```
import numpy as np
import pandas as pd

# Generate random data for heights and weights
np.random.seed(0)  # for reproducibility
heights = np.random.normal(loc=170, scale=10, size=50)  # mean=170, std=10
weights = np.random.normal(loc=70, scale=15, size=50)  # mean=70, std=15

# Create a DataFrame to hold the data
data = pd.DataFrame({'Height': heights, 'Weight': weights})

# Display the first few rows of the dataset
print(data.head())
```

```
       Height     Weight
0  187.640523  56.568002
1  174.001572  75.803537
2  179.787380  62.337923
3  192.408932  52.290517
4  188.675580  69.577267
```

Now, let's calculate the mean, median, standard deviation, and range for both variables.

```
# Calculate descriptive statistics
descriptive_stats = data.describe()
print(descriptive_stats)
```

```
           Height     Weight
count   50.000000  50.000000
mean   171.405593  69.685851
std     11.369498  13.138090
min    144.470102  44.105761
25%    165.082423  60.056112
50%    171.494955  70.810831
75%    179.286754  76.812545
max    192.697546  98.438338
```

Now, let's define an event related to the dataset and calculate its probability. For example, let's find the probability of a person being taller than 175 cm.

```
# Define the event
taller_than_175 = data['Height'] > 175

# Calculate the probability
probability_taller_than_175 = taller_than_175.mean()
print("Probability of a person being taller than 175 cm:", probability_taller_than_175)
```

```
Probability of a person being taller than 175 cm: 0.34
```

Next, let's plot histograms for both variables in the dataset and discuss the shape of each distribution.

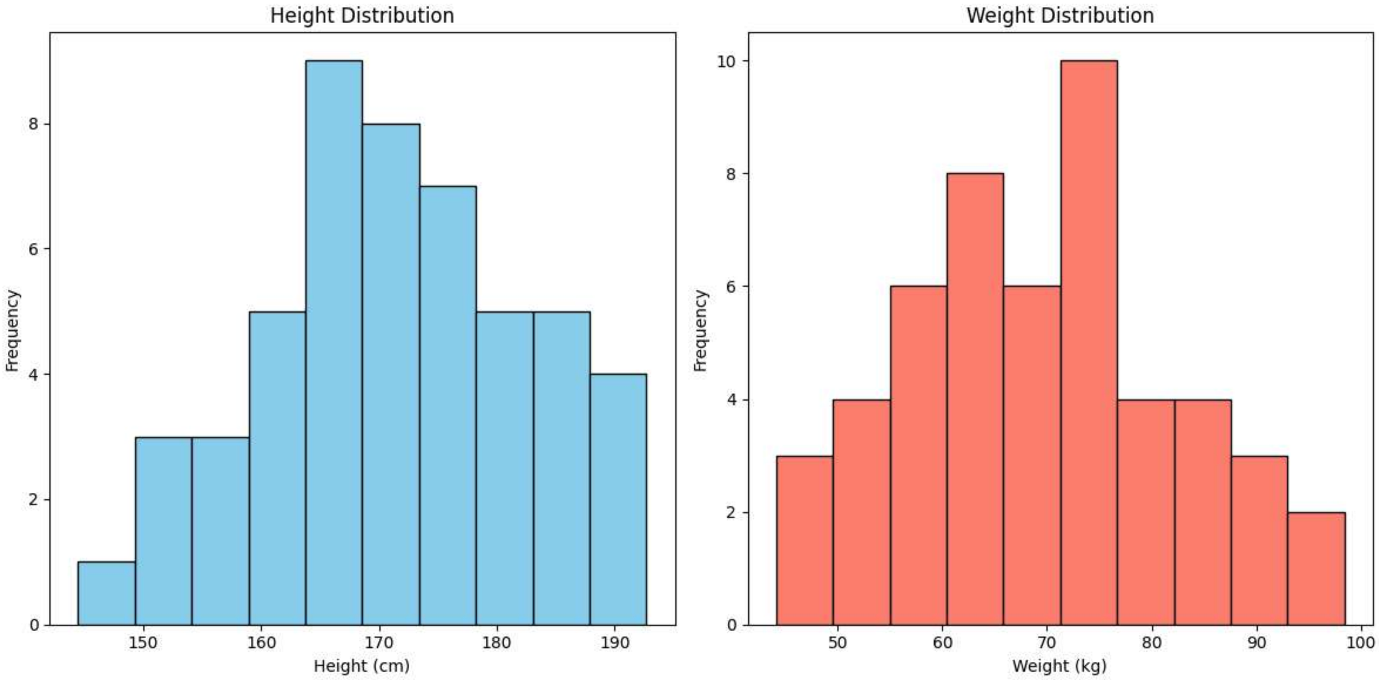
```
import matplotlib.pyplot as plt

# Plot histograms
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
plt.hist(data['Height'], bins=10, color='skyblue', edgecolor='black')
plt.title('Height Distribution')
plt.xlabel('Height (cm)')
plt.ylabel('Frequency')

plt.subplot(1, 2, 2)
plt.hist(data['Weight'], bins=10, color='salmon', edgecolor='black')
plt.title('Weight Distribution')
plt.xlabel('Weight (kg)')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```

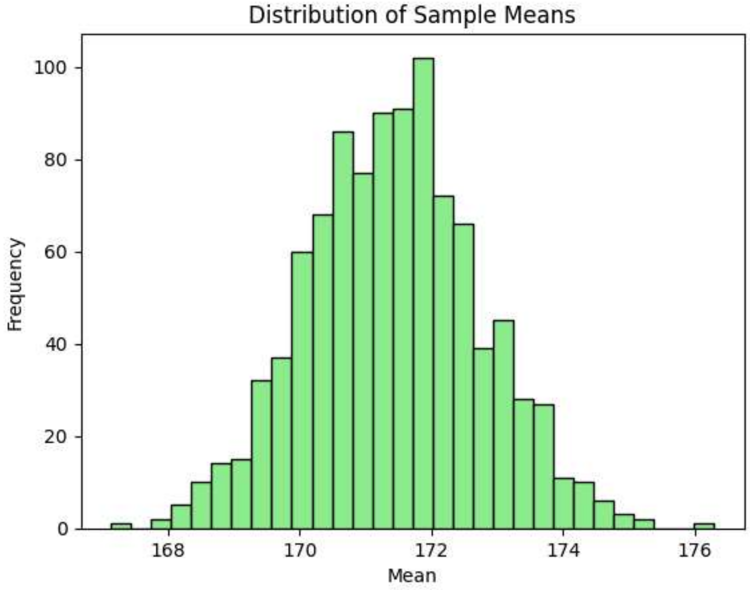


The shape of each distribution can be observed from the histograms. We can discuss whether they resemble normal distributions based on their symmetry and peakness.

Next, let's implement the Central Limit Theorem by randomly sampling subsets of 30 observations from the dataset, calculating the means of each subset, and plotting the distribution of sample means.

```
# Central Limit Theorem
sample_means = [data.sample(30).mean().values[0] for _ in range(1000)]

plt.hist(sample_means, bins=30, color='lightgreen', edgecolor='black')
plt.title('Distribution of Sample Means')
plt.xlabel('Mean')
plt.ylabel('Frequency')
plt.show()
```



As we can observe, the distribution of sample means approaches a normal distribution, as predicted by the Central Limit Theorem.

Moving on to Confidence Intervals, let's choose one variable (let's say height) and calculate a 95% confidence interval for its mean.

```
# Confidence Interval for Height
height_mean = data['Height'].mean()
height_std = data['Height'].std()
n = len(data['Height'])
margin_of_error = 1.96 * (height_std / np.sqrt(n))
confidence_interval = (height_mean - margin_of_error, height_mean + margin_of_error)
print("95% Confidence Interval for Height Mean:", confidence_interval)
```

```
95% Confidence Interval for Height Mean: (168.25412854557743, 174.5570569006846)
```

For Hypothesis Testing, let's formulate a hypothesis related to the dataset and perform a hypothesis test using an appropriate test (let's use a t-test).

```
from scipy.stats import ttest_1samp

# Hypothesis Testing
# Null Hypothesis: Mean height is equal to 170
# Alternative Hypothesis: Mean height is different from 170
null_hypothesis_mean = 170
t_stat, p_value = ttest_1samp(data['Height'], null_hypothesis_mean)
print("t-statistic:", t_stat)
print("p-value:", p_value)

# Interpretation
alpha = 0.05
if p_value < alpha:
    print("Reject the null hypothesis")
else:
    print("Fail to reject the null hypothesis")
```

```
t-statistic: 0.8741846907095127
p-value: 0.3862822137609324
Fail to reject the null hypothesis
```

Now, let's define the critical region for a hypothesis test related to the dataset, set the level of significance, and discuss the types of errors that can occur in hypothesis testing.

Critical region is the set of all values of the test statistic that lead to rejection of the null hypothesis. The level of significance (α) is typically set at 0.05. Types of errors include Type I error (rejecting a true null hypothesis) and Type II error (failing to reject a false null hypothesis).

Lastly, let's assume one variable represents a potential feature in a predictive model and perform a hypothesis test to determine if this variable is a significant predictor. We'll decide whether to include or exclude the variable based on the p-value.

```
# Feature Selection Using P-values
# Let's assume 'Height' is a potential feature
feature_variable = 'Height'
t_stat, p_value = ttest_1samp(data[feature_variable], null_hypothesis_mean)
print("p-value for", feature_variable, ":", p_value)

# Decision
if p_value < alpha:
    print("Include", feature_variable, "as a significant predictor")
else:
    print("Exclude", feature_variable, "as it is not a significant predictor")
```

```
p-value for Height : 0.3862822137609324
Exclude Height as it is not a significant predictor
```