1. Data Collection and Data preprocessing

The dataset was directly uploaded to the Google Colab environment then necessary preprocessing steps were taken. The codes and the outputs are given below.

- There were no null values in the data.
- I replaced the values female and male with 0 and 1 respectively.
- Changed the name of some columns
- Replaced yes and no with 1 and 0 respectively for several columns (Family History With Overweight, Eat High Caloric Food Frequently, Smoking, etc..)

```
import pandas as pd

df=pd.read_csv('/content/ObesityDataSet_raw_and_data_sinthetic.csv')
```

```
[24] df.head()
```

	Gender	Age	Height	Weight	family_history_with_overweight	FAVC	FCVC	NCP	CAEC	SMOKE	CH2O	SCC	FAF	TUE	CALC	MTRANS	NObeyesdad
0	Female	21.0	1.62	64.0	yes	no	2.0	3.0	Sometimes	no	2.0	no	0.0	1.0	no	Public_Transportation	Normal_Weight
1	Female	21.0	1.52	56.0	yes	no	3.0	3.0	Sometimes	yes	3.0	yes	3.0	0.0	Sometimes	Public_Transportation	Normal_Weight
2	Male	23.0	1.80	77.0	yes	no	2.0	3.0	Sometimes	no	2.0	no	2.0	1.0	Frequently	Public_Transportation	Normal_Weight
3	Male	27.0	1.80	87.0	no	no	3.0	3.0	Sometimes	no	2.0	no	2.0	0.0	Frequently	Walking	Overweight_Level_I
4	Male	22.0	1.78	89.8	no	no	2.0	1.0	Sometimes	no	2.0	no	0.0	0.0	Sometimes	Public_Transportation	Overweight_Level_II

```
#checking and counting missing values if present

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

	0
Gender	0
Age	0
Height	0
Weight	0
family_history_with_overweight	0
FAVC	0
FCVC	0
NCP	0
CAEC	0
SMOKE	0
CH2O	0
SCC	0
FAF	0
TUE	0
CALC	0
MTRANS	0
NObeyesdad	0

Figure 01: dataset collection and viewing

```python
[17] # Data preprocessing

# Gender
df['Gender'] = df['Gender'].replace({"Female": 0, "Male": 1})

# Age
df['Age'] = df['Age'].astype(int)

# Height
df['Height'] = df['Height'].round(2)

# Weight
df['Weight'] = df['Weight'].round(1)

# Rename columns
new_column_names = {
    'family_history_with_overweight': 'Family_History_With_Overweight',
    'FAVC': 'Eat_High_Caloric_Food_Frequently',
    'FCVC': 'Vegetable_Consumption_Frequency',
    'NCP': 'Number_Of_Main_Meals_Daily',
    'CAEC': 'Consumption_Of_Food_Between_Meal',
    'SMOKE': 'Smoking',
    'CH2O': 'Liquid_Intake_Daily',
    'SCC': 'Calorie_Consumption_Monitoring',
    'FAF': 'Physical_Activity',
    'TUE': 'Time_Using_Technological_Devices',
    'CALC': 'Alcohol_Consumption',
    'MTRANS': 'Type_Of_Transportation',
    'NObeyesdad': 'Obesity_Level'
}
df = df.rename(columns=new_column_names)

# Family History With Overweight
df['Family_History_With_Overweight'] = df['Family_History_With_Overweight'].replace({"no": 0, "yes": 1})

# Eat High Caloric Food Frequently
df['Eat_High_Caloric_Food_Frequently'] = df['Eat_High_Caloric_Food_Frequently'].replace({"no": 0, "yes": 1})
```

```python
# Vegetable Consumption Frequency
df['Vegetable_Consumption_Frequency'] = df['Vegetable_Consumption_Frequency'].astype(int)

# Number Of Main Meals Daily
df['Number_Of_Main_Meals_Daily'] = df['Number_Of_Main_Meals_Daily'].astype(int)

# Consumption Of Food Between Meal
df['Consumption_Of_Food_Between_Meal'] = df['Consumption_Of_Food_Between_Meal'].replace({
    "no": 0, "Sometimes": 1, "Frequently": 2, "Always": 3})

# Smoking
df['Smoking'] = df['Smoking'].replace({"no": 0, "yes": 1})

# Liquid Intake Daily
df['Liquid_Intake_Daily'] = df['Liquid_Intake_Daily'].round(1)

# Calorie Consumption Monitoring
df['Calorie_Consumption_Monitoring'] = df['Calorie_Consumption_Monitoring'].replace({"no": 0, "yes": 1})

# Physical Activity
df['Physical_Activity'] = df['Physical_Activity'].astype(int)

# Time Using Technological Devices
df['Time_Using_Technological_Devices'] = df['Time_Using_Technological_Devices'].round(2)

# Alcohol Consumption
df['Alcohol_Consumption'] = df['Alcohol_Consumption'].replace({
    "no": 0, "Sometimes": 1, "Frequently": 2, "Always": 3})

# Type Of Transportation
df['Type_Of_Transportation'] = df['Type_Of_Transportation'].replace({
    "Automobile": 1, "Bike": 2, "Motorbike": 3, "Public_Transportation": 4, "Walking": 5})

# Obesity Level
df['Obesity_Level'] = df['Obesity_Level'].replace({
    'Insufficient_Weight': 0, 'Normal_Weight': 1,
    'Overweight_Level_I': 2, 'Overweight_Level_II': 2,
    'Obesity_Type_I': 3, 'Obesity_Type_II': 3, 'Obesity_Type_III': 3})
```

Figure 02: code snippet of preprocessing

1. Data visualization and distribution analysis

1. Numerical Distribution Analysis

For continuous or numerical variables, distribution analysis helps identify the shape, spread, and central tendency of the data.

That includes:

- age
- Height
- Weight
- Liquid intake

Skewness: Measures the asymmetry of the distribution. A skewness of 0 means the data is perfectly symmetrical. Positive skew indicates a long tail on the right, and negative skew means a long tail on the left.

Kurtosis: Measures the "tailedness" or the peak of the distribution. Higher kurtosis means more data is concentrated in the tails.

Visualizations:

- Boxplot: Shows the distribution through quartiles and highlights potential outliers.
- Histogram: Displays the frequency distribution of a variable.
- KDE Plot (Kernel Density Estimate): Shows the probability density function of the variable.

2. Categorical Distribution Analysis

For categorical variables, distribution analysis helps understand the frequency or proportion of each category.

That includes:

- Gender
- Family history with overweight
- Eat high calory foods
- Smoking… etc

Visualizations:

- Count plot: Shows the count of occurrences for each category.
- Pie Chart: Illustrates the proportion of each category as slices of a pie.

```python
# Data visualization and Distribution analysis

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import skew, kurtosis

# Load the dataset
DataDF = pd.read_excel("Obesity_DataSet.xlsx")

# Gender Distribution: Countplot and Piechart
plt.figure(figsize=(5,3))
sns.countplot(data=DataDF, x='Gender', palette='muted')
plt.title('Gender Distribution')
plt.show()

# Piechart for Gender
gender_count = DataDF['Gender'].value_counts()
plt.figure(figsize=(4,4))
plt.pie(gender_count.values, labels=['Male', 'Female'], autopct='%1.1f%%', colors=['#008fd5','#e5ae37'])
plt.title("Gender Distribution")
plt.legend(['Male', 'Female'], loc='best')
plt.show()

# Function to plot skewness, kurtosis, boxplot, and histogram for numerical columns
def plot_column_distribution(column, xlabel, hue_column=None):
    # Skewness and Kurtosis
    print(f"Skewness of '{column}':", DataDF[column].skew().round(2))
    print(f"Kurtosis of '{column}':", DataDF[column].kurtosis().round(2))

    # Boxplot
    sns.boxplot(x=DataDF[column], color='#008fd5')
    plt.title(f'{xlabel} Distribution')
    plt.show()

    # Histogram
    plt.figure(figsize=(10,5))
    sns.histplot(data=DataDF, x=column, hue=hue_column, multiple='stack', kde=True, palette='colorblind')
    plt.title(f'{xlabel} Distribution')
    plt.show()
```

```python
[20] # Age Distribution
plot_column_distribution('Age', 'Age', hue_column='Obesity_Level')

# Height Distribution
plot_column_distribution('Height', 'Height', hue_column='Obesity_Level')

# Weight Distribution
plot_column_distribution('Weight', 'Weight', hue_column='Obesity_Level')

# Function to plot countplot and pie chart for categorical columns
def plot_categorical_distribution(column, labels):
    plt.figure(figsize=(5,3))
    sns.countplot(data=DataDF, x=column, palette='muted')
    plt.title(f'{column} Distribution')
    plt.show()

    count = DataDF[column].value_counts()
    plt.figure(figsize=(4,4))
    plt.pie(count.values, labels=labels, autopct='%1.1f%%', colors=['#008fd5','#e5ae37','#fc4f30','#6d904f'])
    plt.title(f'{column} Distribution')
    plt.legend(labels, loc='best')
    plt.show()

# Family History With Overweight
plot_categorical_distribution('Family_History_With_Overweight', ['Yes', 'No'])

# Eat High Caloric Food Frequently
plot_categorical_distribution('Eat_High_Caloric_Food_Frequently', ['Yes', 'No'])

# Vegetable Consumption Frequency
plot_categorical_distribution('Vegetable_Consumption_Frequency', ['Low', 'Normal', 'High'])

# Number Of Main Meals Daily
plot_categorical_distribution('Number_Of_Main_Meals_Daily', ['1', '2', '3', '4'])

# Consumption Of Food Between Meal
plot_categorical_distribution('Consumption_Of_Food_Between_Meal', ['Sometimes', 'Frequently', 'Always', 'No'])

# Smoking
plot_categorical_distribution('Smoking', ['No', 'Yes'])

# Liquid Intake Daily Distribution
plot_column_distribution('Liquid_Intake_Daily', 'Liquid Intake')
```

Figure 03: Code snippet of data visualization and distribution(brief explanation is in the comments)

1. **Gender**

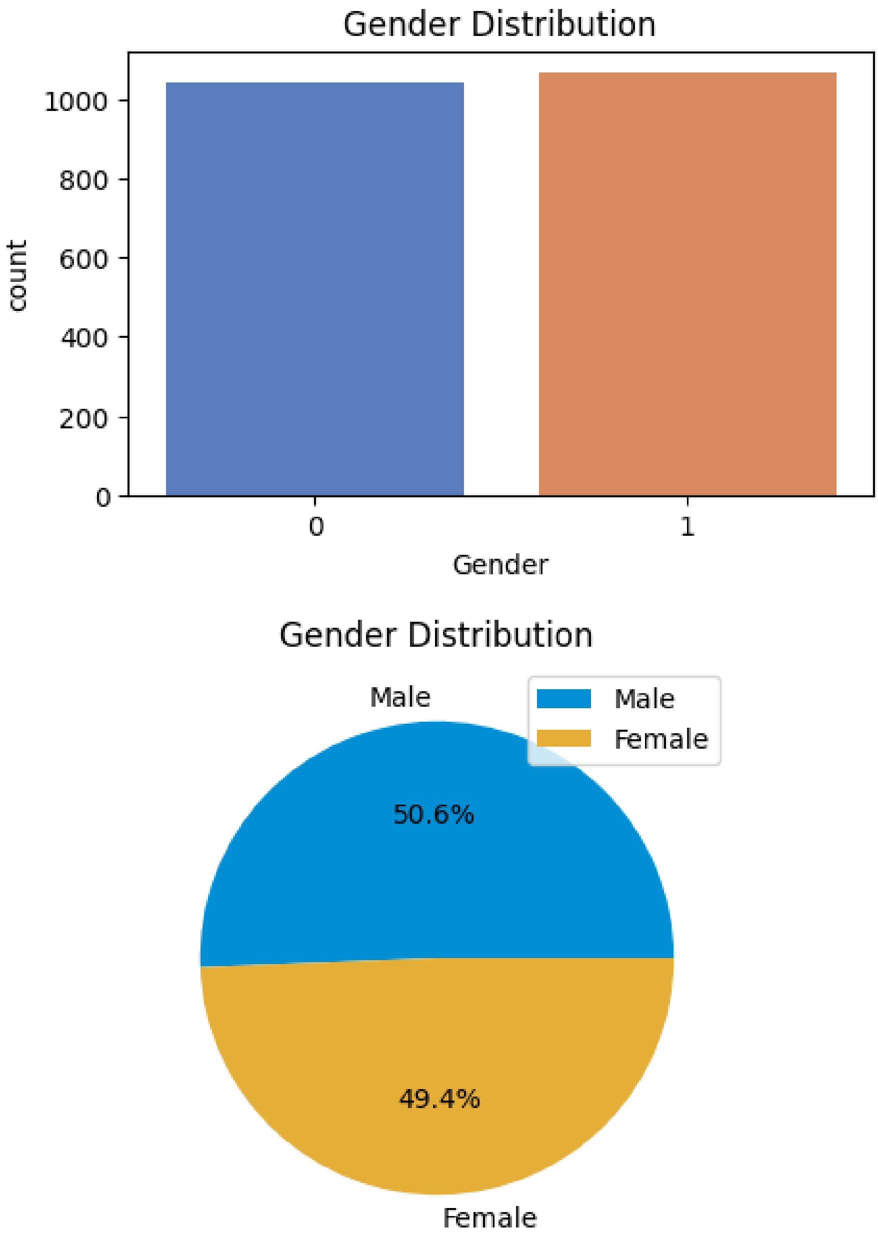
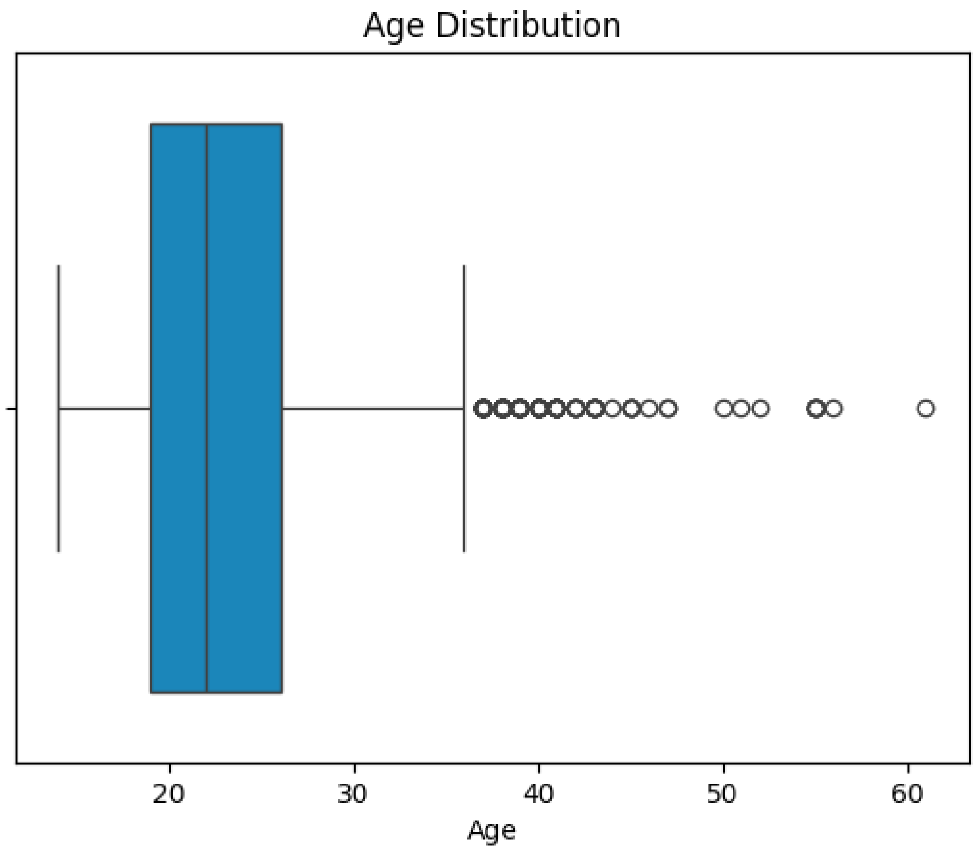


Figure 04: Gender variable visualization

2. Age



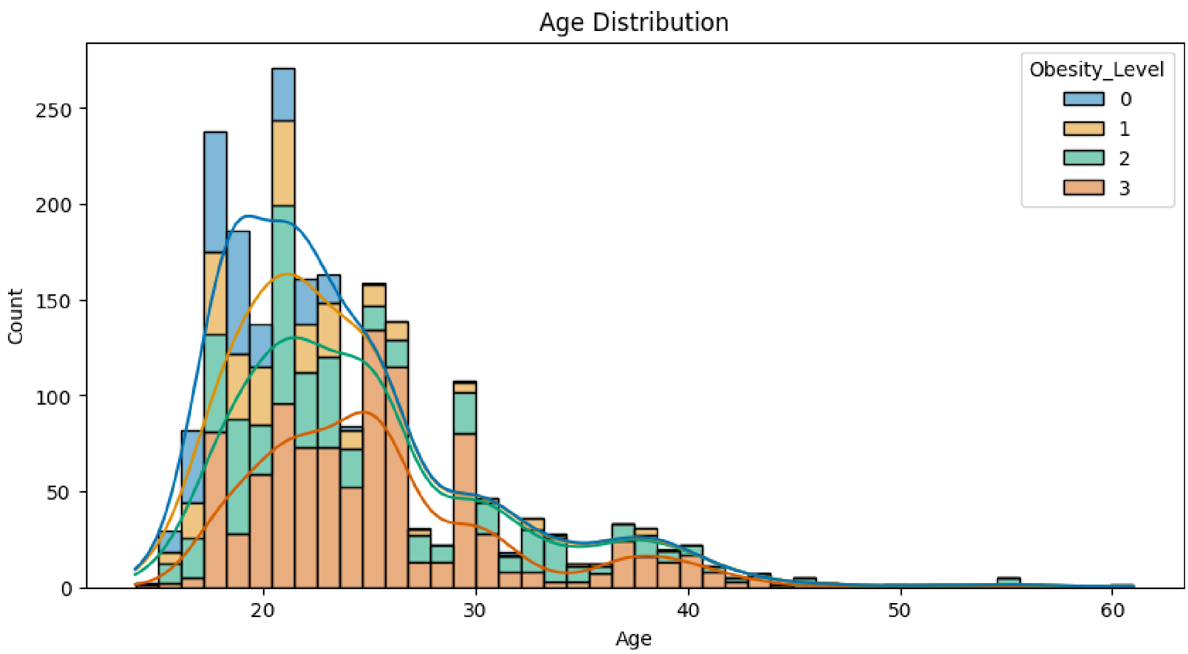
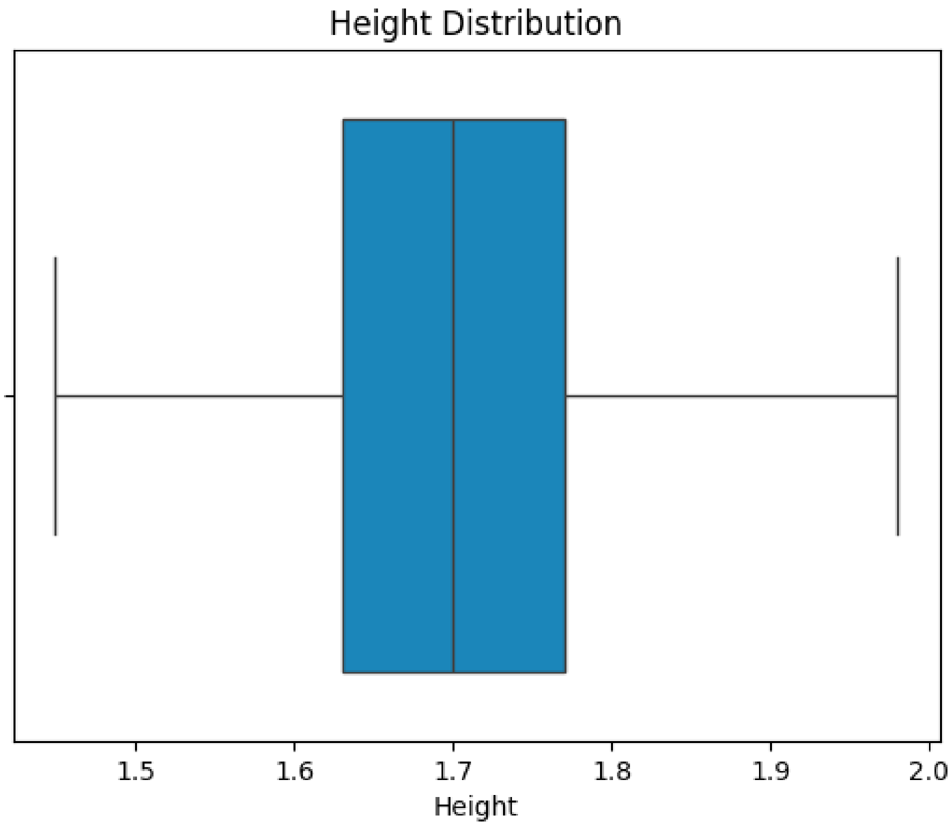


Figure 05: Age Variable visualization and Distribution

- Skewness of 'Age': 1.56
- Kurtosis of 'Age': 2.99

3. Height



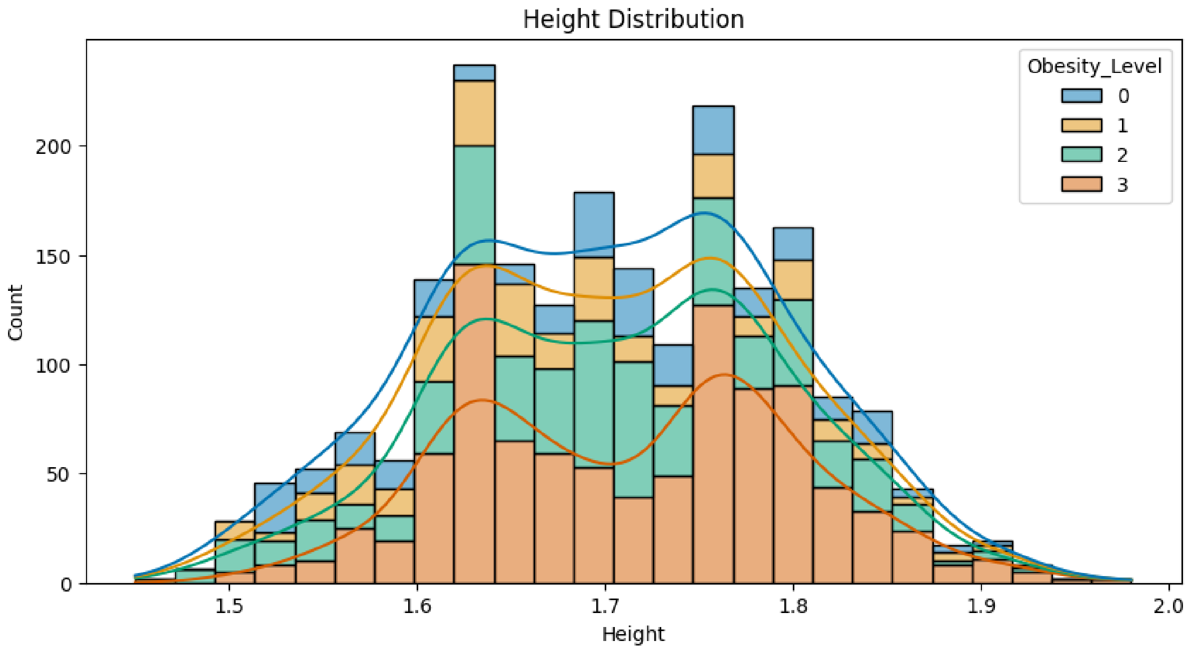


Figure 06: Height Variable visualization and Distribution

- Skewness of 'Height': -0.01
- Kurtosis of 'Height': -0.57

4. Weight

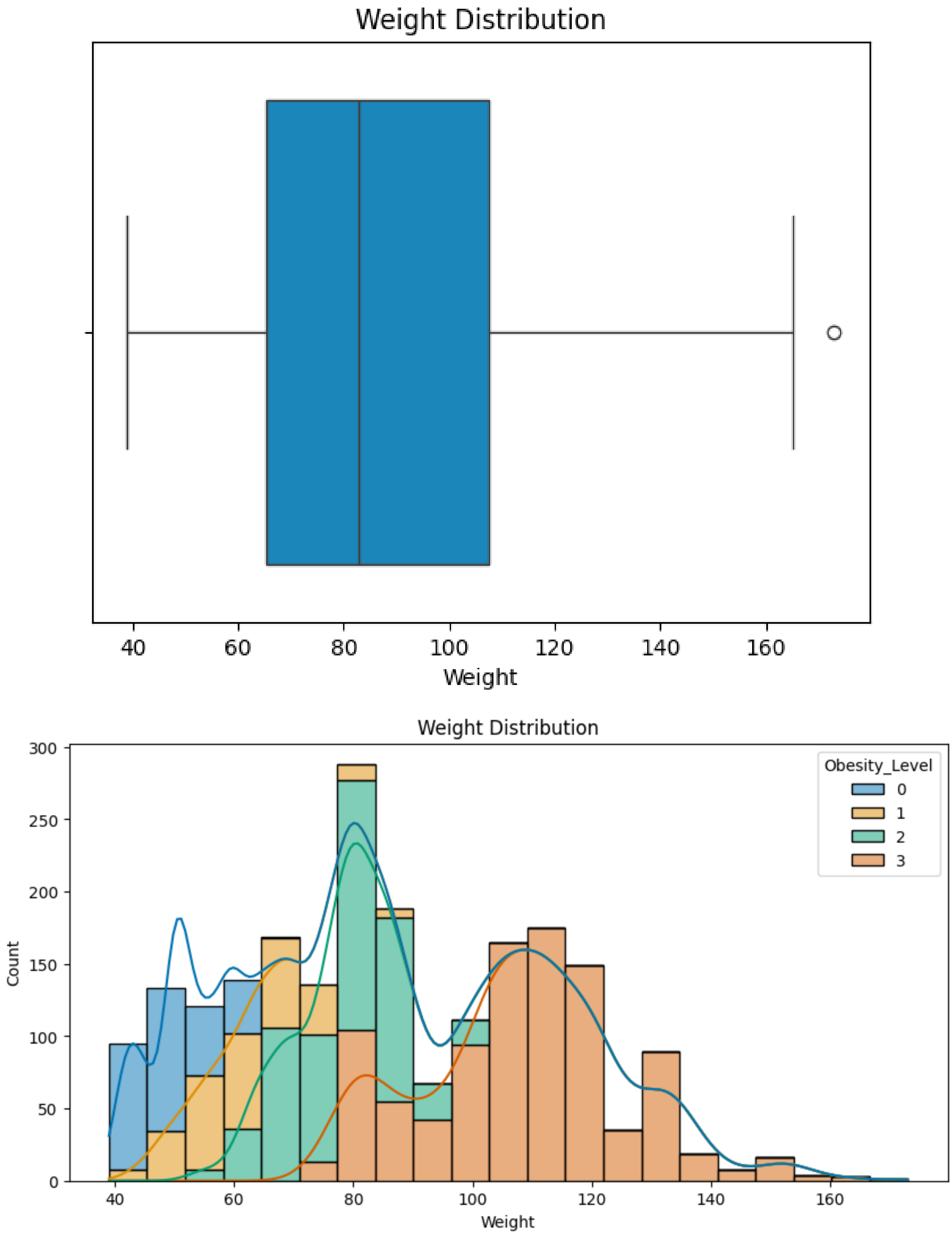


Figure 07: Weight Variable visualization and Distribution

- Skewness of 'Weight': 0.26
- Kurtosis of 'Weight': -0.7

5. Family history with overweight

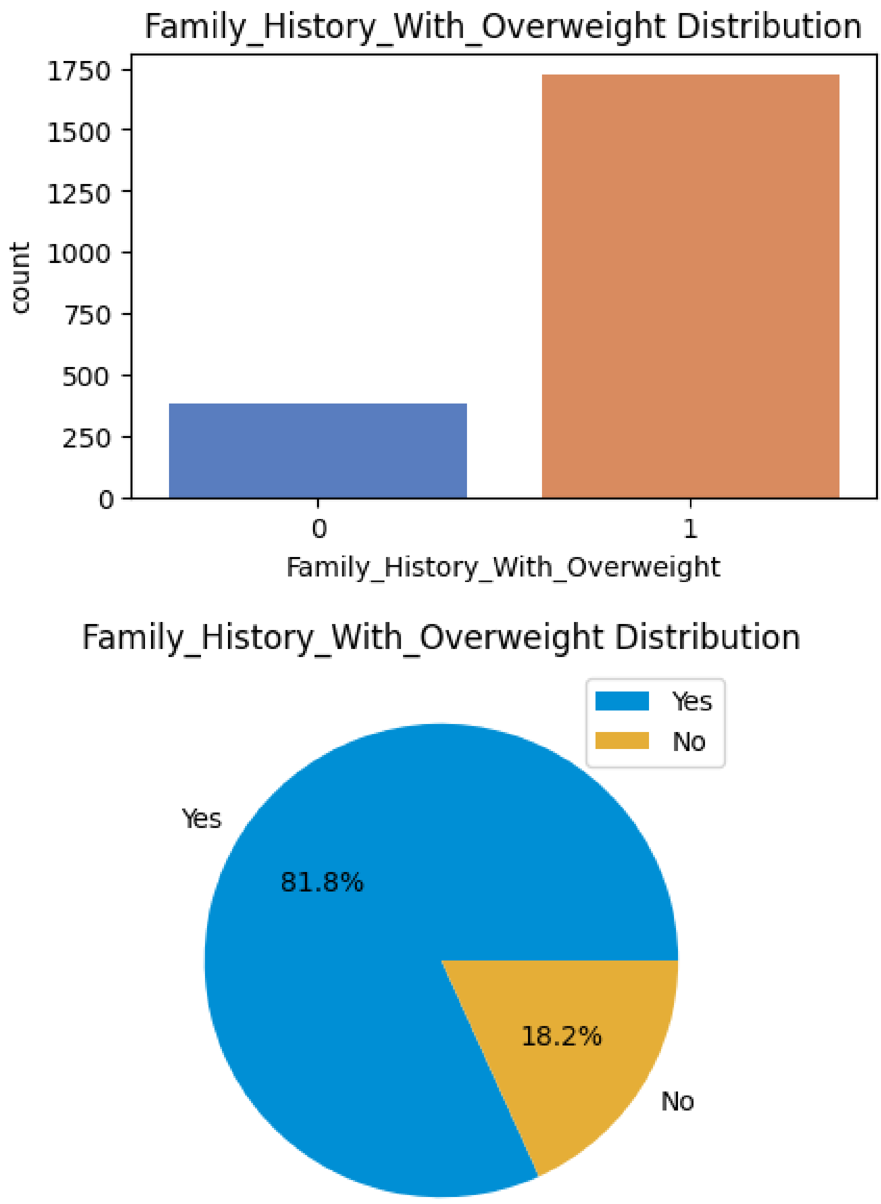
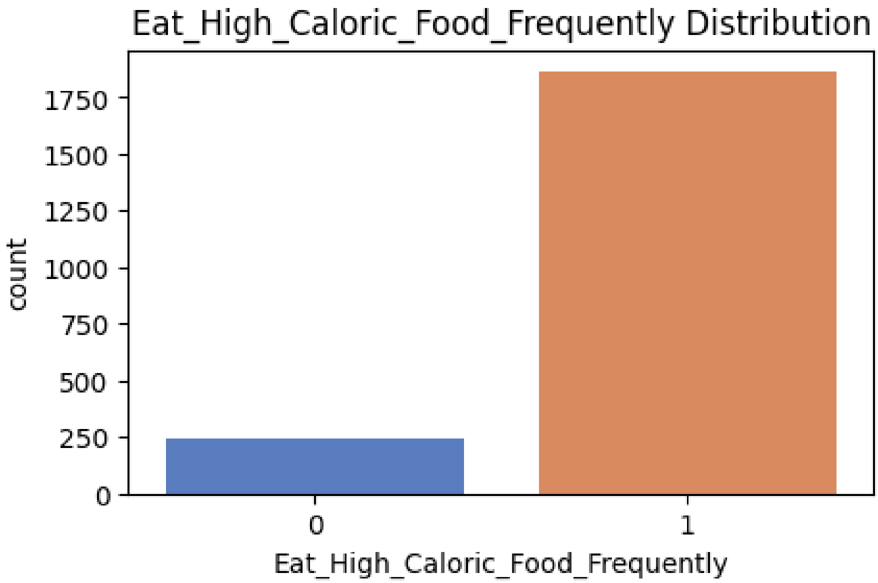


Figure 08: Family history with overweight Variable visualization

6. High calory food consumption



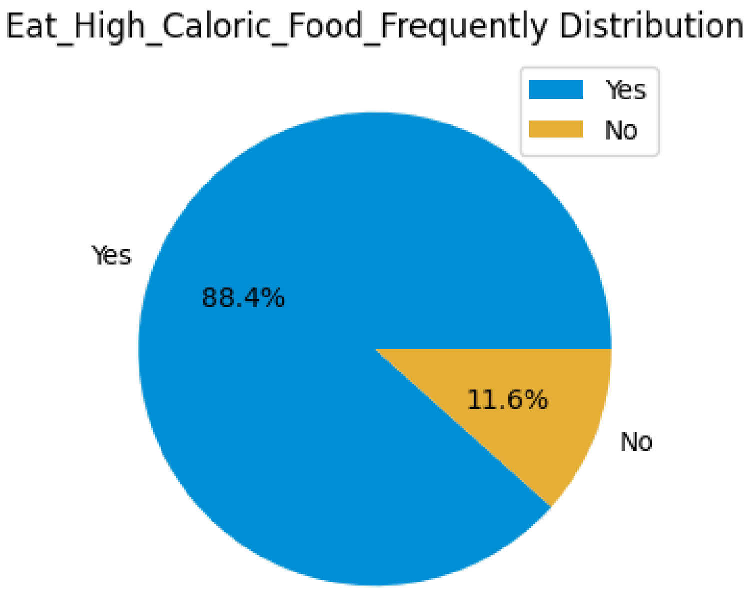
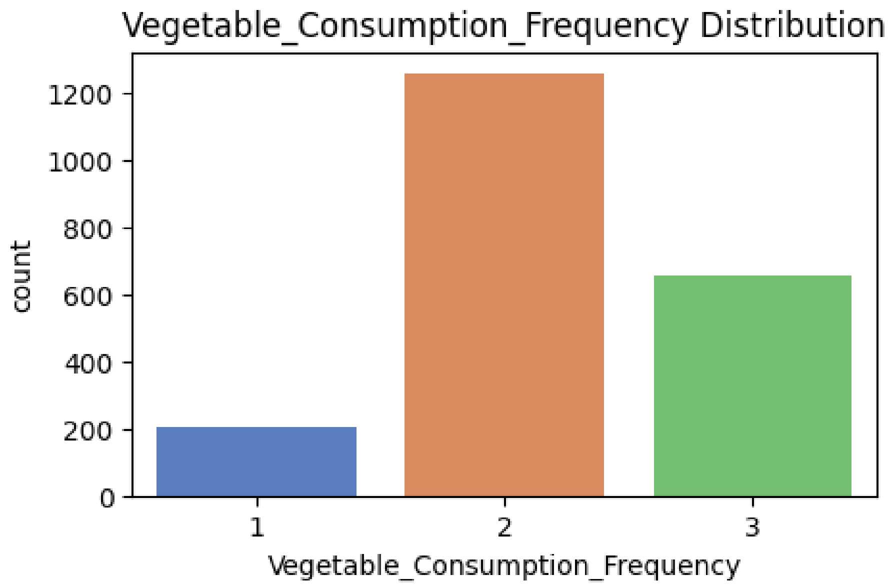


Figure 09: Eating high calory food Variable visualization

7. Vegetable consumption frequency



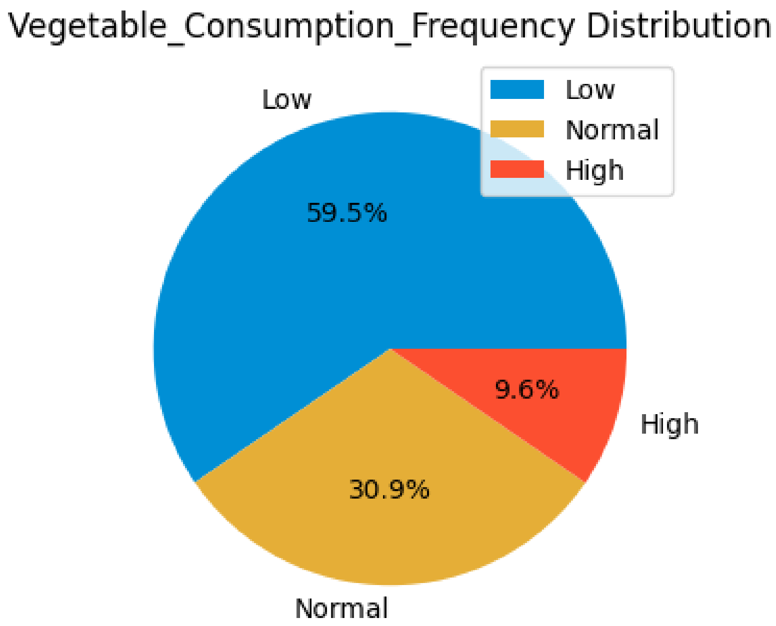


Figure 10: Vegetable consumption Variable visualization

8. Daily main meal consumption

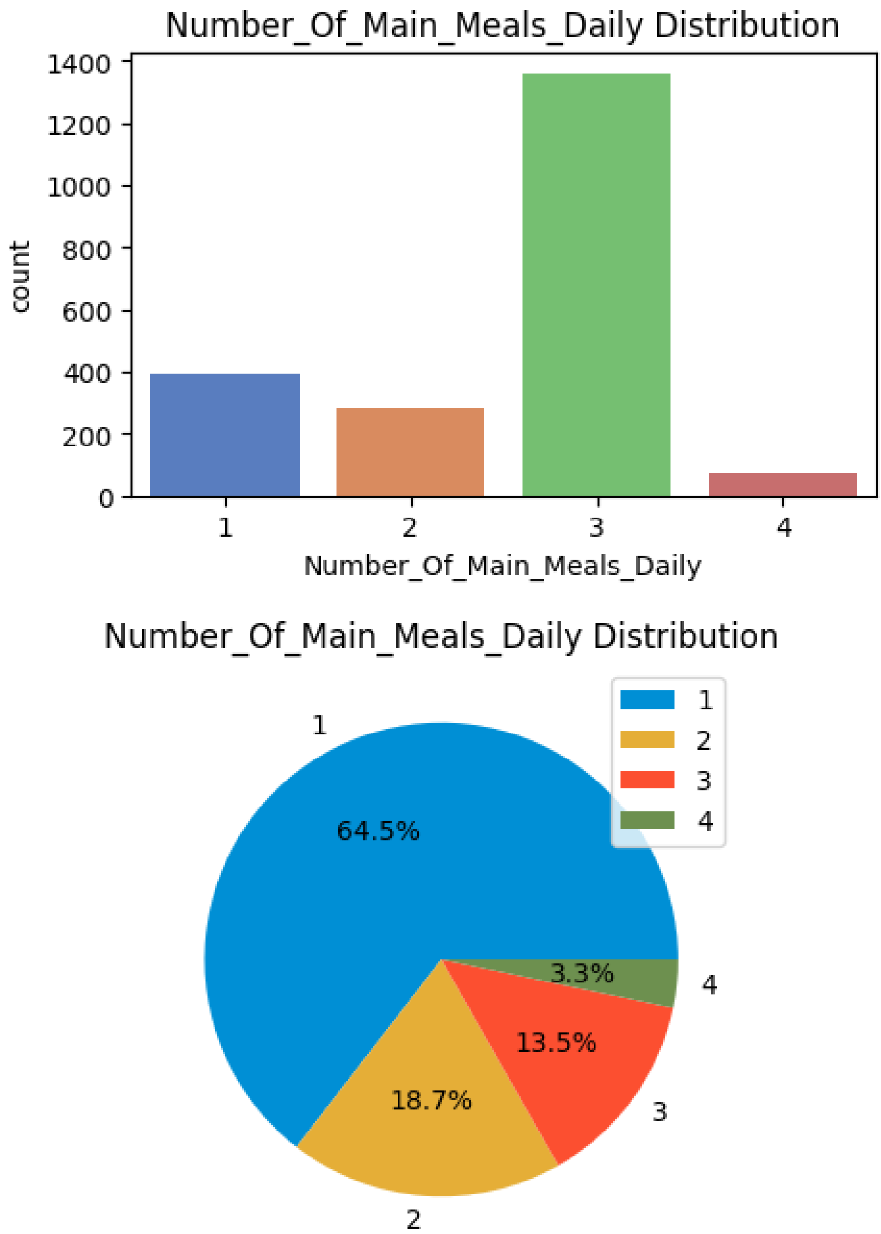


Figure 11: Number of main meals daily Variable visualization

9. Consumption of food between meal

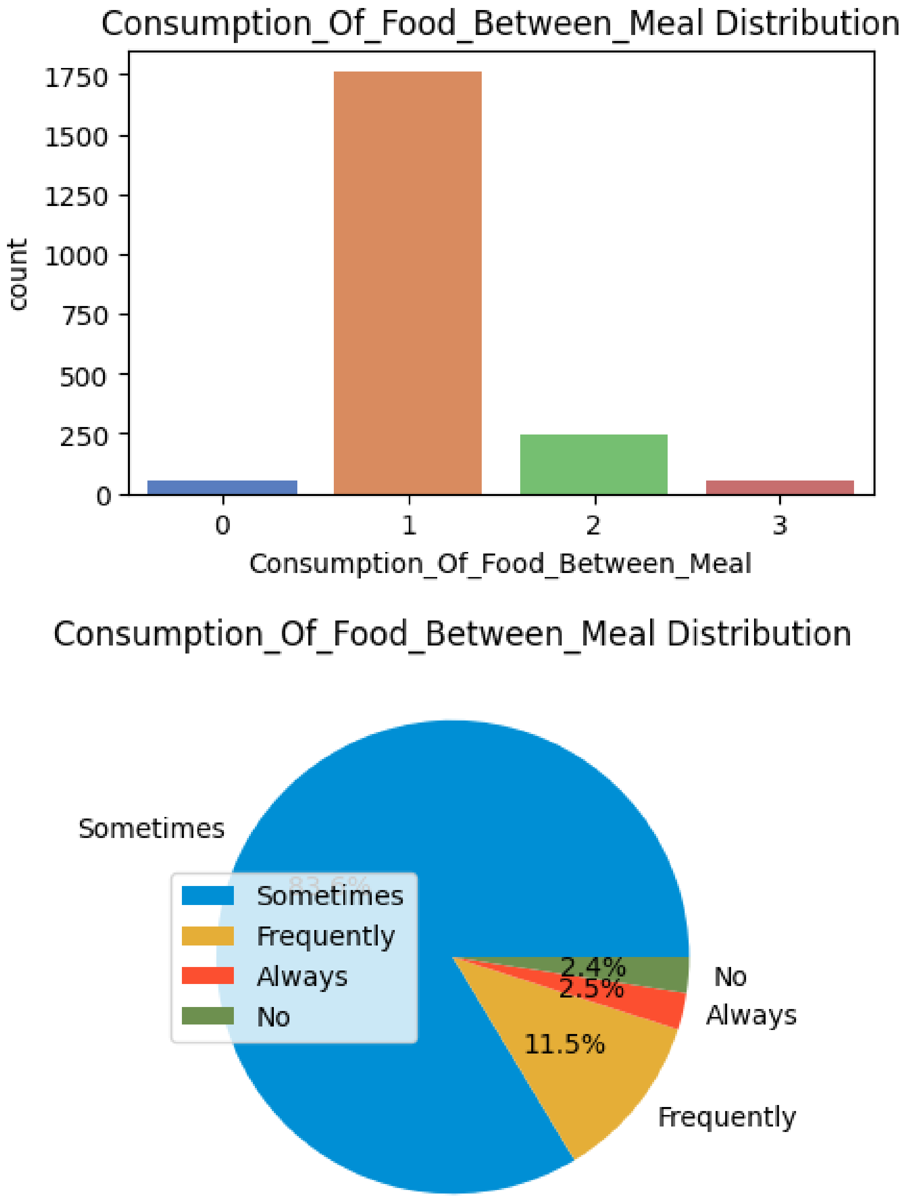
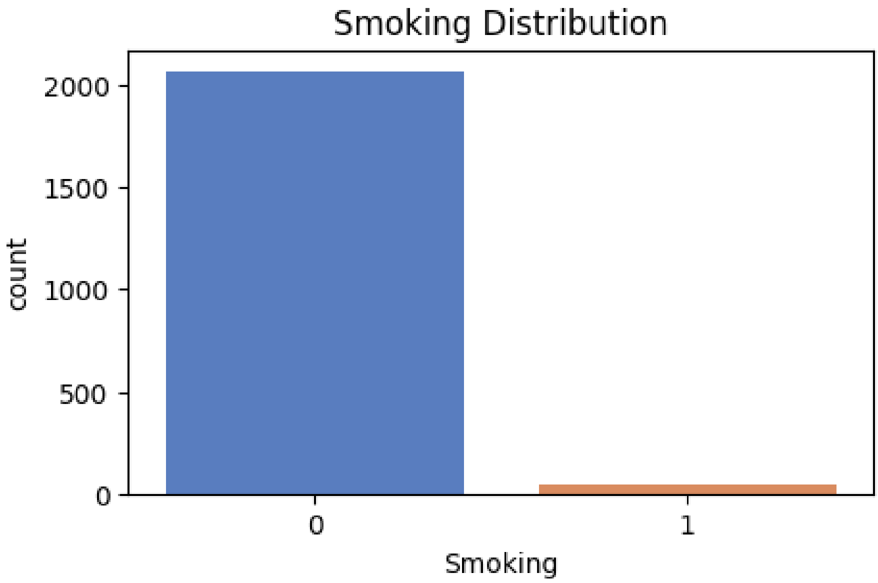


Figure 12: Consumption of food between meal Variable visualization

10. Smoking



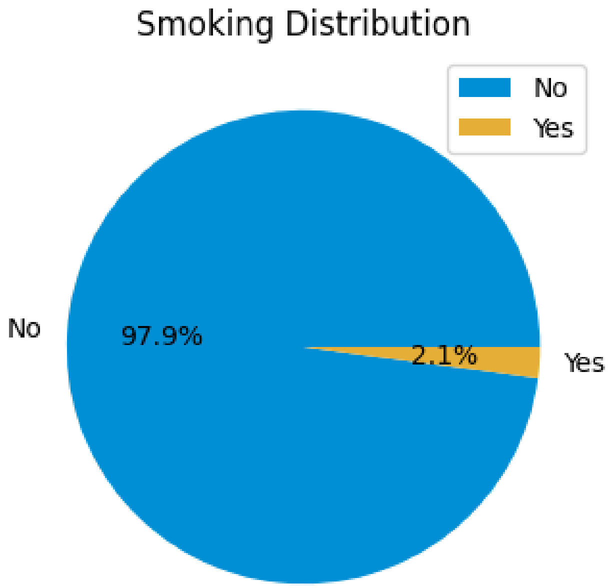
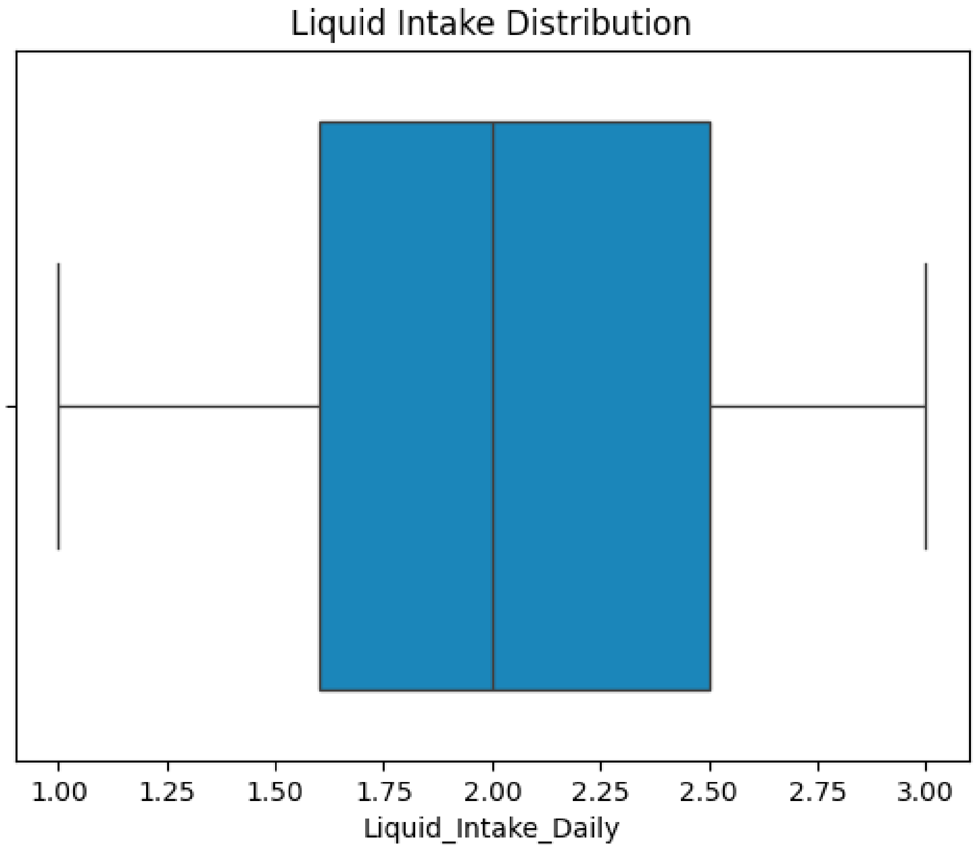


Figure 13: Smoking Variable visualization

11. Liquid intake



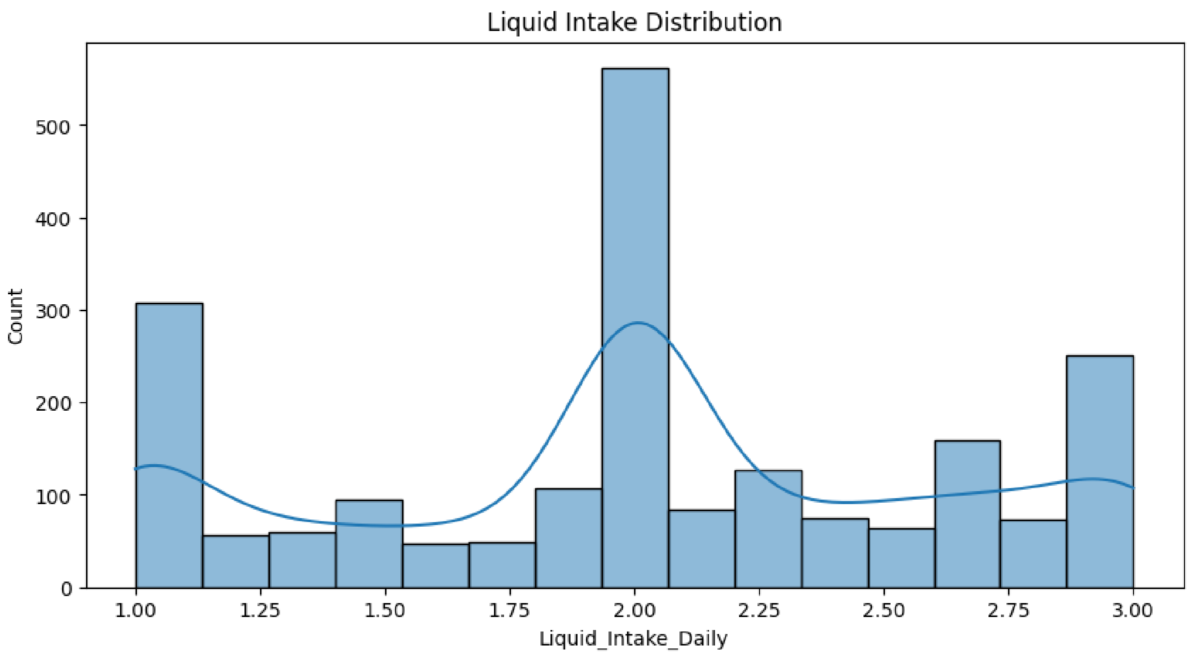
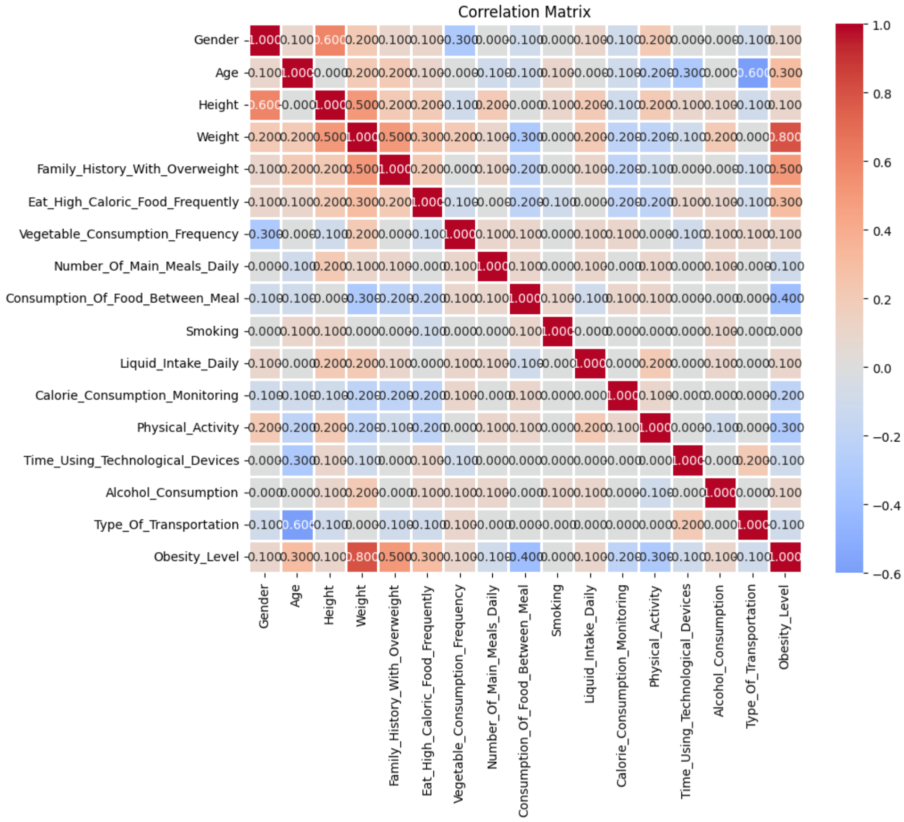


Figure 14: Liquid intake Variable visualization and distribution

- Skewness of 'Liquid_Intake_Daily': -0.1
- Kurtosis of 'Liquid_Intake_Daily': -0.88

2. Pearson Correlation analysis



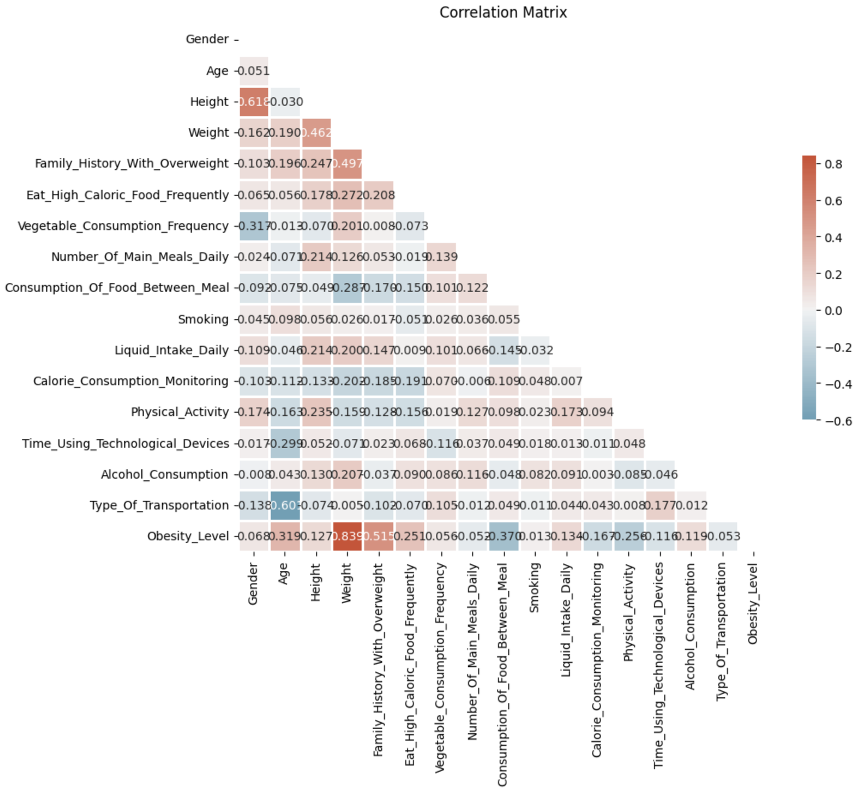


Figure 15: Correlation matrix heatmap

Both snippets are useful for visually and numerically exploring the relationships between different features in a dataset, which is an essential part of exploratory data analysis (EDA) before building predictive models.

The values in a correlation matrix depict the strength and direction of the linear relationship between pairs of variables. The values in the matrix are the Pearson correlation coefficients, which range from **-1** to **+1**. Here's what the different values represent:

Pearson Correlation Coefficient Values:

1. **+1**:
    - A perfect positive linear relationship.
    - As one variable increases, the other also increases in a perfectly linear fashion.
    - Example: Height and weight often have a positive correlation.
2. **-1**:
    - A perfect negative linear relationship.
    - As one variable increases, the other decreases in a perfectly linear fashion.
    - Example: The speed of a car and the time it takes to reach a destination are negatively correlated.
3. **0**:
    - No linear relationship between the variables.
    - The variables do not affect each other in a linear manner.
    - Example: The number of ice creams sold, and shoe size might have no correlation.

Interpretation of Correlation Values:

- **0.7 to 1.0 or -0.7 to -1.0**:
    - **Strong correlation** (positive or negative).
    - High values (close to 1 or -1) indicate a strong relationship between the variables.
- **0.3 to 0.7 or -0.3 to -0.7**:
    - **Moderate correlation**.
    - Moderate values suggest some relationships, but it's not as strong.
- **0.0 to 0.3 or -0.0 to -0.3**:
    - **Weak correlation**.
    - Low values suggest a weak or negligible relationship between the variables.