingness depends on observed data.

3.3 NMAR (Not Missing At Random)

Definition: Missing data that is related to the missing values themselves. The probability of missingness is dependent on the unobserved data.

Example: In a medical study, patients with severe symptoms may be less likely to report certain side effects, leading to missing data on these symptoms. The missingness is related to the severity of the symptoms, which are unobserved.

Handling NMAR: Handling NMAR data can be challenging. Techniques include using specific models that account for the missingness mechanism or using domain knowledge to infer missing values.