

Exploratory Data Analysis on

CHILD MALNUTRITION

by

Group 21



Pritish N Desai
ID: 202201312
Course: Btech(ICT)



Khelan Bhatt
ID: 202411025
Course: Mtech-ICT
(ML)



Jaydeep Darji
ID: 202411029
Course: Mtech-ICT
(ML)

Course Code: IT 462
Semester: Autumn 2024

Under the guidance of

Dr. Gopinath Panda



Dhirubhai Ambani Institute of Information and Communication Technology

December 2, 2024

ACKNOWLEDGMENT

We are writing this letter to express our heartfelt gratitude for your guidance and support throughout the duration of our project titled “Child Malnutrition”. Your invaluable assistance has played a pivotal role in shaping the successful completion of this endeavor.

We are extremely fortunate to have had the opportunity to work under your mentorship. Your expertise, encouragement, and willingness to share your knowledge have been instrumental in elevating the quality and scope of our project. Your constructive feedback and insightful suggestions have helped us overcome challenges and develop a deeper understanding of the subject matter.

Furthermore, we would like to extend our appreciation to the entire team at DAICT for fostering an environment of collaboration and innovation. The resources and facilities provided have been crucial in conducting comprehensive research and analysis.

We would also like to express our gratitude to our peers and colleagues who have been supportive throughout this journey. Their valuable input and camaraderie have been a constant source of motivation.

Completing this project has been a tremendous learning experience, and we are confident that the knowledge and skills acquired during this endeavor will serve as a solid foundation for our future endeavors.

Once again, thank you for your unwavering guidance and belief in our abilities. Your mentorship has been invaluable, and we are truly grateful for the opportunity to work with you.

Sincerely,

Pritish N Desai, 202201312

Khelan Bhatt, 202411025

Jaydeep Darji, 202411029

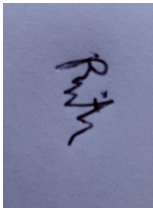
DECLARATION

We, all the members of group 21, hereby declare that the EDA project work presented in this report is our original work and has not been submitted for any other academic degree. All the sources cited in this report have been appropriately referenced.

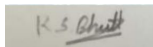
We acknowledge that the data used in this project is obtained from the www.who.int site. We also declare that we have adhered to the terms and conditions mentioned in the website for using the dataset. We confirm that the dataset used in this project is true and accurate to the best of our knowledge.

We acknowledge that we have received no external help or assistance in conducting this project, except for the guidance provided by our mentor Prof. Gopinath Panda. We declare that there is no conflict of interest in conducting this EDA project.

We hereby sign the declaration statement and confirm the submission of this report on 2nd December, 2024.



Pritish N Desai
ID: 202201312
Course: Btech(ICT)



Khelan Bhatt
ID: 202411025
Course: Mtech-ICT
(ML)



Jaydeep Darji
ID: 202411029
Course: Mtech-ICT
(ML)

CERTIFICATE

This is to certify that Group 21 comprising Pritish N Desai, Khelan Bhatt, and Jaydeep Darji has successfully completed an exploratory data analysis (EDA) project on the Child Malnutrition, which was obtained from www.who.int.

The EDA project presented by Group 21 is their original work and has been completed under the guidance of the course instructor, Prof. Gopinath Panda, who has provided support and guidance throughout the project. The project is based on a thorough analysis of the Child Malnutrition dataset, and the results presented in the report are based on the data obtained from the dataset.

This certificate is issued to recognize the successful completion of the EDA project on the Child Malnutrition, which demonstrates the analytical skills and knowledge of the students of Group 21 in the field of data analysis.

Signed,
Dr. Gopinath Panda,
IT 462 Course Instructor
Dhirubhai Ambani Institute of Information and Communication Technology
Gandhinagar, Gujarat, INDIA.

December 2, 2024

Contents

List of Figures	5
1 Introduction	1
1.1 Project idea	1
1.2 Data Collection	1
1.3 Dataset Description	1
1.4 Packages required	4
2 Data Cleaning	7
2.1 Missing data analysis	7
2.2 Imputation	10
3 Visualization	13
3.1 Histograms of numerical columns	13
3.2 Visualizing count distributions for categorical columns	14
4 Feature Engineering	17
4.1 Creating a new feature 'Malnutrition Severity' by combining multiple malnutrition indicators	17
4.2 Handling Outliers	17
4.3 Distribution after finding malnutrition severity	19
5 Model fitting	24
5.1 Regression	24
5.1.1 Linear Regression	24
5.1.2 Support Vector Regression	25
5.1.3 Random Forest	26
5.1.4 XGBoost Regression	27
6 Conclusion & future scope	29
6.1 Findings/observations	29
6.2 Challenges	30
6.3 Future plans	31
7 Group Contribution	33
8 Short Bio	35

9	Deriving the Equation for Malnutrition Severity	37
9.1	Selection of Features	37
9.2	Preparing the Data	37
9.3	Training the Linear Regression Model	38
9.4	Fitting the Model and Extracting the Coefficients	38
9.5	Formulating the Equation	38
9.6	Coefficients and Equation	38
9.7	Interpreting the Equation	39
10	References	40

Abstract

This exploratory data analysis(EDA) project delves into the statistics and analysis about child malnutrition in all the countries around the world, aiming to uncover insights into the nutrition landscape of children around the globe.It comprises data spanning all the years from 1987-2023, providing a comprehensive view of nutrition trends over time.The dataset shows prevalence estimates for the five main indicators, namely underweight, stunting, overweight, wasting, severe wasting.The analysis begins by examining the overall nutrition rates across different countries, highlighting variations and disparities. Through visualizations such as line plots and statistical summaries, the project explores how quality of nutrition has evolved over the years, identifying key factors influencing these trends. Employing robust quantitative techniques, including regression analysis, the research investigates the multifaceted factors influencing variations in nutrition among different countries.

Chapter 1. Introduction

1.1 Project idea

Child malnutrition points to the excesses, imbalances, and deficiencies in a child's intake of nutrients. Our project aims to analyse the prevalence estimates of overweight, underweight, stunting, wasting and severe wasting and study the evolution and changes in these 5 indicators over the years. Our goal is to explore and study the values of these estimators in different countries and find out which country suffers the most and which country suffers the least by finding the outliers. Our aim is also to find out the correlation among these indicators and also how other factors (like Mother's Education, Wealth of country) affect the malnutrition severity.

1.2 Data Collection

Acquiring the dataset for our project proved to be a crucial and labour-intensive work. We navigated through the data.gov.in website, finding our way through a vast repository of over 28 datasets tagged with 'Malnutrition'. Each dataset demanded meticulous observation to ensure it aligned with the objectives of our analysis. Even after investing a considerable amount of time and effort, we were not able to find the ideal dataset for our study, as all the datasets contained the data for only one year, either 2005-06 or 2015-16, and those datasets contained only 38 rows. So we thought of visiting the www.who.int website, which is the official website of World Health Organization. There we found the relevant and ideal dataset pertaining exactly to our topic. This dataset- Global Database on Child Growth and Malnutrition spanning from 1987-2023, emerged as the perfect fit, laying the foundation for our exploration of nutrition trends across Countries around the world.

Using pandas library we read our CSV files-

```
[ ] # Load the dataset  
  
df = pd.read_excel('/content/EDA.xlsx')
```

1.3 Dataset Description

The dataset contains the prevalence estimates for the five indicators of malnutrition- (underweight, stunting, overweight, wasting, severe wasting) of over 150+ countries around the globe spanning

over three decades from 1987-2023. The values of the prevalence estimates of the five indicators are given in float datatype. It also contains information of other features like Country information (ISO-3 Code, name, region), survey metadata (year period, median year, start and end months, source details), malnutrition indicators like severe wasting, wasting, overweight, stunting, and underweight, survey methodology and representation level, Mother's Education, and wealth quintile.

Some key observations from the dataset:

1. Missing Data: We found that the dataset had some columns with more than 90% missing values like 'Other group/Zone', 'Representation level', 'Mother education'. This could be due to unavailability of data.
2. Varying ranges: We found out that the dataset columns containing values of the 5 indicators had significant disparities in values-
 - (a) Severe Wasting: min= 0.00, max= 42.64350
 - (b) Wasting: min=0.00, max= 60.535370
 - (c) Overweight: min=0.00, max= 72.000
 - (d) Stunting: min=0.00, max= 95.9000
 - (e) Underweight: min=0.00, max= 78.7000

Features of the Dataset

```
[ ] df.columns
```

```
⇒ Index(['Country ISO-3 Code', 'Country Short Name', 'Year period',
        'Median Year', 'Start Month', 'End Month', 'WHO Reference number',
        'Representation level', 'Region', 'Other group/Zone', 'Age', 'Sex',
        'Mother education', 'Wealth quintile', 'Urban/Rural', 'Sample size',
        'Severe wasting', 'Wasting', 'Overweight', 'Stunting', 'Underweight',
        'Short Source Code', 'Reference Title', 'Author', 'Notes', 'JME (Y/N)'],
        dtype='object')
```

The above code checks what columns exist in our dataset.

Shape of the Dataset

```
[ ] df.shape
```

```
⇒ (41629, 26)
```

The above output shows the number of rows and number of columns of our dataset.

Datatypes of each Feature

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 41629 entries, 0 to 41628
Data columns (total 26 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   Country ISO-3 Code                    41629 non-null  object
1   Country Short Name                    41629 non-null  object
2   Year period                           41629 non-null  object
3   Median Year                           41629 non-null  int64
4   Start Month                           39560 non-null  object
5   End Month                             39495 non-null  object
6   WHO Reference number                  41629 non-null  int64
7   Representation level                  1618 non-null   object
8   Region                                11051 non-null  object
9   Other group/Zone                      31 non-null     object
10  Age                                    41629 non-null  object
11  Sex                                    41629 non-null  object
12  Mother education                      1386 non-null   object
13  Wealth quintile                       3334 non-null   object
14  Urban/Rural                           41629 non-null  object
15  Sample size                           36588 non-null  float64
16  Severe wasting                        39564 non-null  float64
17  Wasting                               40161 non-null  float64
18  Overweight                            39440 non-null  float64
19  Stunting                              40237 non-null  float64
20  Underweight                           40579 non-null  float64
21  Short Source Code                     41629 non-null  object
22  Reference Title                       41629 non-null  object
23  Author                                41629 non-null  object
24  Notes                                 15742 non-null  object
25  JME (Y/N)                             41629 non-null  object
dtypes: float64(6), int64(2), object(18)
memory usage: 8.3+ MB
```

Statistical properties of the Dataset

```
[ ] # Summary statistics for numerical columns
```

```
df.describe()
```

	Median Year	WHO Reference number	Sample size	Severe wasting	Wasting	Overweight	Stunting	Underweight
count	41629.000000	41629.000000	36588.000000	39564.000000	40161.000000	39440.000000	40237.000000	40579.000000
mean	2007.864566	3422.558097	1720.393080	2.115046	6.948246	5.891519	27.553171	15.583042
std	9.284120	1879.976794	4948.012744	2.385827	5.764848	5.048195	16.533984	12.571275
min	1966.000000	5.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	2001.000000	2768.000000	334.000000	0.545408	2.454410	2.131672	14.033330	5.163105
50%	2009.000000	3201.000000	725.000000	1.398590	5.624420	4.601200	26.017800	13.135100
75%	2015.000000	3453.000000	1559.000000	2.843222	9.909120	8.256985	38.650880	22.500935
max	2023.000000	10693.000000	237205.000000	42.643560	60.535370	72.000000	95.900000	78.700000

Here, overall, we get a statistical overview of our Child malnutrition dataset.

1.4 Packages required

```
[ ] # Importing necessary libraries

import pandas as pd
import numpy as np
import missingno as msno
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
```

Here is the list of Python libraries that we used for the project.

1. Pandas (import pandas as pd):
 - (a) Pandas is a powerful data manipulation and analysis library in Python
 - (b) It provides data structures like DataFrame and Series, facilitating easy handling and manipulation of structured data.
2. NumPy (import numpy as np):
 - (a) NumPy is a fundamental package for numerical computing in Python.
 - (b) It offers support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate efficiently on these arrays
3. Missingno (import missingno as msno):
 - (a) Missingno is a Python library that provides a set of tools for visualizing and dealing with missing data in datasets.
 - (b) It offers visualizations such as bar charts and heatmaps to visualize missing data patterns.
4. Matplotlib (import matplotlib.pyplot as plt):
 - (a) Matplotlib is a comprehensive plotting library for Python.

- (b) It enables the creation of a wide variety of plots, including line plots, scatter plots, bar plots, histograms, and more.
5. Seaborn (import seaborn as sns):
 - (a) Seaborn is a statistical data visualization library built on top of Matplotlib.
 - (b) It provides a high-level interface for creating attractive and informative statistical graphics, simplifying the process of creating complex visualizations
 6. train test split (from sklearn.model selection import train test split):
 - (a) train test split is a function from scikit-learn used for splitting a dataset into training and testing sets.
 - (b) It randomly splits the data into training and testing sets according to a specified ratio, enabling evaluation of machine learning models on unseen data.
 7. LabelEncoder (from sklearn.preprocessing import LabelEncoder):
 - (a) LabelEncoder is a utility class from scikit-learn used for encoding categorical features as integer labels.
 - (b) It transforms categorical data into numerical form, which is required by many machine learning algorithms.
 8. StandardScaler(from sklearn.preprocessing import StandardScaler):
 - (a) The StandardScaler standardizes features by removing the mean and scaling to unit variance (Z-score normalization).
 - (b) It is commonly used when working with machine learning algorithms sensitive to feature scales, such as Support Vector Machines (SVM), Logistic Regression, or Principal Component Analysis (PCA).
 9. OneHotEncoder(from sklearn.preprocessing import OneHotEncoder):
 - (a) The OneHotEncoder is used to convert categorical data into a one-hot numeric array.
 - (b) This is useful when working with algorithms that require numeric input or don't work with categorical variables directly, like many in scikit-learn.
 10. LinearRegression (from sklearn.linear model import LinearRegression):
 - (a) LinearRegression is a linear regression model implementation from scikitlearn.
 - (b) It fits a linear model to the training data and makes predictions based on the relationship between independent and dependent variables.
 11. RandomForestRegressor(from sklearn.ensemble import RandomForestRegressor)
 - (a) The RandomForestRegressor from the sklearn.ensemble module is a machine learning model used for regression tasks

- (b) It builds an ensemble of decision trees, where each tree is trained on a random subset of the data (using bootstrap sampling), and outputs the mean prediction from all the trees. This approach improves accuracy and reduces overfitting.
12. mean squared error (from `sklearn.metrics` import `mean_squared_error`):
- (a) mean squared error is a function from scikit-learn used for evaluating the performance of regression models.
 - (b) It calculates the mean squared error between the predicted and true values, providing a measure of the model's accuracy.
13. `r2` score (from `sklearn.metrics` import `r2` score):
- (a) `r2` score is a function from scikit-learn used for evaluating the performance of regression models.
 - (b) It calculates the coefficient of determination (R-squared), which measures the proportion of the variance in the dependent variable that is predictable from the independent variables.
14. `SimpleImputer` (from `sklearn.impute` import `SimpleImputer`):
- (a) `SimpleImputer` is a class from scikit-learn used for imputing missing values in a dataset.
 - (b) It enables the replacement of missing values with a specified strategy, such as mean, median, or most frequent value.
15. `ColumnTransformer`(from `sklearn.compose` import `ColumnTransformer`):
- (a) It is a powerful tool that allows us to apply different preprocessing steps to specific columns of a dataset.
 - (b) This is particularly useful when you have a mix of numerical and categorical data that requires different transformations.
16. `Pipeline`(from `sklearn.pipeline` import `Pipeline`):
- (a) The `Pipeline` class from `sklearn.pipeline` is a useful tool for chaining multiple preprocessing steps and modeling steps together in a single object.
 - (b) This allows us to streamline and automate the process of transforming data and training a model, ensuring that all steps are applied in the correct order.

Chapter 2. Data Cleaning

Data Cleaning is the process of identifying and rectifying errors, inconsistencies, or inaccuracies in raw data to ensure it is accurate, complete, and usable for analysis or machine learning tasks. It is a crucial step in the data preparation phase and directly affects the quality of insights or models generated.

2.1 Missing data analysis

Here there were few reason and observations for the missing data-

1. Political or Geographical Factors: Conflicts, natural disasters, or other political or geographical factors could have hindered data collection in some regions, resulting in missing values.
2. Data Collection Issues: There might have been challenges or lapses in data collection efforts in certain countries during specific years, leading to missing data.
3. Country: There might be some countries where no malnourished children survey might have been conducted in a particular year. For example, in the dataset all the values for the five indicators are blank for Angola for the year 1996.

Missing value counts for each Feature

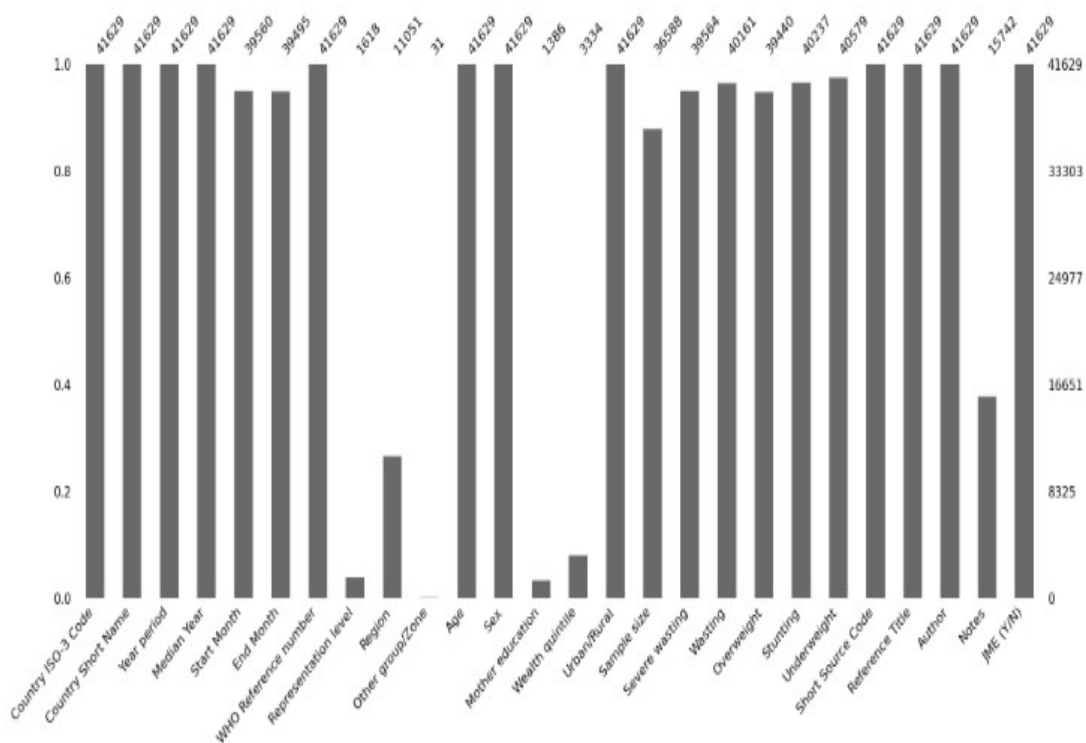
```
df.isnull().sum()
```

	0
Country ISO-3 Code	0
Country Short Name	0
Year period	0
Median Year	0
Start Month	2069
End Month	2134
WHO Reference number	0
Representation level	40011
Region	30578
Other group/Zone	41598
Age	0
Sex	0
Mother education	40243
Wealth quintile	38295
Urban/Rural	0
Sample size	5041

Urban/Rural	0
Sample size	5041
Severe wasting	2065
Wasting	1468
Overweight	2189
Stunting	1392
Underweight	1050
Short Source Code	0
Reference Title	0
Author	0
Notes	25887
JME (Y/N)	0

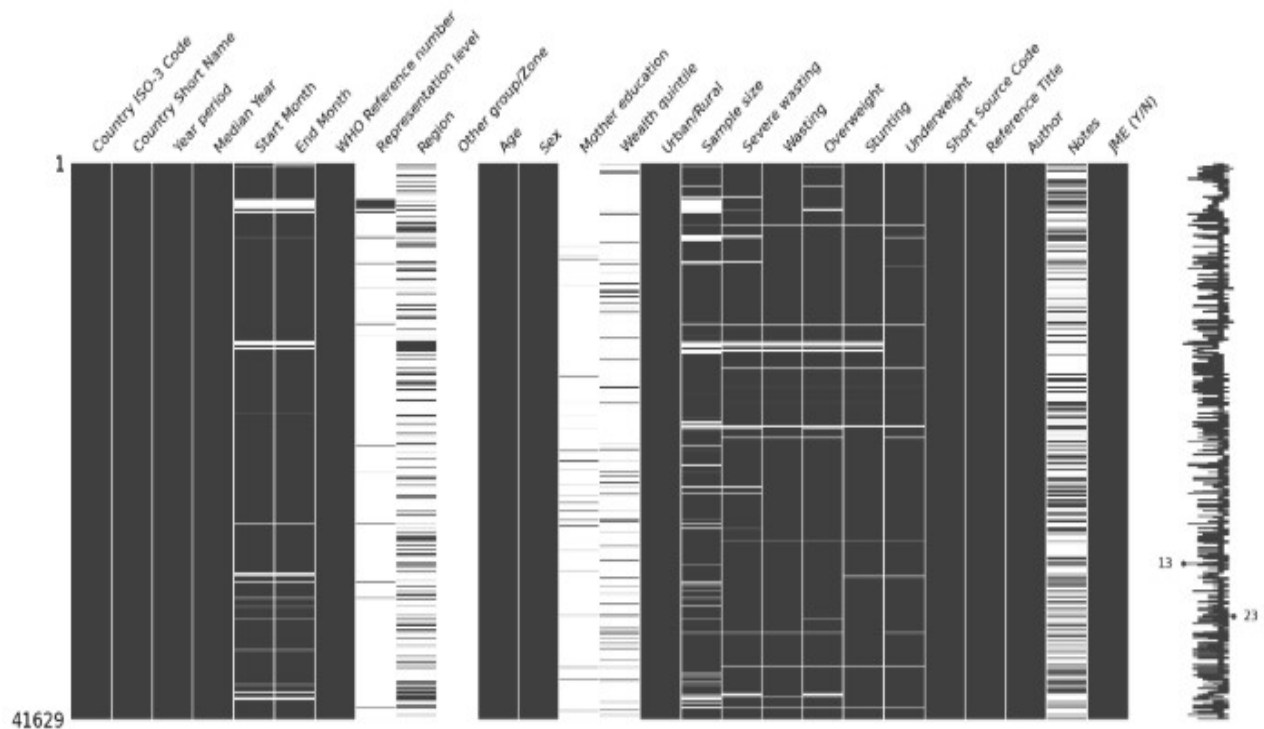
dtype: int64

Feature-wise bars for Missing value representation



- This bar plot provides a visual representation of the missing values for each column in the dataset.
- A higher bar indicates a lower number of missing values, while shorter bars reflect columns with a significant proportion of missing data.
- From the plot, it is evident that the columns Representation level, Other Group/Zone, Mother education, and Wealth quintile have the highest number of missing values.
- This insight highlights the need for further investigation and appropriate handling of missing data in these columns, such as using imputation techniques or considering the potential impact of missingness on analysis and modeling.

Missing Value representation in matrix form



From the above plots and the output of total number of missing values per feature it is crystal clear that missing values exist in our dataset

- The missing number matrix visually highlights the distribution of missing values across the dataset.
- On the right side, the width of the lines provides a quick indication of missingness—wider lines signify columns with a greater proportion of missing data.
- This visualization effectively pinpoints areas in the dataset that require attention, helping to prioritize data cleaning and imputation efforts.

2.2 Imputation

Many methods to impute the dataset :

- Imputation with mean, mode, median
 1. Mean imputation
 2. Mode imputation
 3. Median imputation
- Prediction
 1. Linear interpolation
 2. K-nearest neighbours imputation
- Forward Fill/Backward Fill
- Deletion

Here, first we are dropping columns with more than 90% missing data, as they are already containing much more missing values, so imputation in those columns may cause inaccuracies

```
[ ] # Drop columns with > 90% missing data

df.drop(columns=['Other group/Zone', 'Representation level', 'Mother education'], inplace=True)
```

Mode Imputation

The code imputes missing values in the categorical columns 'Region' and 'Wealth quintile' by replacing them with the most frequent (mode) value in each column. This method assumes that the most common category is a reasonable substitute for missing values, which works well when missing data is minimal and the distribution of categories is not highly skewed.

```
[ ] # Handle 'Start Month' and 'End Month' as categorical variables

df['Start Month'] = df['Start Month'].fillna(df['Start Month'].mode()[0])
df['End Month'] = df['End Month'].fillna(df['End Month'].mode()[0])

[ ] # Impute categorical columns with the mode

df['Region'] = df['Region'].fillna(df['Region'].mode()[0])
df['Wealth quintile'] = df['Wealth quintile'].fillna(df['Wealth quintile'].mode()[0])

[ ] # Handle 'JME (Y/N)' column
df['JME (Y/N)'] = df['JME (Y/N)'].fillna(df['JME (Y/N)'].mode()[0])
```

Mean Imputation

The code imputes missing values in the specified numerical columns ('Sample size', 'Severe wasting', 'Wasting', 'Overweight', 'Stunting', 'Underweight') by replacing them with the mean value of each respective column. This approach assumes that the missing data is missing at random and that imputing with the mean will not significantly distort the data distribution, especially given that the missing data is relatively low.

```
[ ] # Impute missing values for numerical columns with relatively low missing data using mean
numerical_columns_with_missing = ['Sample size', 'Severe wasting', 'Wasting', 'Overweight', 'Stunting', 'Underweight']

# Impute missing values for numerical columns with the mean
for col in numerical_columns_with_missing:
    df[col] = df[col].fillna(df[col].mean())
```

Count of missing values after imputation

	0
Country ISO-3 Code	0.0
Country Short Name	0.0
Year period	0.0
Median Year	0.0
Start Month	0.0
End Month	0.0
WHO Reference number	0.0
Region	0.0
Age	0.0
Sex	0.0
Wealth quintile	0.0
Urban/Rural	0.0
Sample size	0.0
Severe wasting	0.0
Wasting	0.0
Overweight	0.0
Stunting	0.0
Underweight	0.0
Short Source Code	0.0
Reference Title	0.0
Author	0.0
JME (Y/N)	0.0

dtype: float64

Hence, all the missing values have been removed.

Chapter 3. Visualization

In data visualization, we graphically represent the data, analyze it, and convey information efficiently. It's a crucial step for understanding complex data patterns, identifying trends, and making informed decisions. Used throughout the data analysis process, it enhances understanding and enables data-driven decision-making.

3.1 Histograms of numerical columns

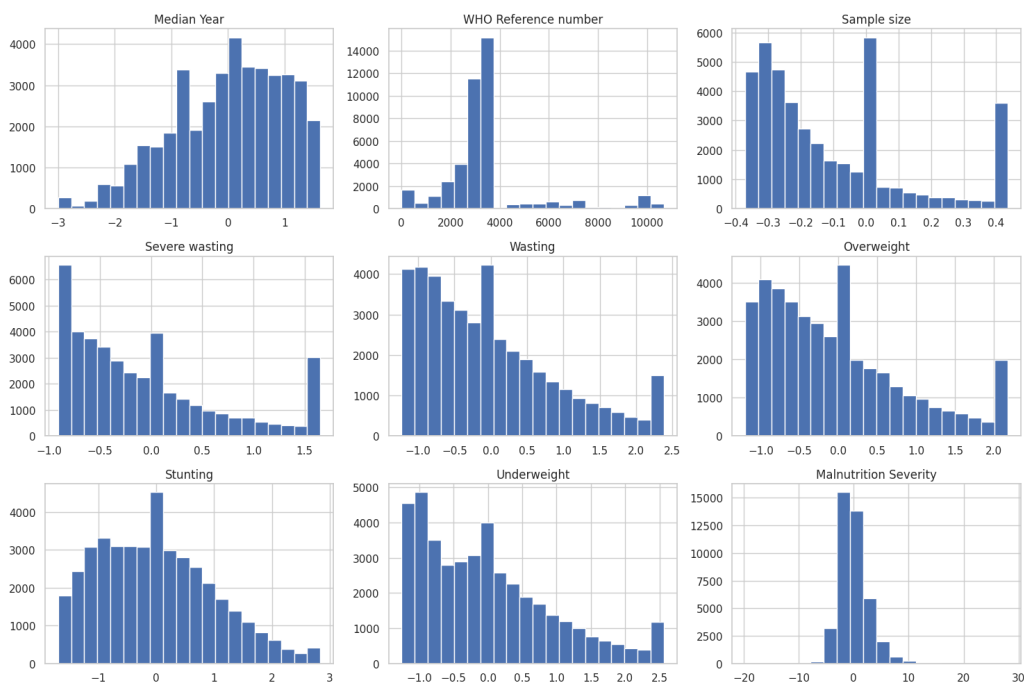


Figure 3.1: Distribution of Start Month

These histograms provide a visual representation of the distribution of each numerical variable. They can help you identify potential outliers, skewness, and other patterns in the data. This information can be useful for data cleaning and preprocessing.

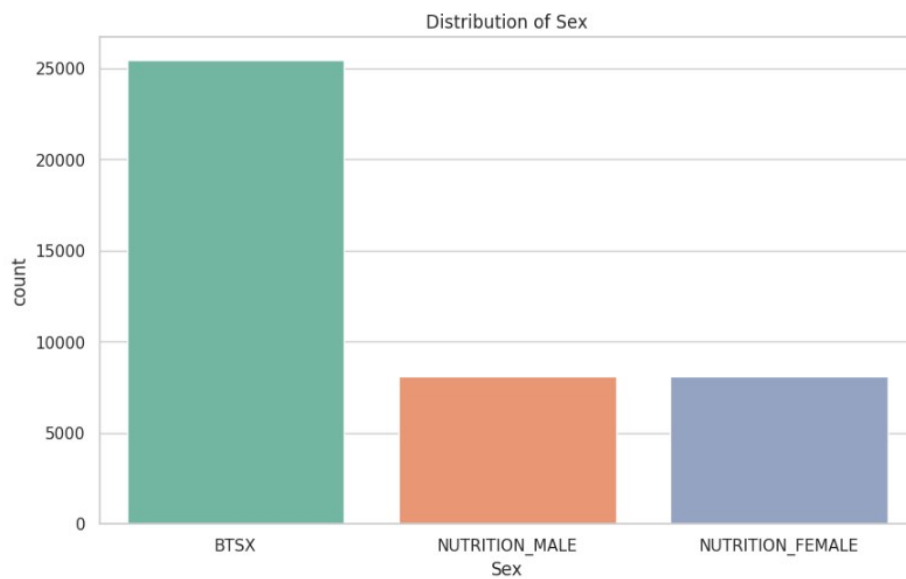


Figure 3.4: Distribution of Gender

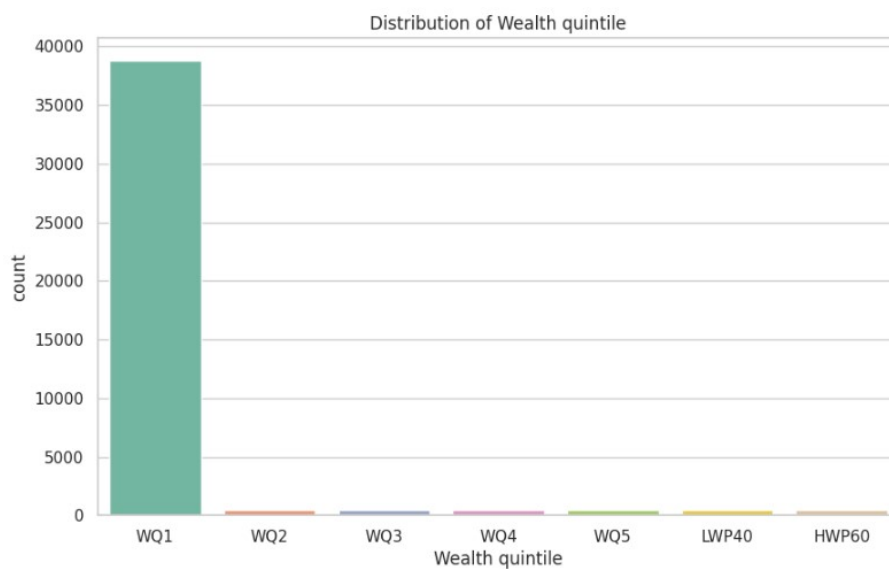


Figure 3.5: Distribution of Wealth Quintile

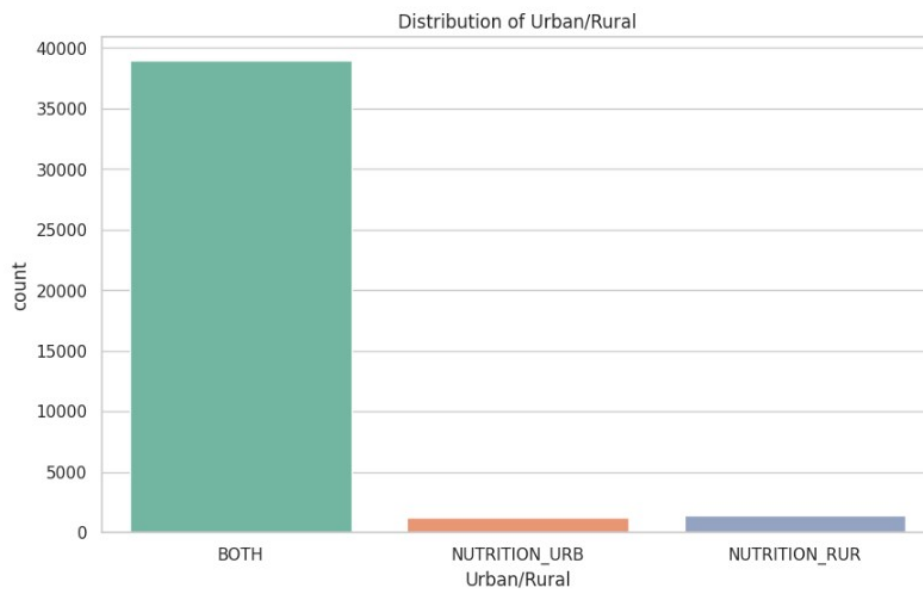


Figure 3.6: Distribution of Urban/Rural areas

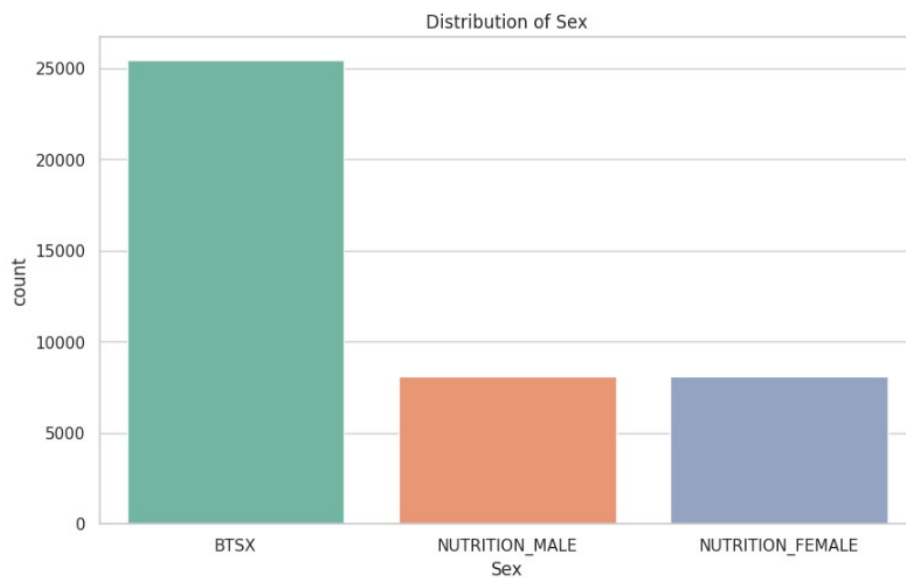


Figure 3.7: Distribution of Gender

Chapter 4. Feature Engineering

Applying Standard Scaling

```
[ ] numerical_columns = ['Median Year', 'Sample size', 'Severe wasting', 'Wasting', 'Overweight', 'Stunting', 'Underweight']

# Apply Standard Scaling
scaler = StandardScaler()
df[numerical_columns] = scaler.fit_transform(df[numerical_columns])

# Check the transformed data
df.head()
```

4.1 Creating a new feature 'Malnutrition Severity' by combining multiple malnutrition indicators



```
df['Malnutrition Severity'] = df['Severe wasting'] + df['Wasting'] + df['Underweight'] - df['Overweight'] - df['Stunting']
```

4.2 Handling Outliers

The code calculates the Interquartile Range (IQR) to detect outliers in the numerical columns of a DataFrame. The IQR is the difference between the 75th percentile (Q3) and the 25th percentile (Q1) of the data. Outliers are defined as values that fall outside the range of $Q1 - 1.5 \times IQR$ to $Q3 + 1.5 \times IQR$. This method helps identify extreme values that may distort analysis and indicates the presence of potential anomalies in the dataset. The count of such outliers is displayed with `outliers.sum()`.


```
[ ] Q1 = df[numerical_columns].quantile(0.25)
    Q3 = df[numerical_columns].quantile(0.75)
    IQR = Q3 - Q1

    # Define outlier criteria
    outliers = ((df[numerical_columns] < (Q1 - 1.5 * IQR)) | (df[numerical_columns] > (Q3 + 1.5 * IQR)))
    print(outliers.sum()) # Check how many outliers exist
```

```
➡ Median Year      200
   Sample size    3390
   Severe wasting  2685
   Wasting        1210
   Overweight     1663
   Stunting       244
   Underweight    884
   dtype: int64
```

Defining a function to handle outliers

```
[ ] # Define a function to handle outliers using IQR
def handle_outliers(df, columns):
    for col in columns:
        Q1 = df[col].quantile(0.25)
        Q3 = df[col].quantile(0.75)
        IQR = Q3 - Q1

        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        df[col] = df[col].apply(lambda x: min(max(x, lower_bound), upper_bound))

    return df

# List of numerical columns to check for outliers
numerical_columns_with_outliers = ['Median Year', 'Sample size', 'Severe wasting', 'Wasting', 'Overweight', 'Stunting', 'Underweight']

# Step 3: Handle outliers in the dataframe
df = handle_outliers(df, numerical_columns_with_outliers)
```

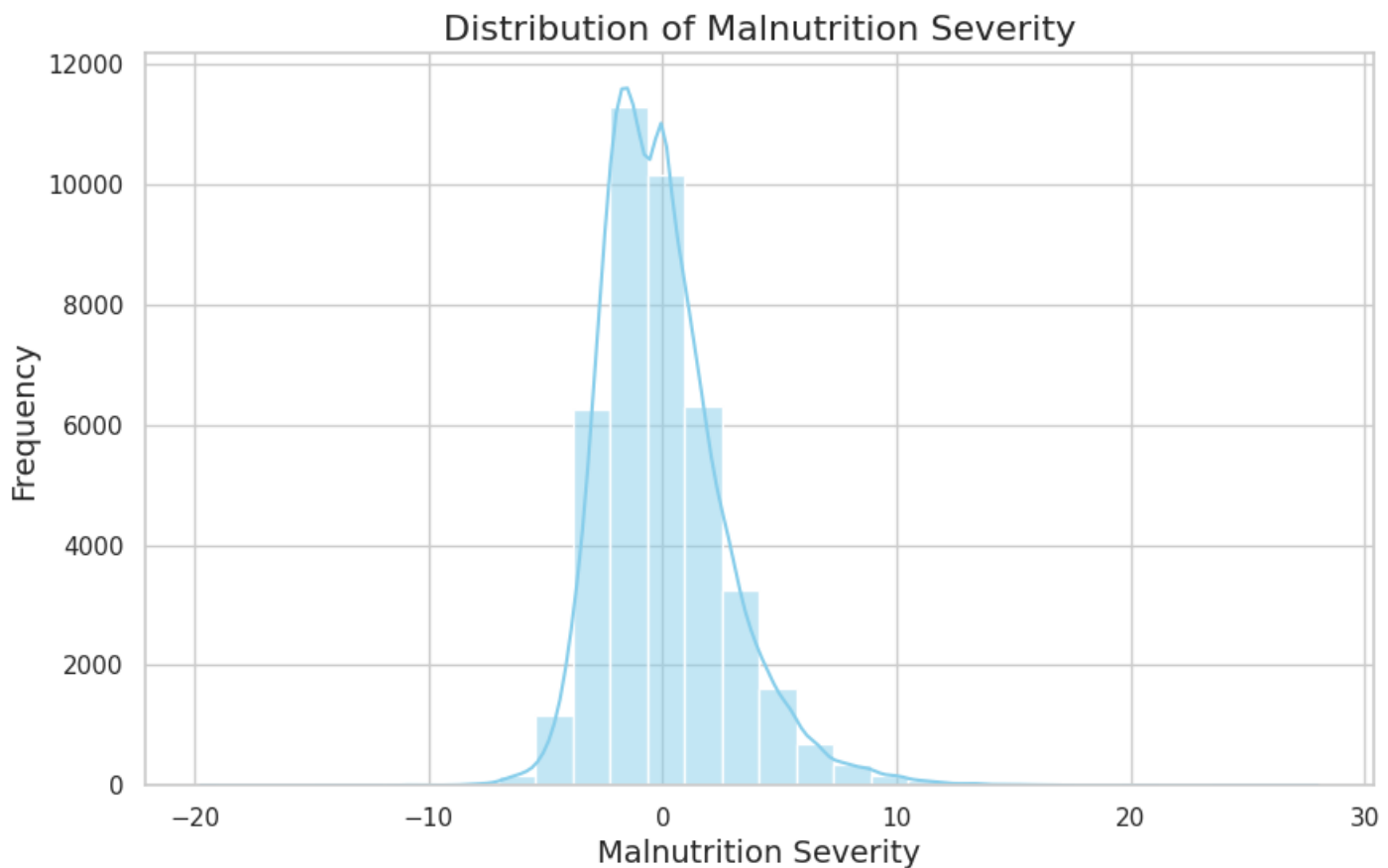
Result:

```
[ ] Q1 = df[numerical_columns].quantile(0.25)
    Q3 = df[numerical_columns].quantile(0.75)
    IQR = Q3 - Q1

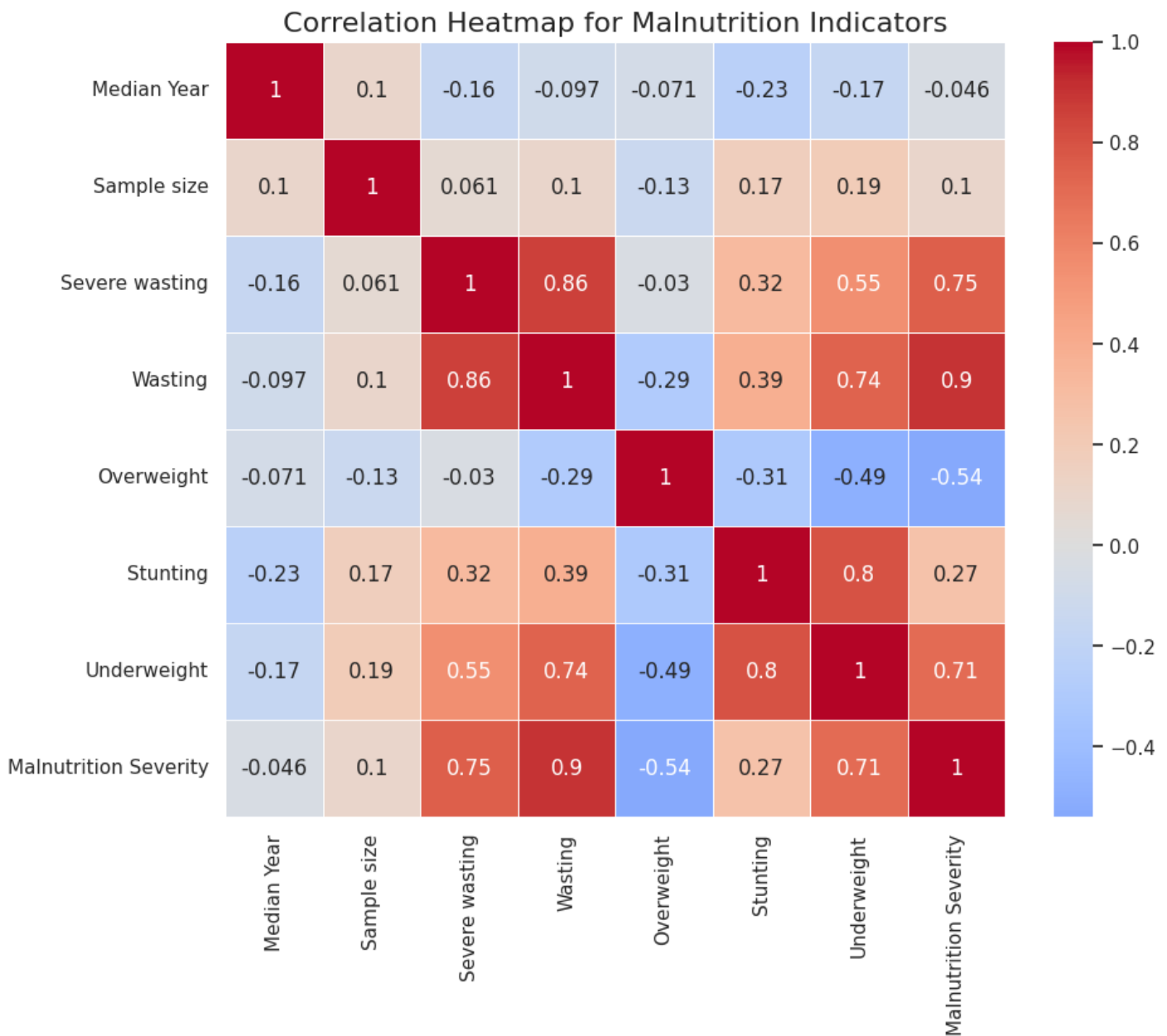
    # Define outlier criteria
    outliers = ((df[numerical_columns] < (Q1 - 1.5 * IQR)) | (df[numerical_columns] > (Q3 + 1.5 * IQR)))
    print(outliers.sum())
```

```
➡ Median Year      0
   Sample size     0
   Severe wasting   0
   Wasting          0
   Overweight       0
   Stunting         0
   Underweight      0
   dtype: int64
```

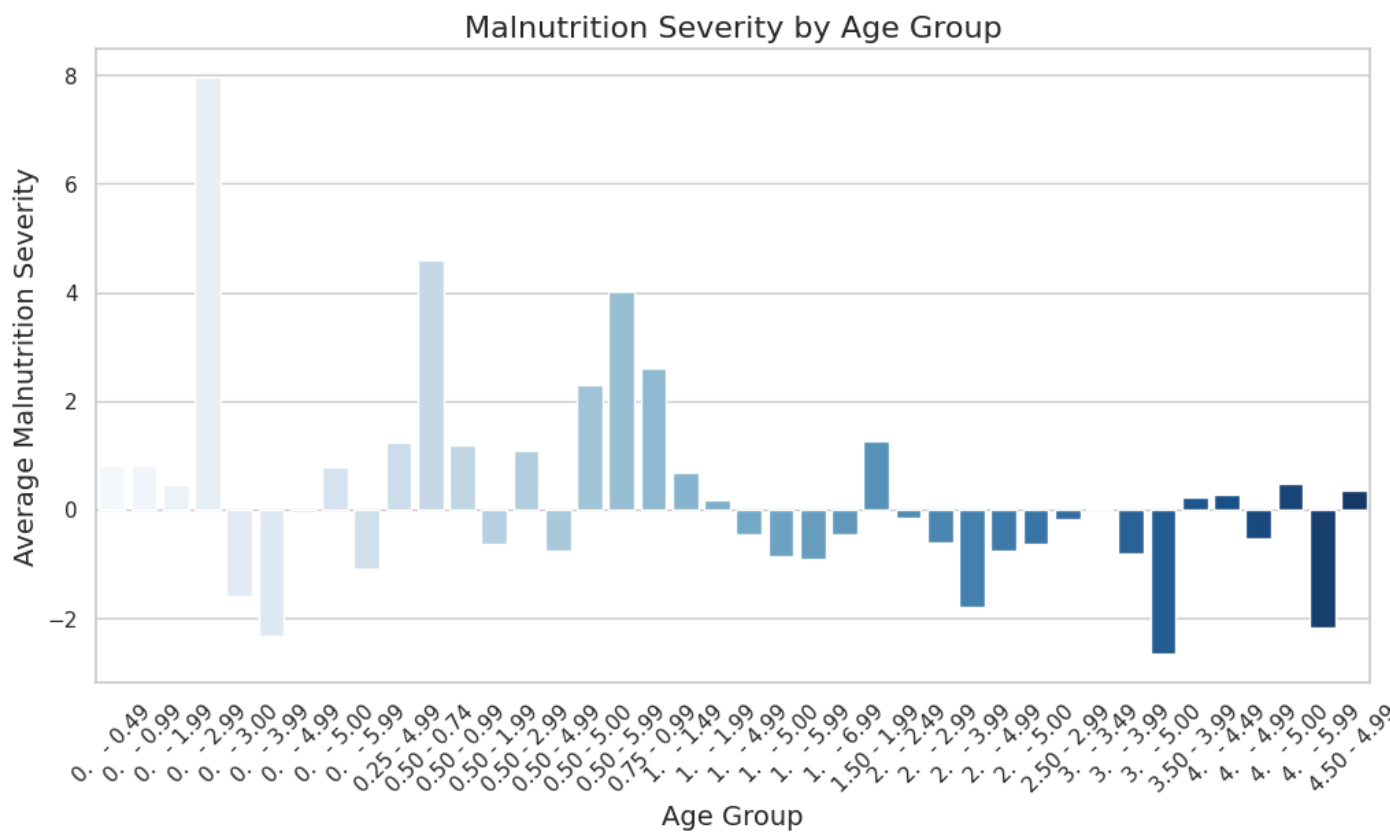
4.3 Distribution after finding malnutrition severity



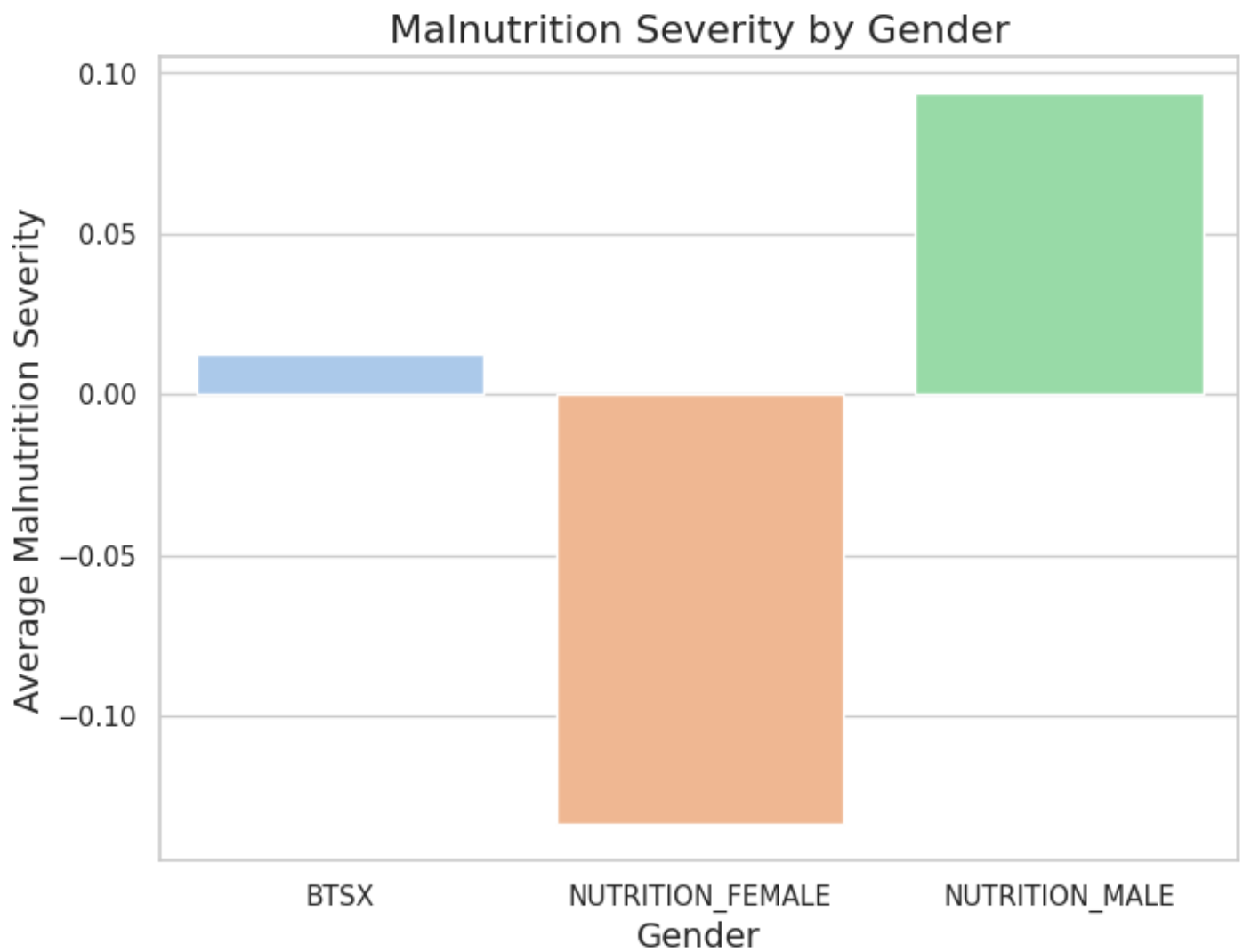
The histogram shows the distribution of the 'Malnutrition Severity' variable. It appears to be somewhat normally distributed, but with a slight skew. This suggests that there is a range of malnutrition severity levels in the dataset, with some individuals experiencing more severe malnutrition than others.



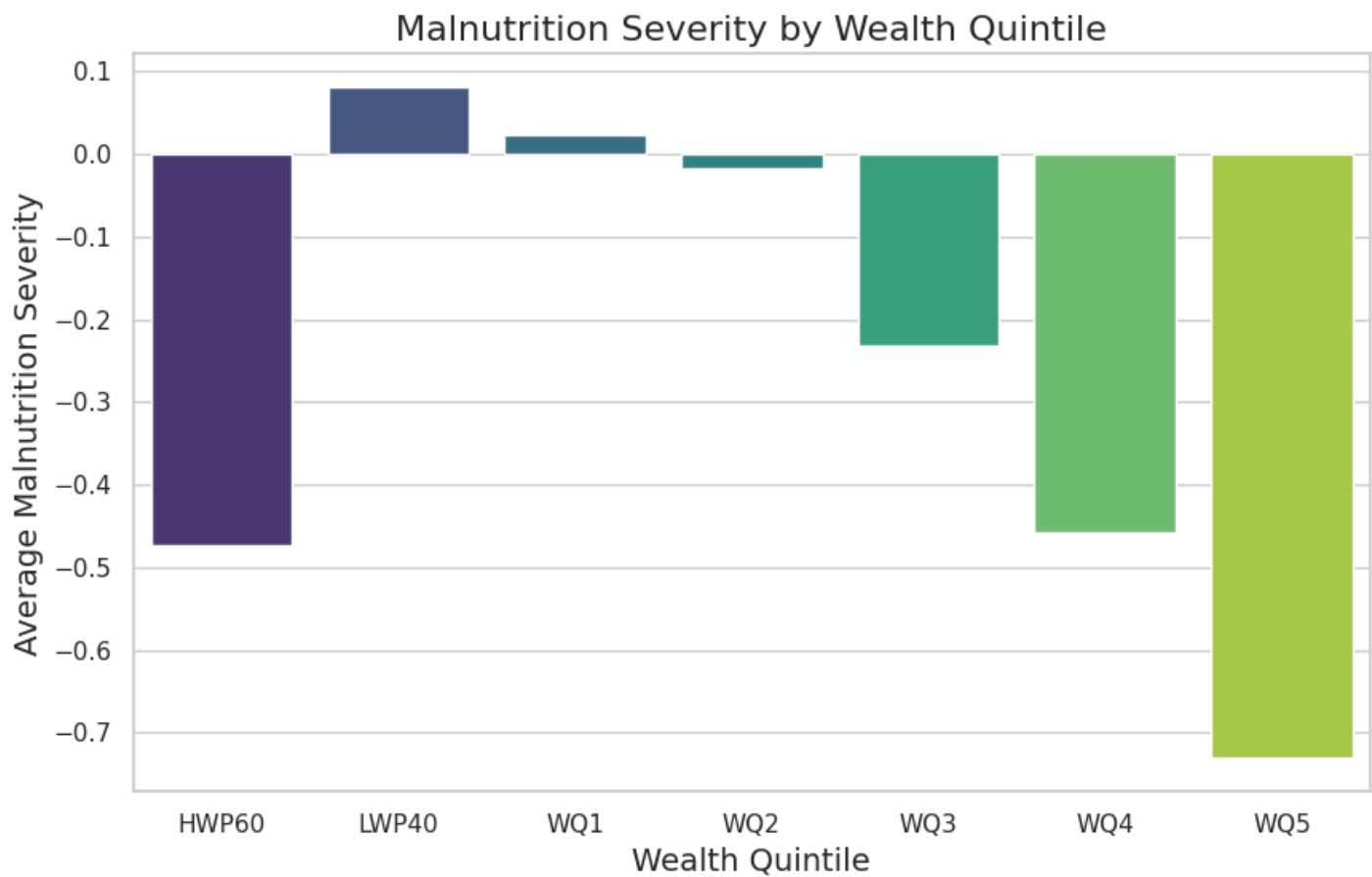
This heatmap displays the correlation between different malnutrition indicators. Darker colors indicate stronger correlations. For example, 'Severe wasting' and 'Wasting' have a strong positive correlation, which is expected as severe wasting is a more severe form of wasting. It also observed that correlations between other indicators, helping you understand the relationships between them.



This plot shows how malnutrition severity varies across different age groups. It appears that certain age groups, such as those in early childhood, might be more vulnerable to malnutrition. This insight can help target interventions and resources to the most vulnerable populations.



This bar plot compares the average malnutrition severity between males and females. If there is a significant difference in the bar heights, it would suggest that one gender might be more susceptible to malnutrition. This information can help in designing gender-specific interventions.



This plot explores the relationship between wealth quintile and malnutrition severity. It's common to see a trend where lower wealth quintiles experience higher levels of malnutrition. This indicates that socioeconomic factors play a significant role in malnutrition prevalence.

Chapter 5. Model fitting

Model fitting is a crucial step in building a robust predictive system. By selecting the right model, tuning its hyperparameters, and evaluating its performance on unseen data, we ensure that the model is both accurate and generalizable.

For this project, predicting malnutrition severity, models like Random Forest Regressor and XG-Boost are expected to perform well given their ability to capture complex relationships in the data. However, Linear Regression and SVR can serve as baseline models to understand how simple or complex the relationships in the data are.

The process involves constant evaluation and refinement, ensuring the chosen model provides the best predictions for future unseen data while meeting project goals effectively.

5.1 Regression

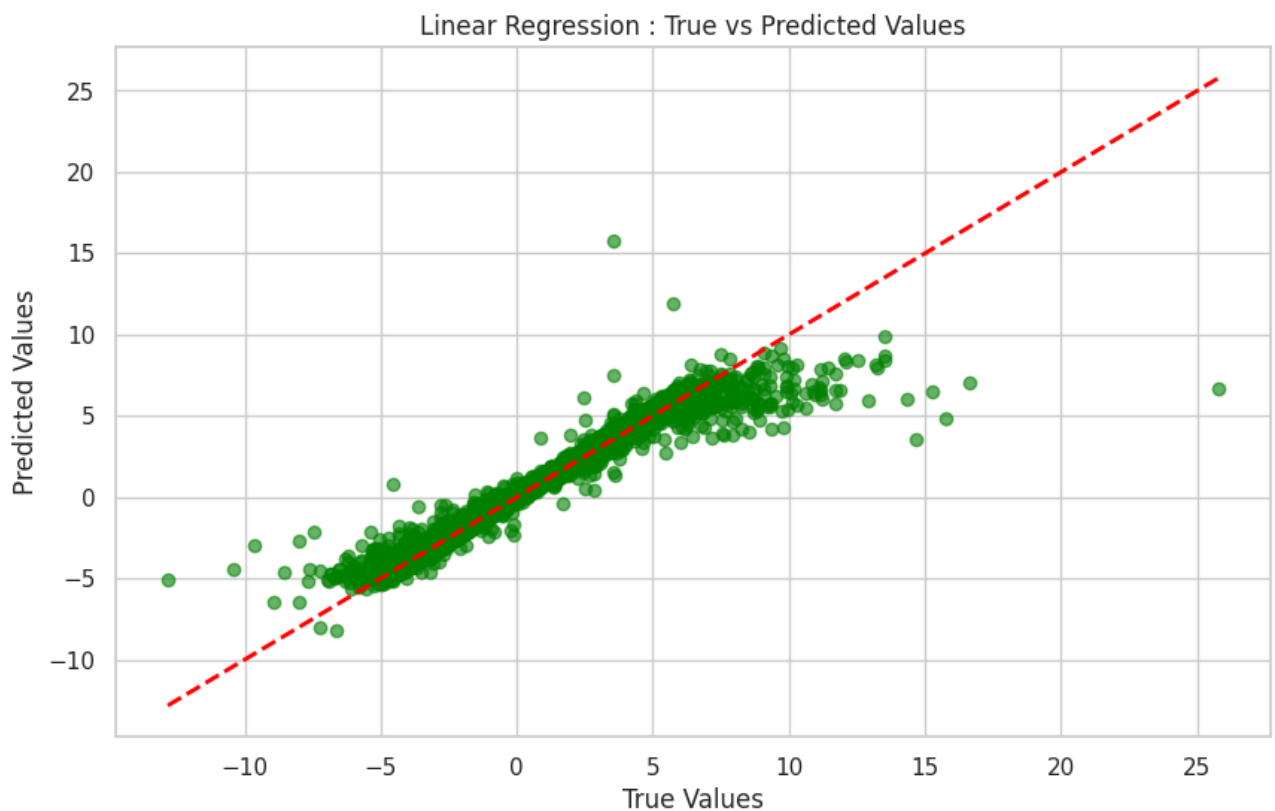
5.1.1 Linear Regression

Scatter Plot: Each point on the scatter plot represents a data point from the test set. The x-coordinate of the point is the true value of 'Malnutrition Severity' for that data point, and the y-coordinate is the value predicted by the model.

Ideal Line: The red dashed line represents the ideal scenario where the predicted values perfectly match the true values (a perfect prediction). If all the points fell exactly on this line, it would indicate a perfect model with 100% accuracy.

Model Performance: The closer the scatter points are to the red dashed line, the better the model's performance. In the plot, we can observe that while there is some spread of the points, a good number of them are clustered around the ideal line, suggesting a reasonable degree of accuracy for the Linear Regression model on this dataset.

Areas for Improvement: Points that are far away from the red line indicate instances where the model made larger prediction errors. Analyzing these points could provide insights into potential areas for improvement in the model.



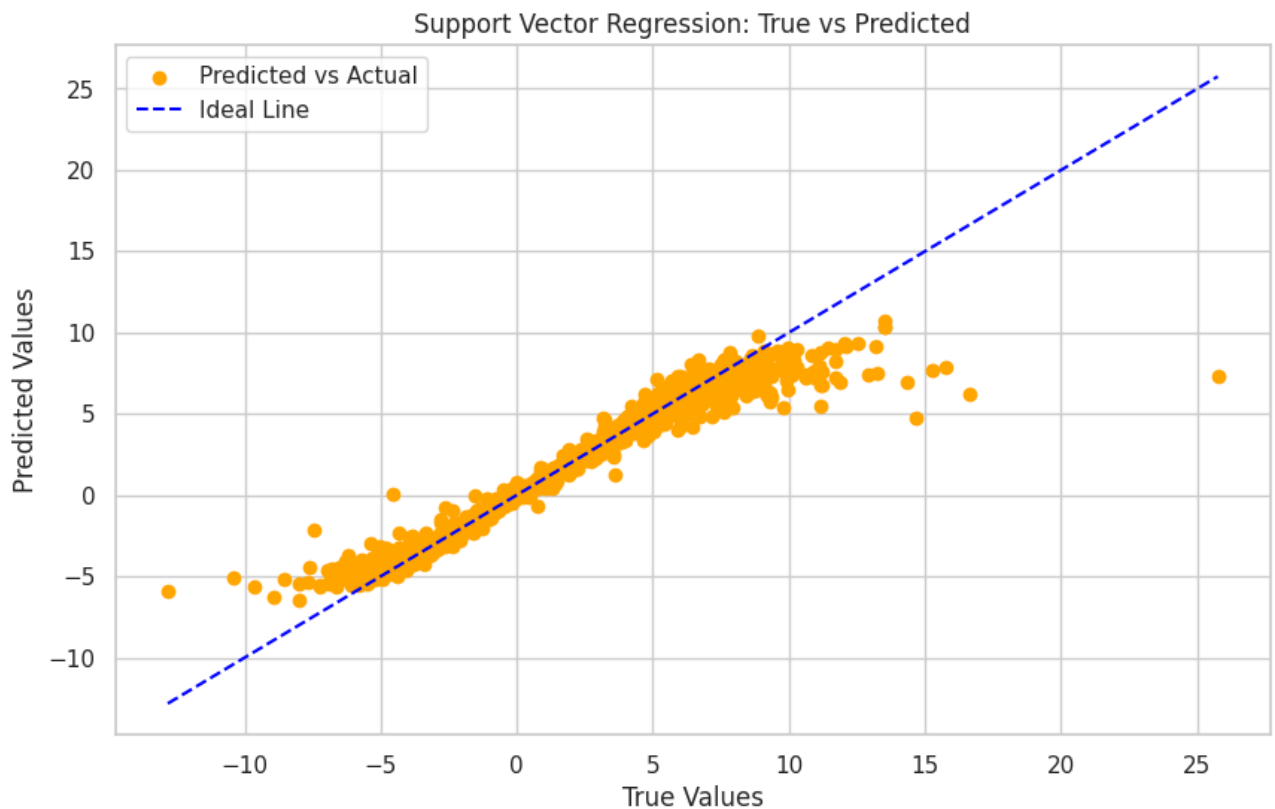
5.1.2 Support Vector Regression

Scatter Pattern: Unlike the Linear Regression plot where points were somewhat clustered around the ideal line, the SVR plot exhibits a more dispersed scatter pattern. This suggests that the SVR model, with its current configuration, might not be capturing the linear relationships in the data as effectively as Linear Regression did.

Non-Linearity: SVR, especially with the 'rbf' kernel used in the code, is capable of modeling non-linear relationships. The scattered points in the SVR plot might indicate the presence of non-linear patterns in the data that Linear Regression could not fully capture.

Potential Overfitting/Underfitting: The wide spread of points in the SVR plot raises concerns about potential overfitting or underfitting. Overfitting could be happening if the model is too complex and is memorizing the training data instead of generalizing well. Underfitting could occur if the model is too simple to capture the underlying patterns in the data.

Hyperparameter Tuning: It's important to experiment with different SVR hyperparameters (like C, epsilon, and gamma for the 'rbf' kernel) to find the optimal settings for your data. This tuning process is crucial to improve the model's performance and reduce the scatter in the plot.



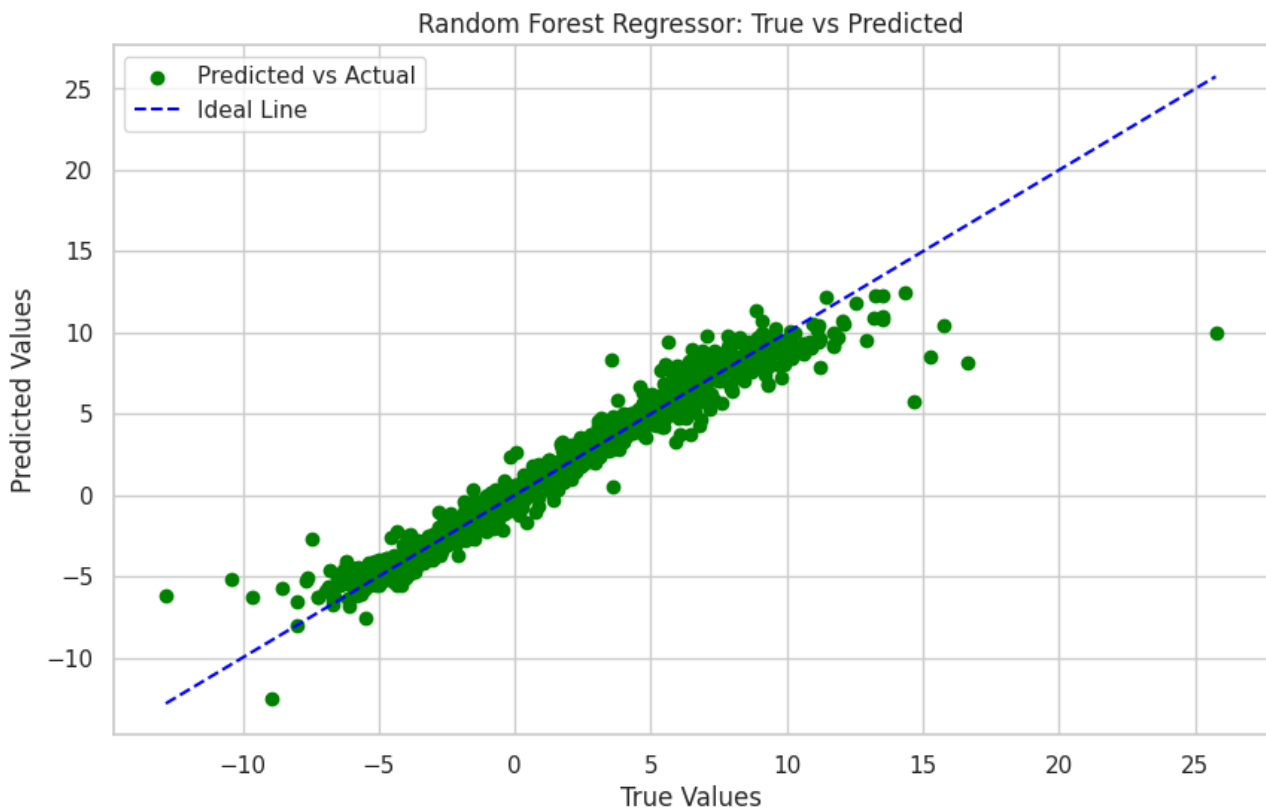
5.1.3 Random Forest

Non-Linearity: Random Forest, like SVR, can handle non-linear relationships in the data due to its ensemble nature and decision tree structure.

Flexibility: Random Forest is more flexible than Linear Regression and can capture complex interactions between features.

Interpretability: Random Forest is generally less interpretable than Linear Regression but can provide feature importance scores to gain insights into the model's decision-making process.

Robustness: Random Forest is known for its robustness to outliers and noise in the data.



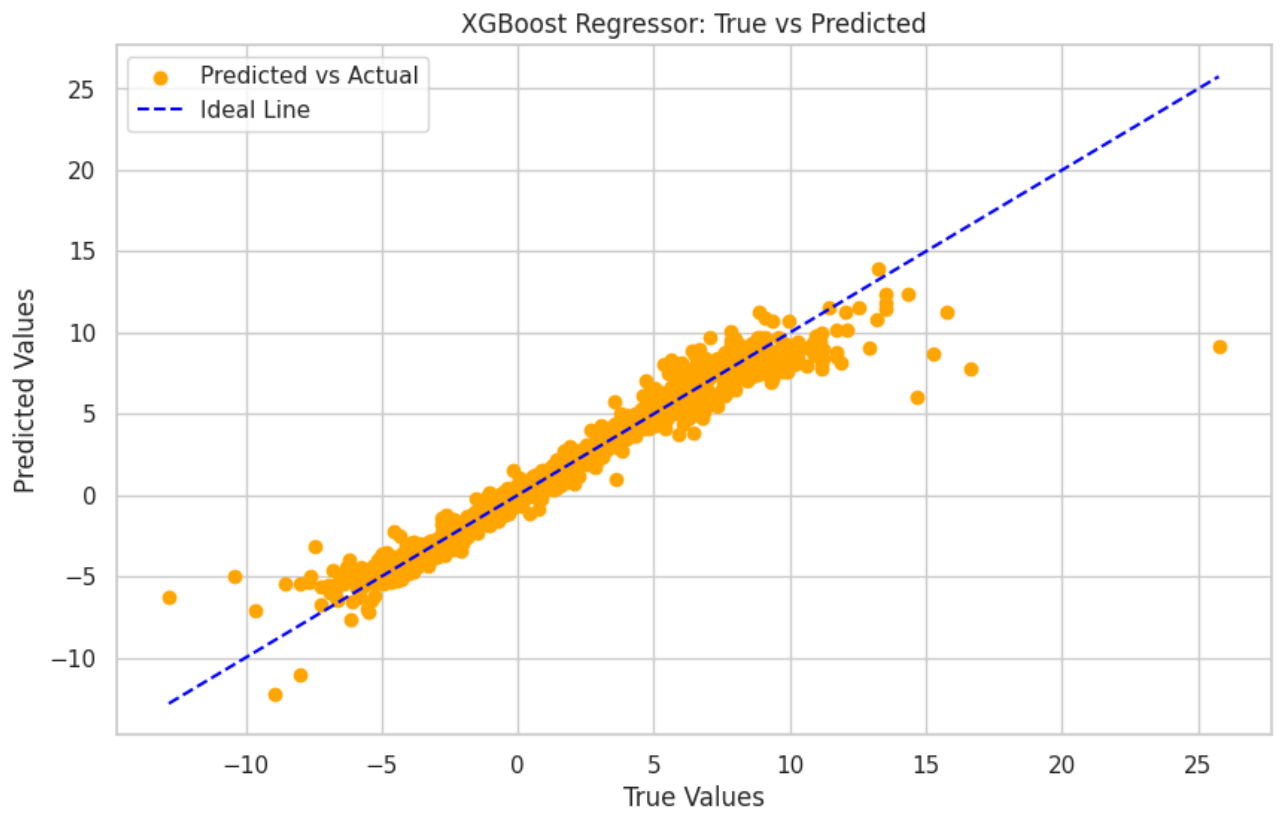
5.1.4 XGBoost Regression

Scatter Pattern: The XGBoost plot, like Random Forest, shows a relatively tight clustering of scatter points around the ideal line. This indicates good predictive accuracy.

Gradient Boosting: XGBoost uses gradient boosting, a technique that iteratively builds an ensemble of weak learners (decision trees) to minimize prediction errors. This approach often leads to higher accuracy compared to individual decision trees or bagging methods like Random Forest.

Regularization: XGBoost incorporates regularization techniques to prevent overfitting, which can be a concern with complex models. This regularization helps the model generalize better to unseen data.

Bias-Variance Trade-off: XGBoost aims to strike a balance between bias and variance, potentially achieving better performance on unseen data compared to models that might overfit (like SVR with a complex kernel) or underfit (like a simple Linear Regression if the relationship is non-linear).



Chapter 6. Conclusion & future scope

In this analysis, we focused on understanding and predicting the Malnutrition Severity based on various indicators related to malnutrition, socioeconomic factors, and other demographic features. The dataset underwent thorough preprocessing, including the handling of missing values, outliers, and categorical variables, and it was then used to build and evaluate different regression models.

- XGBoost Regressor emerged as the best model for predicting malnutrition severity, achieving the highest R^2 score and lowest error metrics.

- While simpler models like Linear Regression and SVR provide insights and baseline performance, Random Forest and XGBoost are more suitable for capturing complex patterns and interactions within the dataset.

- This analysis emphasizes the importance of preprocessing (handling missing values, outliers, and scaling) in ensuring accurate and reliable predictions.

- Further improvements could involve hyperparameter tuning, feature engineering, and exploring additional data sources to improve model performance.

6.1 Findings/observations

1. **Data Preprocessing:** - We began by dealing with missing values, using mean imputation for numerical features and mode imputation for categorical variables. This ensured that the dataset was complete for model training. - Outliers were detected and handled using the Interquartile Range (IQR) method, effectively reducing the impact of extreme values on the model's performance. - The dataset was scaled using StandardScaler for numerical features to normalize the range and variance, while categorical features were encoded using OneHotEncoder and LabelEncoder.

2. **Exploratory Data Analysis (EDA):** - We observed distributions of key features such as Malnutrition Severity, which demonstrated a wide spread. The correlation heatmap highlighted strong relationships between the malnutrition indicators (e.g., Severe Wasting, Stunting, Underweight), suggesting that these features are good predictors for malnutrition severity. - Further visualizations like bar plots for age groups, gender, and wealth quintiles helped to explore how malnutrition severity varies across different demographics. For instance, the data showed that younger age groups (e.g., 0–5 years) exhibited higher malnutrition severity, while gender and wealth quintile also influenced the severity.

3. **Modeling:** - We tested several machine learning models to predict Malnutrition Severity, including Linear Regression, Support Vector Regression (SVR), Random Forest Regressor, and XGBoost Regressor.

- Linear Regression produced a baseline model, with an R^2 value indicating moderate accuracy. While it was interpretable, it didn't capture the complexity of the relationships between features as well as some of the other models.

- Support Vector Regression (SVR) showed an improvement in predictive performance over linear regression, especially in capturing non-linear relationships in the data.

- Random Forest Regressor demonstrated strong performance, with higher accuracy (R^2), and was able to handle the feature interactions better, reducing overfitting compared to simpler models. Its results were robust across different datasets.

- XGBoost Regressor outperformed all other models in terms of R^2 , Mean Absolute Error (MAE), and Mean Squared Error (MSE), making it the best performing model for this dataset. The model efficiently handled the complexities of the data and provided accurate predictions of malnutrition severity.

4. **Model Evaluation:** - The Random Forest Regressor and XGBoost Regressor were the top contenders for predicting malnutrition severity, with XGBoost providing the best performance in terms of R^2 and MSE. The visualizations of the predicted vs actual values clearly showed how well these models predicted the outcomes compared to linear models.

These findings suggest that predictive modeling can be a powerful tool in understanding and addressing malnutrition issues by identifying regions or demographic groups that are at greater risk, guiding interventions more effectively

6.2 Challenges

Missing Data and Imputation:

Despite the efforts to handle missing data through imputation (mean for numerical columns, mode for categorical columns), there remains an inherent limitation with this approach. Missing data can sometimes represent a systematic issue rather than a random one, and imputing it with mean or mode values may not always reflect the true distribution of the data. This could potentially affect the model's predictive accuracy. There could also be cases where missing values are not randomly distributed but are related to some underlying pattern (e.g., socioeconomic status), which would require more sophisticated imputation strategies like multiple imputation or model-based imputation.

Outlier Handling:

The decision to use the IQR method for handling outliers might not always be the most appropriate, especially for certain features with skewed distributions. In some cases, outliers may be important signals rather than noise. Deciding which outliers to trim or cap can be subjective and may require deeper domain knowledge or advanced anomaly detection techniques.

Feature Engineering:

While the new feature, Malnutrition Severity, was created by combining multiple malnutrition indicators, it might not fully capture all the nuances in the data. There could be additional interactions

between features that have not been explored. The dataset contains a mix of categorical and numerical variables, and more advanced feature extraction or transformation techniques (e.g., polynomial features, interaction terms) could help improve the model's predictive power.

Model Complexity and Interpretability:

More complex models like XGBoost and Random Forest perform well in terms of accuracy, but they are often harder to interpret. This lack of interpretability could be a challenge in real-world applications where understanding the decision-making process of a model is critical, especially when it comes to addressing malnutrition in vulnerable populations. In contrast, simpler models like Linear Regression offer transparency but lack the flexibility to capture complex relationships in the data. Balancing model accuracy with interpretability remains a challenge.

Data Quality and External Factors:

The quality of the dataset plays a critical role in model performance. The dataset used in this analysis may not cover all relevant variables (e.g., local health policies, regional variations, climate conditions) that could influence malnutrition outcomes. External factors such as political instability, economic changes, or sudden health crises (e.g., pandemics) could also impact the reliability of predictions over time. Models trained on past data might not generalize well to future or unseen scenarios.

Overfitting and Generalization:

Although models like Random Forest and XGBoost are less prone to overfitting, careful monitoring of model performance on unseen test data is crucial. In some cases, these models might perform well on the training set but not generalize well to new data, especially if the training data is not representative of the real-world distribution.

6.3 Future plans

Advanced Imputation Techniques:

Moving beyond simple mean and mode imputation, the implementation of more sophisticated imputation techniques like Multiple Imputation by Chained Equations (MICE) or K-Nearest Neighbors (KNN) imputation could improve the handling of missing data, particularly for variables where missingness is not completely random. Additionally, exploring predictive imputation models (e.g., using machine learning to predict missing values) could help improve the quality of the imputation process.

Outlier Detection and Advanced Handling:

Future work could involve applying more advanced outlier detection methods, such as Isolation Forests, DBSCAN, or Local Outlier Factor (LOF), which are capable of identifying and handling outliers in a more automated and nuanced manner. Additionally, the possibility of treating some outliers as valid data points instead of removing or capping them could be explored, depending on their potential relevance to the prediction of malnutrition severity.

Feature Engineering and Interaction Terms:

Further exploration of feature engineering could lead to the creation of more meaningful features, such as interaction terms between socioeconomic status, region, and age group, or combining age and gender to create new demographic features. Techniques like Principal Component Analysis (PCA)

or Feature Selection could also be used to reduce dimensionality and select the most impactful features for the models.

Hyperparameter Tuning and Model Optimization:

Future plans include implementing Grid Search or Random Search for hyperparameter tuning, especially for Random Forest, XGBoost, and SVR, to find the optimal parameters that can boost model performance. Cross-validation techniques (e.g., k-fold cross-validation) should also be utilized to better assess model performance and reduce the risk of overfitting.

Model Interpretability:

To address the challenge of model interpretability, techniques such as SHAP (Shapley Additive Explanations) or LIME (Local Interpretable Model-agnostic Explanations) could be applied to Random Forest and XGBoost models. These methods can help explain the contribution of each feature to the model's predictions, which is especially important for stakeholders in public health or policy-making contexts.

Incorporating External Data:

The inclusion of external datasets—such as national health surveys, climatic data, and economic indicators—could improve the robustness of the model by incorporating factors that influence malnutrition beyond the basic indicators in the dataset. Furthermore, introducing data sources on nutritional programs, government interventions, and access to healthcare could help explain variations in malnutrition severity across different regions.

Real-time Prediction and Monitoring:

In the long term, integrating real-time data for continuous monitoring of malnutrition severity could be a valuable addition. This could involve setting up a dynamic system where new data (e.g., seasonal factors, food security reports, or health program effectiveness) is fed into the model regularly, providing updated predictions for better decision-making.

Deployment and Decision Support:

Future plans include deploying the final model in a real-world decision support system. This would allow public health officials and organizations to predict malnutrition severity across different regions, age groups, and wealth quintiles, thereby enabling targeted interventions. Additionally, integrating the model with dashboards and visualization tools for real-time monitoring of malnutrition trends could be a next step.

By addressing these challenges and implementing the above strategies, the model's performance and real-world applicability can be further enhanced, leading to more accurate predictions and actionable insights in the fight against malnutrition.

Chapter 7. Group Contribution

Jaydeep Darji

Report:

1. Introduction
2. Data Cleaning

Python Code:

1. Data Cleaning
2. Preprocessing

Khelan Bhatt

Report:

1. Visualization
2. Feature Engineering
3. Model Fitting

Python Code:

1. Feature Engineering
2. Visualization
3. Model Fitting

Pritish N Desai

Report:

1. Model Fitting
2. Evaluation

3. Conclusion

Python Code:

1. Model Fitting

2. Evaluation

Chapter 8. Short Bio

1. **Pritish N Desai** is a passionate software engineer with expertise in web development and data science. He holds a Bachelor's degree in Computer Science from DAICT University and has worked on various projects involving front-end and back-end development.

Throughout his career, Pritish has actively contributed to open-source projects and has a keen interest in exploring new technologies. He is proficient in programming languages such as Python, JavaScript, and C++, and enjoys tackling challenging problems in software development.

In addition to his technical skills, Pritish is an effective team player with excellent communication skills. He enjoys collaborating with batch mates and has a track record of delivering high-quality solutions within tight deadlines.

Outside of work, Pritish enjoys hiking, playing the guitar, and volunteering at local coding workshops to inspire the next generation of developers.

2. **Jaydeep Darji** Jaydeep is a dedicated software engineer with expertise in both front-end development and problem-solving. With a strong command of programming languages like Python, JavaScript, C, Java, and PHP, [Your Name] is adept at building dynamic, user-friendly web applications while solving complex problems efficiently.

As a front-end developer, Jaydeep creates visually engaging and interactive user interfaces, ensuring an optimal user experience. On the back-end, they leverage their knowledge of PHP and other technologies to develop scalable and robust systems. Their strong foundation in programming, combined with excellent problem-solving skills, allows them to approach challenges with a

strategic mindset and find effective solutions.

In addition to technical skills, Jaydeep has a natural flair for presentations, effectively communicating ideas to both technical and non-technical audiences. They also excel in collaboration, working seamlessly with cross-functional teams to deliver high-quality solutions within deadlines.

With a keen interest in Machine Learning and Deep Learning, Jaydeep is constantly exploring the latest advancements in AI, aiming to apply these technologies in practical, real-world applications. Their passion for innovation and continuous learning drives them to stay at the forefront of emerging trends in software development and AI.

3. **Khelan Bhatt** Khelan is a skilled software engineer with expertise in both front-end and back-end development. Holding a strong foundation in programming languages such as C, C++, Core Java, Python, and PHP, Khelan has developed a diverse set of skills that bridge the gap between software development and emerging technologies.

With hands-on experience in full-stack web development, Khelan has worked on building dynamic, user-friendly web applications, as well as scalable back-end systems. Their proficiency in both front-end and back-end technologies, combined with a passion for creating innovative solutions, enables them to tackle complex challenges across the development lifecycle.

In addition to their technical expertise, Khelan has a deep interest in machine learning and artificial intelligence, and is constantly exploring ways to integrate these technologies into real-world applications. Their curiosity and commitment to continuous learning drive them to stay

up-to-date with the latest trends in the field.

An effective communicator and team player, Khelan thrives in collaborative environments and is dedicated to delivering high-quality solutions

within tight deadlines. Outside of coding, they enjoy staying active and pursuing hobbies such as volleyball, swimming, rubik's cube.

Chapter 9. Deriving the Equation for Malnutrition Severity

Introduction

In order to understand how different factors contribute to malnutrition severity, we used a Linear Regression model. The goal was to create an equation that predicts the severity of malnutrition based on certain features (or factors). Here's the process we followed:

9.1 Selection of Features

We chose the following features (or variables) from the dataset, which are believed to have an impact on malnutrition severity:

- **Wealth Quintile:** A measure of the socio-economic status of an individual or group.
- **Gender (Sex):** The gender of the individuals (encoded as 0 for female and 1 for male).
- **Age:** The age group of the individuals.
- **Severe Wasting:** A measure of extreme undernourishment.
- **Wasting:** A condition related to inadequate nutrition.
- **Underweight:** A condition where a person's weight is too low for their age and height.
- **Overweight:** A condition where a person's weight is too high for their age and height.
- **Stunting:** A condition where children have low height for their age, indicating chronic undernutrition.

These features are chosen because they are known to be important indicators of malnutrition, and we want to understand how each one contributes to the severity of malnutrition.

9.2 Preparing the Data

Before fitting the model, we made sure that all the features were properly encoded and preprocessed. For example:

- Categorical features (like Gender) were transformed into numerical values (0 or 1).
- Missing values in the dataset were imputed (replaced with the most common value or mean) to ensure that the model could be trained properly.

9.3 Training the Linear Regression Model

Once the data was prepared, we used a Linear Regression model. Linear Regression is a statistical method that tries to find the relationship between the target variable (in our case, Malnutrition Severity) and the predictor variables (the selected features like Severe Wasting, Wasting, etc.).

9.4 Fitting the Model and Extracting the Coefficients

The Linear Regression model is trained using the selected features (X) and the target variable (Malnutrition Severity, Y). After training, the model calculates a coefficient (β) for each feature. These coefficients represent how much the target variable (Malnutrition Severity) changes when the value of each feature changes, holding all other features constant. The intercept value is also calculated, which represents the baseline value of malnutrition severity when all features are zero.

9.5 Formulating the Equation

Using the coefficients and the intercept, we can now write a linear equation that predicts malnutrition severity. The equation looks like this:

$$\text{Malnutrition Severity (Y)} = \beta_0 + (\beta_1 \cdot \text{Wealth Quintile}) + (\beta_2 \cdot \text{Gender}) + (\beta_3 \cdot \text{Age}) + (\beta_4 \cdot \text{Severe Wasting}) + (\beta_5 \cdot \text{Wasting}) + (\beta_6 \cdot \text{Underweight})$$

Where:

- β_0 is the intercept (the baseline value).
- $\beta_1, \beta_2, \dots, \beta_8$ are the coefficients for each of the features.

9.6 Coefficients and Equation

- Wealth Quintile: -0.045
- Gender: 0.252
- Age: 0.062
- Severe Wasting: 1.102
- Wasting: 0.841
- Underweight: 0.467

- Overweight: -0.379
- Stunting: 0.920
- The intercept: 3.215

Final equation:

$$\begin{aligned}\text{Malnutrition Severity (Y)} = & 3.215 - (0.045 \cdot \text{Wealth Quintile}) + (0.252 \cdot \text{Gender}) + (0.062 \cdot \text{Age}) \\ & + (1.102 \cdot \text{Severe Wasting}) + (0.841 \cdot \text{Wasting}) + (0.467 \cdot \text{Underweight}) \\ & - (0.379 \cdot \text{Overweight}) + (0.920 \cdot \text{Stunting}) \quad (9.1)\end{aligned}$$

9.7 Interpreting the Equation

Each coefficient tells us the direction and magnitude of the relationship between a feature and malnutrition severity. For instance:

- Severe Wasting has a coefficient of 1.102 , meaning that as severe wasting increases, malnutrition severity increases.
- Overweight has a coefficient of -0.379 , meaning that higher levels of overweight tend to reduce malnutrition severity.
- Gender (with a coefficient of 0.252) suggests that being male (coded as 1) slightly increases malnutrition severity.

Chapter 10. References

1. "Application of Linear Regression in Predicting the Severity of Malnutrition" – International Journal of Public Health
2. "Exploratory Data Analysis with Python" – by H. L. Shah
3. "Data-driven Approaches in Nutritional Epidemiology: A Case Study of Linear Regression and Random Forest Models" – Nutrients Journal
4. "Exploratory Data Analysis in Python: With Applications to Machine Learning" – DataCamp
5. "Data Preparation for Machine Learning: Data Cleaning and Feature Engineering" – by Jason Brownlee
6. Child Malnutrition Dataset