Statistical Data Analysis Report on Food Nutrition

Food & Nutrition Overview

`

The effective management of food intake and nutrition are both key to good health. Understanding good nutrition and paying attention to what you eat can help you maintain or improve your health.

What Is Good Nutrition?

Food and nutrition are the way that we get fuel, providing energy for our bodies. We need to replace nutrients in our bodies with a new supply every day. Water is an important component of nutrition. Fats, proteins, and carbohydrates are all required. Maintaining key vitamins and minerals are also important to maintaining good health. For pregnant women and adults over 50, vitamins such as vitamin D and minerals such as calcium and iron are important to consider when choosing foods to eat, as well as possible dietary supplements.

A healthy diet includes a lot of natural foods. A sizeable portion of a healthy diet should consist of fruits and vegetables, especially ones that are red, orange, or dark green. Whole grains, such as whole wheat and brown rice, should also play a part in your diet. For adults, dairy products should be non-fat or low-fat. Protein can consist of lean meat and poultry, seafood, eggs, beans, legumes, and soy products such as tofu, as well as unsalted seeds and nuts.
Good nutrition also involves avoiding certain kinds of foods. Sodium is used heavily in processed foods and is dangerous for people with high blood pressure. The USDA advises adults to consume less than 300 milligrams (mg) per day of cholesterol (found in meat and full-fat dairy products among others). Fried food, solid fats, and trans fats found in margarine and processed foods can be harmful to heart health. Refined grains (white flour, white rice) and refined sugar (table sugar, high fructose corn syrup) are also bad for long-term health, especially in people with diabetes. Alcohol can be dangerous to health in amounts more than one serving per day for a woman and two per day for a man.

There are many high-quality, free guidelines available for healthy eating plans that give more details on portion size, total calorie consumption, what to eat more of, and what to eat less of to get healthy and stay that way.

Nutritional Deficiencies:

Even if you are getting enough to eat, if you are not eating a balanced diet, you may still be at risk for certain nutritional deficiencies. Also, you may have nutritional deficiencies due to certain health or life conditions, such as pregnancy, or certain medications you may be taking, such as high blood pressure medications. People who have had intestinal diseases or had sections of intestines removed

due to disease or weight loss surgery also may be at risk for vitamin deficiencies. Alcoholics are also at high risk of having nutritional deficiencies.

One of the most common nutritional deficiencies is iron deficiency anemia. Your blood cells need iron in order to supply your body with oxygen, and if you don't have enough iron, your blood will not function properly. Other nutritional deficiencies that can affect your blood cells include low levels of vitamin B12, folate, or vitamin C.

Vitamin D deficiency may affect the health of your bones, making it difficult for you to absorb and use calcium (another mineral that you may not be getting enough of). Although you can get vitamin D by going out in the sun, many people with concerns about skin cancer may end up with low levels of vitamin D by not getting enough sun.

Problem Statement

Perform a descriptive statistical analysis on the dataset to make decisions about the food they consume on a daily basis. The goal of this analysis is to maintain good nutrition for a healthier diet.

Initial Analysis

Using Python, obtain the dataset in the form of a table and determine the number of records and characteristics included in the dataset.

Measures of Central Tendency

To determine the average amount of nutrients contained in the various types of food products, find the mean, median, and mode of the characteristics in the dataset.

The measures of Dispersion

To determine the range of the associated dataset, find the minimum and maximum value of the features. To determine the dispersion of data points feature-wise, find the variance and standard deviation of the chosen features.

Measures of Position:

Find the first, second, and third quartile of the selected features in the dataset.
They have taken the following dataset for this case study: nutrition_data.xlsx

The df table captures all names of different food items of the weight of 100 g with all the nutrition content present in corresponding food items.

The df table captures all names of different food items of the weight of 100 g with all the nutrition content present in corresponding food items.

name	Calories	Total Fat (g)	Cholesterol (mg)	Sugars (g)	Protein (g)
Cornstarch	381	0.1	0	0.00	0.26
Nuts, pecans	691	72	0	3.97	9.17
Cauliflower, raw	25	0.3	0	1.91	1.92
Vegetarian fillets	290	18.0	0	0.80	23.0
Mango nectar, canned	51	0.1	0	12.45	0.11
Crackers, rusk toast	407	7.2	78	0.00	13.50
Chicken, boiled	215	15.0	84.0	0	19.40

Table: nutrition_1.csv

Case Study Questions:

Q1. Load the CSV file into table format and find the shape of the data frame.

```
In [63]: import pandas as pd

In [64]: df = pd.read_csv("C:\\Users\\sharm\\Desktop\\Nutrition project\\nutrition_1.csv", index_col = 0)

In [65]: df.head()
```

Out[65]:

	Unnamed: 0	name	serving_size	calories	total_fat	saturated_fat	cholesterol	sodium	choline	folate	...	fat	saturated_fatty_acids	monounsaturated_
0	0	Cornstarch	100 g	381	0.1	NaN	0.0	9.0	0.4 mg	0.00 mcg	...	0.05 g	0.009	
1	1	Nuts, pecans	100 g	691	72.0	6.2	0.0	0.0	40.5 mg	22.00 mcg	...	71.97 g	6.180	
2	2	Eggplant, raw	100 g	25	0.2	NaN	0.0	2.0	6.9 mg	22.00 mcg	...	0.18 g	0.034	
3	3	Teff, uncooked	100 g	367	2.4	0.4	0.0	12.0	13.1 mg	0	...	2.38 g	0.449	
4	4	Sherbet, orange	100 g	144	2.0	1.2	1.0	46.0	7.7 mg	4.00 mcg	...	2.00 g	1.160	

5 rows × 77 columns

Q2. Remove the unnecessary columns named "Unnamed: 0", "Unnamed: 0.1", lucopene, and "Serving_Size"

Q3. Check for the Null and Duplicate values which are present in the dataset and remove them

```
In [112]: null_values = df.isnull().sum()
          print("null values in each column:")
          print(null_values)
```

```
null values in each column:
Unnamed: 0      0
name            0
serving_size    0
calories        0
total_fat       0
               ..
alcohol         0
ash             0
caffeine        0
theobromine     0
water           0
Length: 77, dtype: int64
```

```
In [113]: df.dropna(inplace=True)
          print("null values has been removed")
```

```
null values has been removed
```

In [114]:
```
duplicate_value = df.duplicated().sum()
print(f"\nduplicates value in each column: {duplicate_value}")
```

```
duplicates value in each column: 0
```

In [115]:
```
df.drop_duplicates(inplace=True)
print("duplicates has been removed")
```

```
duplicates has been removed
```

In [116]:
```
output_file_path = 'cleaned_file.csv'
df.to_csv(output_file_path, index=False)

print(f"\nThe cleaned file is saved as '{output_file_path}'.")
```

```
The cleaned file is saved as 'cleaned_file.csv'.
```

Q4. Find the mean of all numeric features.

In [74]:
```
def calculate_central_tendency(df):
    central_tendency = {}
    for column in df.columns:
        if df[column].dtype in ['int64', 'float64']:
            mean_value = df[column].mean()
            median_value = df[column].median()
            mode_value = df[column].mode().iloc[0] if not df[column].mode().empty else None

            central_tendency[column] = {
                'Mean': mean_value,
                'Median': median_value,
                'Mode': mode_value
            }
    return central_tendency

# Calculate central tendency for each column
central_tendency = calculate_central_tendency(df)

# Display the results
for column, values in central_tendency.items():
    print(f"\nColumn: {column}")
    print(f"Mean: {values['Mean']}")
    print(f"Median: {values['Median']}")
    print(f"Mode: {values['Mode']}")
```

Q5. Find the median for features like sodium, and folic_acid. Find the mean of features like fiber, and sugars.

In [118]:
```
columns_name = ["folic_acid", "sodium"]
median_values = df[columns_name].median()
print(f"\nmedian values are: {median_values}")
```

```
median values are: folic_acid      0.0
sodium        115.0
dtype: float64
```

In [119]:
```
features = ["fiber", "sugars"]
mean_features = df[features].mean()
print(f"\nMean of fiber and sugars are: {mean_features}")
```

```
Mean of fiber and sugars are: fiber     2.150201
sugars    6.575701
dtype: float64
```

Q6. Find the maximum and minimum values for the alcohol column.

```
In [120]: # Find the maximum and minimum values for the alcohol column.
          max_value = df["alcohol"].max()
          print(f"\nmaxiumum value of alcohol column: {max_value}")

          maxiumum value of alcohol column: 26.0

In [121]: min_value = df["alcohol"].min()
          print(f"\nminimum value of alcohol column: {min_value}")

          minimum value of alcohol column: 0.0
```

Q7. How many unique values are present in the feature, choline?

```
In [122]: # How many unique values are present in the feature, choline?
          unique_value = df['choline'].nunique()
          print(f"\nunique_number is: {unique_value}")

          unique_number is: 1183
```

Q8. What is the standard deviation for features, fiber, folic_acid, lucopene?

```
In [123]: # What is the standard deviation for features, fiber, folic_acid, lucopene?
          columns = ['fiber', 'folic_acid', 'lycopene']
          for column in columns:
              if column in df.columns:
                  std_deviation = df[column].std()
                  print(f"The standard deviation for the '{column}' column is: {std_deviation}")
              else:
                      print(f"The '{column}' column does not exist in the dataset.")

          The standard deviation for the 'fiber' column is: 4.4860643285973
          The standard deviation for the 'folic_acid' column is: 104.05929322348679
          The 'lycopene' column does not exist in the dataset.
```

Q9. Find the variance for total_fat to get how total_fat is varying among different food items.

```
In [124]: #  Find the variance for total_fat to get how total_fat is varying among different food items.

          if 'total_fat' in df.columns:
              fat = df['total_fat'].var()
              print(f"\n variance of total_fat is: {fat}")

           variance of total_fat is: 272.61513737223794
```

Q10. How to get a number of records, mean, standard deviation, min value, max values, and quartiles with 1 line of code?

```
# How to get a number of records, mean, standard deviation, min value, max values, and quartiles with 1 line of code?
summary_statistics = df[['fiber', 'folic_acid','total_fat']].describe()

# Print the summary statistics
print(summary_statistics)
```

```
             fiber   folic_acid    total_fat
count  7199.000000  7199.000000  7199.000000
mean      2.150201    20.615224    12.625281
std       4.486064   104.059293    16.511061
min       0.000000     0.000000     0.100000
25%       0.000000     0.000000     2.800000
50%       0.200000     0.000000     7.400000
75%       2.500000     0.000000    16.000000
max      79.000000  1611.000000   100.000000
```

Q11. Find the correlation between different features available in the dataset.

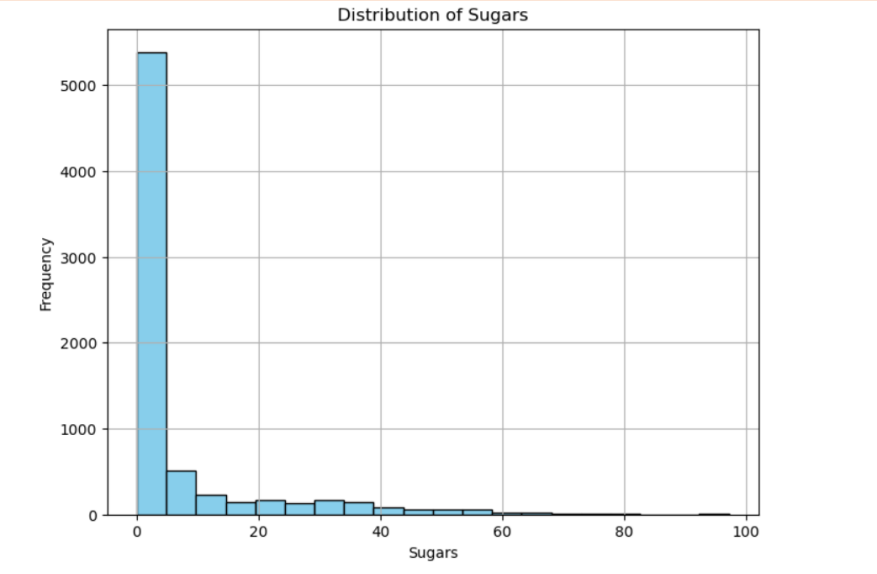
In [126]:
```
# Find the correlation between different fiber and folic acid available in the dataset.
corr_relation = df[['fiber','folic_acid']].corr()
print(corr_relation)
```

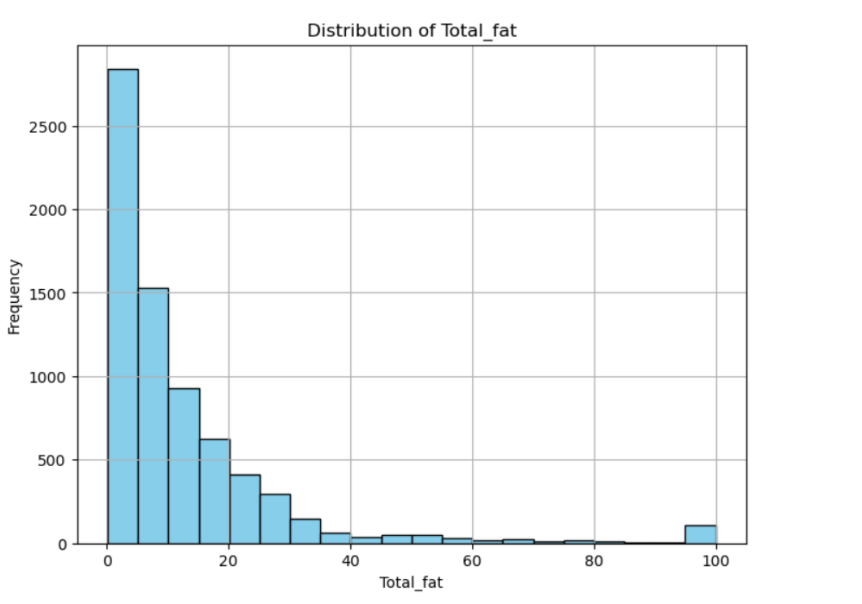
```
               fiber  folic_acid
fiber       1.000000    0.166918
folic_acid  0.166918    1.000000
```

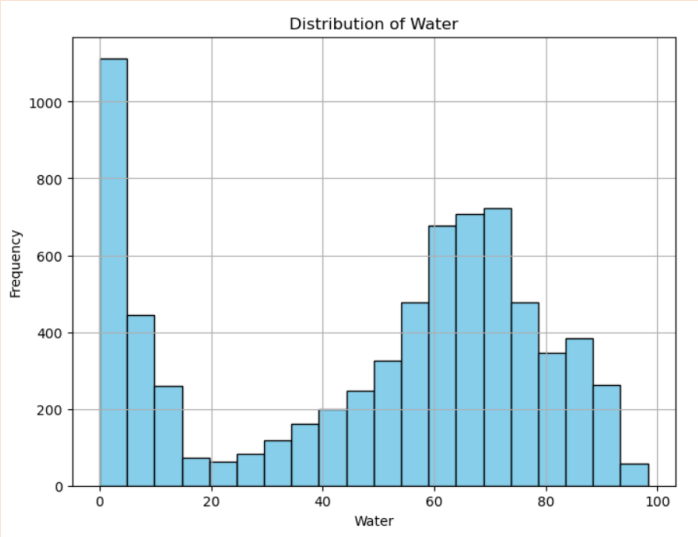
Q12. Find the distribution of columns, sugars, total_fat, water, and protein by plotting the histogram.

In [127]:
```
# Find the distribution of columns, sugars, total_fat, water, and protein by plotting the histogram.
import matplotlib.pyplot as plt
```

In [128]:
```
column_plot = ['sugars', 'total_fat', 'water']
for column in column_plot:
    plt.figure(figsize=(8,6))
    plt.hist(df[column], bins=20, color='skyblue', edgecolor='black')
    plt.xlabel(column.capitalize())
    plt.ylabel('Frequency')
    plt.title(f"Distribution of {column.capitalize()}")
    plt.grid(True)
    plt.show()
```







Q13. Find the skewness and kurtosis for all the numeric features available in the dataset.

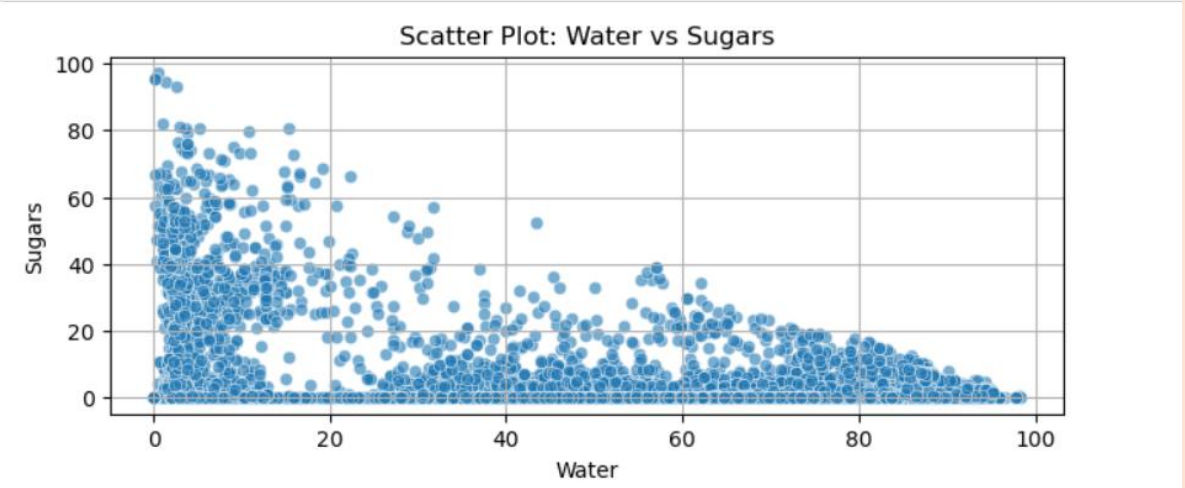
In [132]:
```
import pandas as pd
```

In [134]:
```
# Find the skewness and kurtosis for all the numeric features available in the dataset.

numeric_columns = df.select_dtypes(include=['number'])
skewness = numeric_columns.apply(lambda x: x.skew())
kurtosis = numeric_columns.apply(lambda x: x.kurtosis())

skew_kurtosis_df = pd.DataFrame({'Kurtosis': kurtosis, 'Skewness': skewness})
print(skew_kurtosis_df)
```

```
                        Kurtosis   Skewness
vitamin_e               288.275028  12.772937
calcium                  89.369794   7.161020
irom                     66.520216   6.649349
protein                   2.269122   0.958588
carbohydrate             -0.563484   0.969658
fiber                    48.637295   5.384535
sugars                    7.346885   2.669446
fructose                319.625396  15.098983
galactose              4099.733286  57.310743
glucose                 753.764602  23.037167
lactose                 640.242799  24.299880
maltose                 150.260533  10.686607
sucrose                 138.274342  10.586978
saturated_fatty_acids    63.194254   6.342268
alcohol                2285.740999  46.130768
ash                     235.326384  11.372485
caffeine               2634.802106  48.629623
theobromine             826.444283  25.903850
water                    -1.186832  -0.477171
```

Q14. Plot the scatter plot for features of water vs sugars to get a visual representation of the relationship between them.

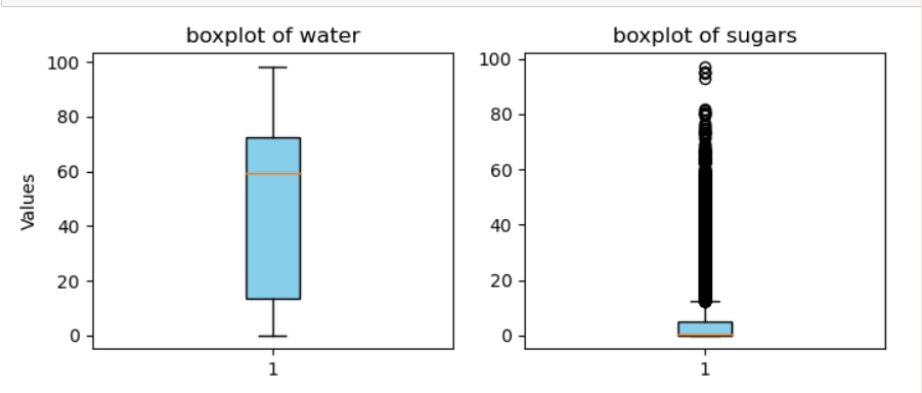
In [145]:
```
# Plot the scatter plot for features of water vs sugars to get a visual representation of the relationship between them.
if 'water' in df.columns and 'sugars' in df.columns:
    # Plot the scatter plot
    plt.figure(figsize=(10, 6))
    plt.scatter(df['water'], df['sugars'], alpha=0.6, edgecolors='w', linewidth=0.5)
    plt.xlabel('Water')
    plt.ylabel('Sugars')
    plt.title('Scatter Plot: Water vs Sugars')
    plt.grid(True)
    plt.show()
else:
    print("The dataset does not contain 'water' and/or 'sugars' columns.")
```



Q15. Plot the box plots for columns, water, and sugars to find if these features have outliers.

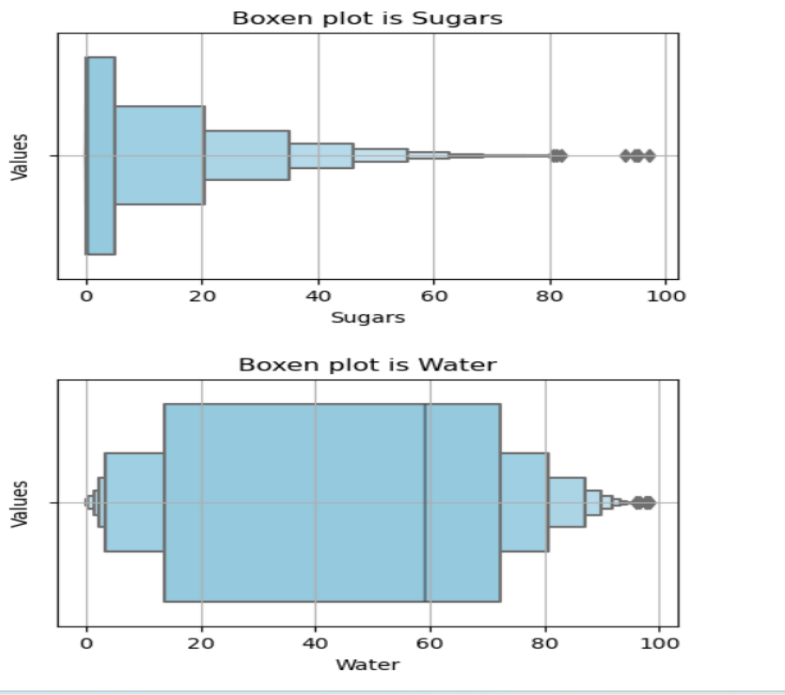
In [157]:
```
import matplotlib.pyplot as plt
```

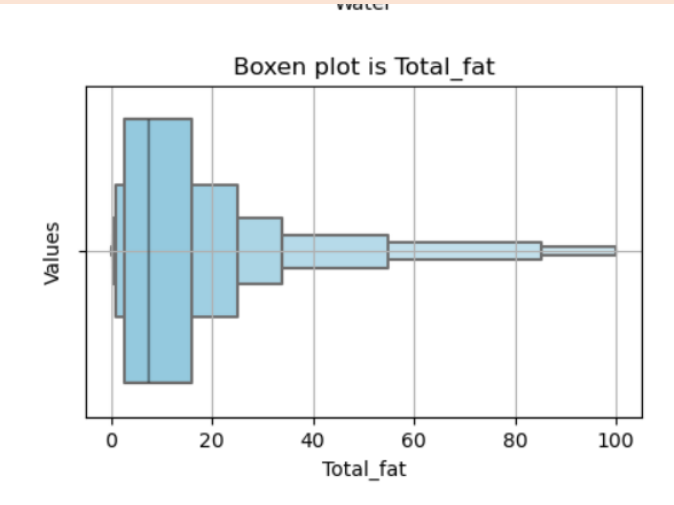
In [159]:
```
# Plot the box plots for columns, water, and sugars to find if these features have outliers.
if 'water' in df.columns and 'sugars' in df.columns:
    plt.figure(figsize=(12,6))
    plt.subplot(1,2,1)
    plt.boxplot(df['water'].dropna(), vert=True, patch_artist=True, boxprops=dict(facecolor='skyblue'))
    plt.ylabel('Values')
    plt.title('boxplot of water')

    plt.subplot(1,2,2)
    plt.boxplot(df['sugars'].dropna(), vert=True, patch_artist=True, boxprops=dict(facecolor='skyblue'))
    plt.title('boxplot of sugars')
    plt.show()
```



Q16. Plot the boxen plots to get details views of values present in the features, sugars, water, and total_fat

In [214]:
```
columns_to_plot = ['sugars', 'water','total_fat']
for column in columns_to_plot:
    if column not in df.columns:
        print("The dataset does not contain '{column}' column.")

    else:
            plt.figure(figsize=(5,3))
            sns.boxenplot(data=df, x=column, color='skyblue')
            plt.xlabel(column.capitalize())
            plt.ylabel('Values')
            plt.title(f'Boxen plot is {column.capitalize()}')
            plt.grid(True)
            plt.show()
```





Q17. Plot the barplot for the variance of features like vitamin_b12, vitamin_e, fiber, fructose, glucose, lactose, sucrose, alcohol, and ash, to get a detailed view of the dataset.

In [184]:
```
# Plot the barplot for the variance of features like vitamin_b12, vitamin_e, fiber, fructose, glucose, lactose, sucrose, alcohol,
columns_to_analyze = ['vitamin_b12', 'vitamin_e', 'fiber', 'fructose', 'glucose', 'lactose', 'sucrose', 'alcohol', 'ash']

for column in columns_to_analyze:
    if column not in df.columns:
        print(f"The dataset does not contain '{column}' column.")
        continue

# Calculate variance for the specified columns
variance_values = df[columns_to_analyze].var()

# Create a DataFrame for the variance values
variance_df = pd.DataFrame(variance_values, columns=['Variance']).reset_index()
variance_df.rename(columns={'index': 'Feature'}, inplace=True)

# Plot the bar plot
plt.figure(figsize=(12, 6))
sns.barplot(x='Feature', y='Variance', data=variance_df, palette='viridis')
plt.title('Variance of Specified Features')
plt.xlabel('Feature')
plt.ylabel('Variance')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```