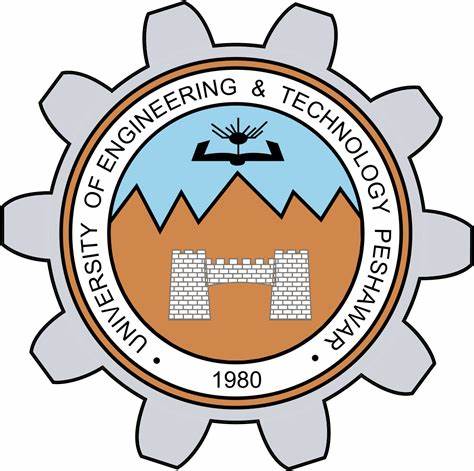
Lab report 04

**Central Tendence and Measure of Dispersion**

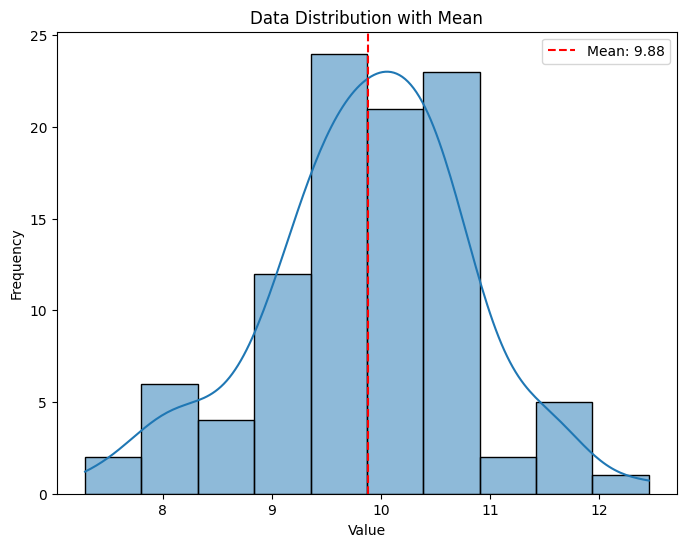
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**



**CSE-422L-Data Analytics Lab**

**Department of Computer System Engineering**

**University of Engineering and Technology Peshawar**

**Lab 04: Central Tendence and Measure of Dispersion**

**Task 1:**

* Generate random data (sample size=100). Hint: [random.normal(loc=10,\_\_=1,100)]
* Calculate the mean of the data.
* Plot the data distribution using a histogram or boxplot, and mark the mean on the plot. Plt.bar.

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

**import** **seaborn** **as** **sns**

**from** **scipy** **import** stats

data = np.random.normal(loc=10, scale=1, size=100)

mean = np.mean(data)

plt.figure(figsize=(8, 6))

sns.histplot(data, kde=**True**, bins=10)

plt.axvline(mean, color='red', linestyle='--', label=f'Mean: {mean:.2f}')

plt.legend()

plt.title('Data Distribution with Mean')

plt.xlabel('Value')

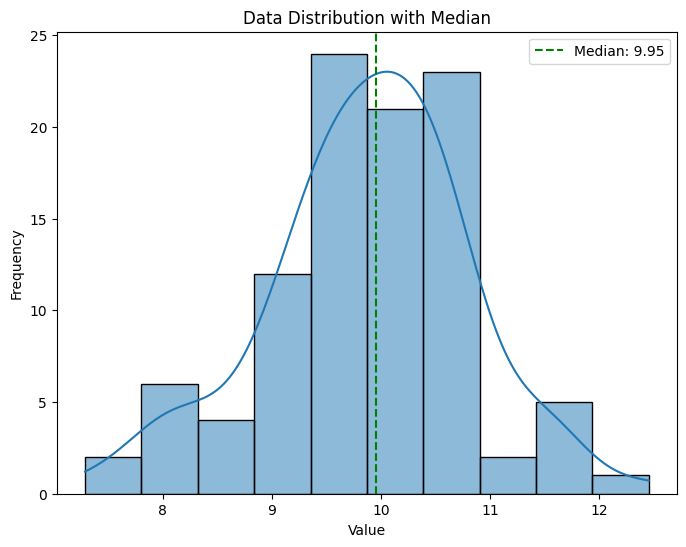
plt.ylabel('Frequency')

plt.show()

np.float64(9.884450140520334)

:01: Data Distribution with Mean

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

****

**Task 2:**

* Use the same dataset
* Calculate the median.
* Use a histogram to visualize the data and mark the median on the plot.

median = np.median(data)

plt.figure(figsize=(8, 6))

sns.histplot(data, kde=**True**, bins=10)

plt.axvline(median, color='green', linestyle='--', label=f'Median: {median:.2f}')

plt.legend()

plt.title('Data Distribution with Median')

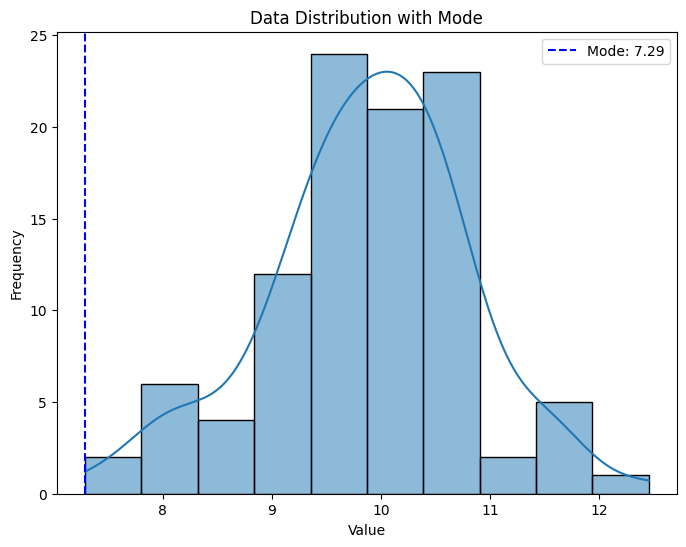
plt.xlabel('Value')

plt.ylabel('Frequency')

plt.show()

np.float64(9.953295061153305)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Task 3:**

* Use the same dataset
* Calculate the mode(s).
* Use a histogram to visualize the data, highlighting the mode(s).

mode\_result = stats.mode(data, keepdims=**True**)

mode = mode\_result.mode **if** isinstance(mode\_result.mode, np.ndarray) **else** mode\_result.mode[0]

mode\_value = mode[0] **if** isinstance(mode, np.ndarray) **else** mode

plt.figure(figsize=(8, 6))

sns.histplot(data, kde=**True**, bins=10)

plt.axvline(mode\_value, color='blue', linestyle='--', label=f'Mode: {mode\_value:.2f}')

plt.legend()

plt.title('Data Distribution with Mode')

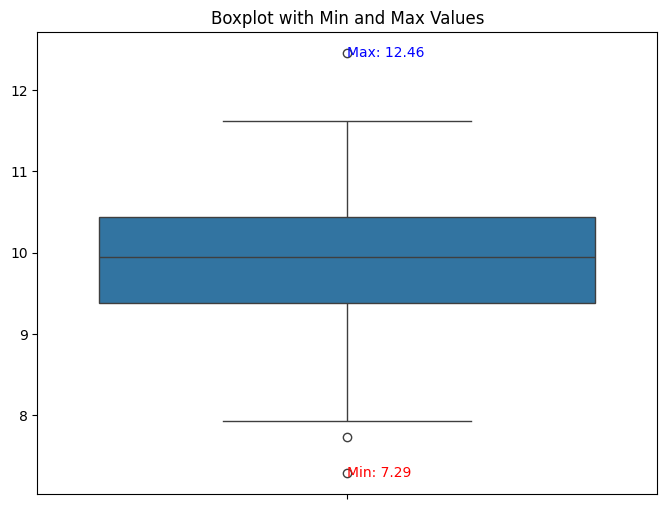
plt.xlabel('Value')

plt.ylabel('Frequency')

plt.show()

np.float64(7.605858443526187)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Task 4:**

* Calculate the minimum and maximum values to find the range.
* Visualize using a boxplot, marking min and max values.

min\_val = np.min(data)max\_val = np.max(data)plt.figure(figsize=(8, 6))sns.boxplot(data)plt.text(0, min\_val, f'Min: {min\_val:.2f}', verticalalignment='center', color='red')plt.text(0, max\_val, f'Max: {max\_val:.2f}', verticalalignment='center', color='blue')plt.title('Boxplot with Min and Max Values')plt.show() min\_val = np.min(data)

max\_val = np.max(data)

plt.figure(figsize=(8, 6))

sns.boxplot(data)

plt.text(0, min\_val, f'Min: {min\_val:.2f}', verticalalignment='center', color='red')

plt.text(0, max\_val, f'Max: {max\_val:.2f}', verticalalignment='center', color='blue')

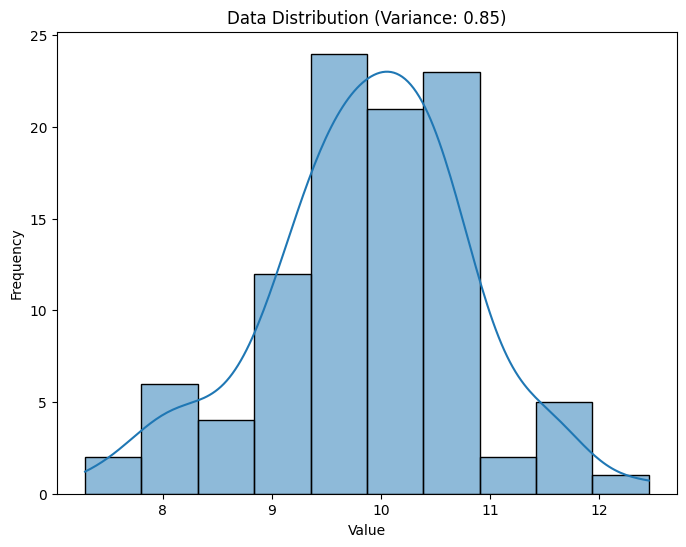
plt.title('Boxplot with Min and Max Values')

plt.show()

(np.float64(7.605858443526187), np.float64(11.941712524352312))

*#min#max*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**Task 5:**

* Calculate variance.
* Plot the data distribution and display variance as text.

variance = np.var(data)

plt.figure(figsize=(8, 6))

sns.histplot(data, kde=**True**, bins=10)

plt.title(f'Data Distribution (Variance: {variance:.2f})')

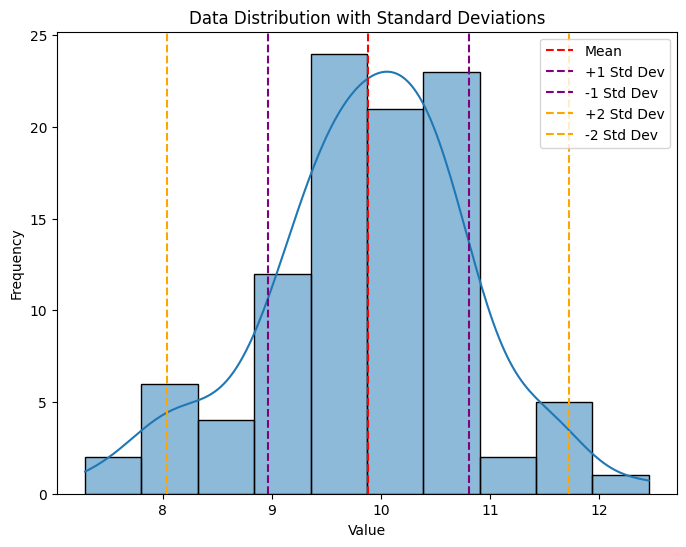
plt.xlabel('Value')

plt.ylabel('Frequency')

plt.show()

np.float64(0.8487579341293062)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Task 6:**

* Calculate standard deviation.
* Visualize data distribution using histogram and annotate ±1, ±2 standard deviations and also show the %age of data lies in each std.

std\_dev = np.std(data)

plt.figure(figsize=(8, 6))

sns.histplot(data, kde=**True**, bins=10)

plt.axvline(mean, color='red', linestyle='--', label='Mean')

plt.axvline(mean + std\_dev, color='purple', linestyle='--', label='+1 Std Dev')

plt.axvline(mean - std\_dev, color='purple', linestyle='--', label='-1 Std Dev')

plt.axvline(mean + 2 \* std\_dev, color='orange', linestyle='--', label='+2 Std Dev')

plt.axvline(mean - 2 \* std\_dev, color='orange', linestyle='--', label='-2 Std Dev')

plt.legend()

plt.title('Data Distribution with Standard Deviations')

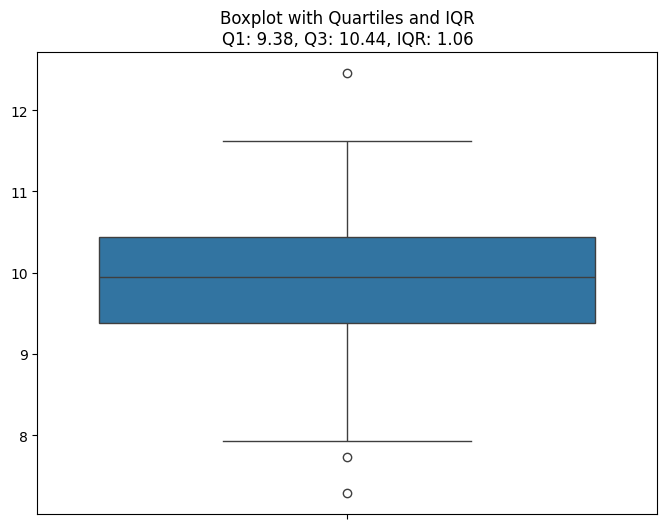
plt.xlabel('Value')

plt.ylabel('Frequency')

plt.show()

np.float64(0.9212805946774881)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Task 7:**

* Compute Q1, Q3, and IQR.
* Plot a boxplot showing quartiles and IQR range.

Q1 = np.percentile(data, 25)

Q3 = np.percentile(data, 75)

IQR = Q3 - Q1

plt.figure(figsize=(8, 6))

sns.boxplot(data)

plt.title(f'Boxplot with Quartiles and IQR**\n**Q1: {Q1:.2f}, Q3: {Q3:.2f}, IQR: {IQR:.2f}')

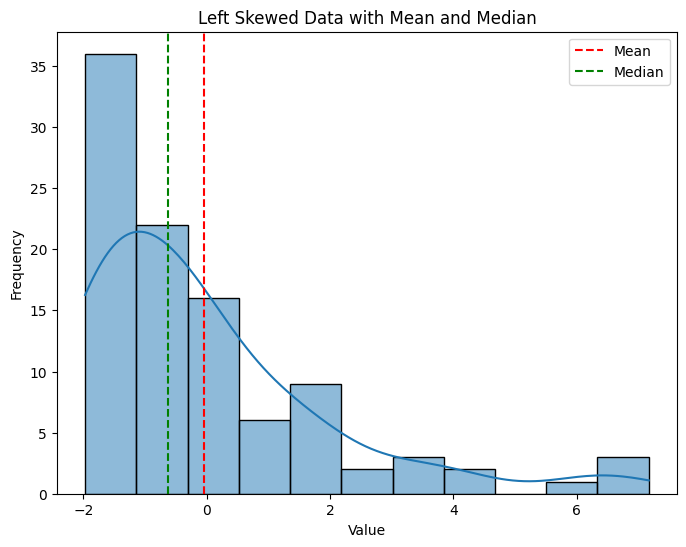
plt.show()

Q1 --> (np.float64(9.379385158713593),

Q3 --> np.float64(10.441176987794115),

IQR --> np.float64(1.0617918290805228))

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**Task 8:**

* Generate a skewed dataset (for both (a) left and (b)right skewed data)
* Calculate mean and median.
* Plot histogram, marking mean,mode and median.

left\_skewed\_data = np.random.exponential(scale=2, size=100) - 2

right\_skewed\_data = np.random.exponential(scale=2, size=100)

left\_mean = np.mean(left\_skewed\_data)

left\_median = np.median(left\_skewed\_data)

plt.figure(figsize=(8, 6))

sns.histplot(left\_skewed\_data, kde=**True**)

plt.axvline(left\_mean, color='red', linestyle='--', label='Mean')

plt.axvline(left\_median, color='green', linestyle='--', label='Median')

plt.legend()

plt.title('Left Skewed Data with Mean and Median')

plt.xlabel('Value')

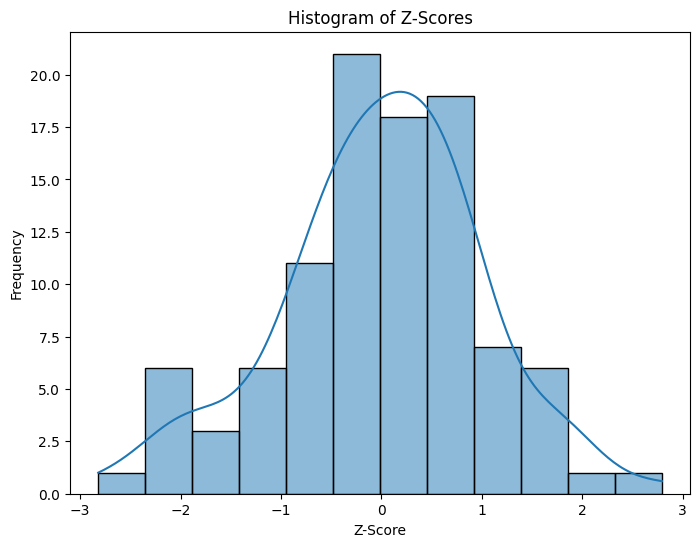
plt.ylabel('Frequency')

plt.show()

(np.float64(-0.03440479865526957), --->left\_mean

np.float64(-0.6257495666102129)) --->left\_median

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



**Task 9:**

* Calculate Z-scores.
* Plot histogram of Z-scores, showing standard normal distribution as reference.

z\_scores = stats.zscore(data)

plt.figure(figsize=(8, 6))

sns.histplot(z\_scores, kde=**True**)

plt.title('Histogram of Z-Scores')

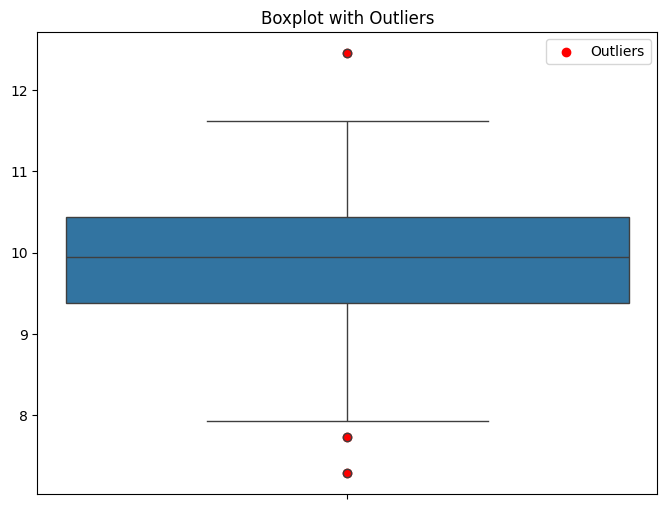
plt.xlabel('Z-Score')

plt.ylabel('Frequency')

plt.show()

Array([ 1.47557185, -0.45897887, 0.16881003, 0.64403198, 1.78909131, 0.07658948, -0.82048796, -1.2279773 , -1.90548818, -0.42658071, -0.05792327, 0.56821028, 0.67307282, -1.01873892, -0.09351565, -0.5458107 , -0.36643674, 0.25965985, -0.40163348, 2.79150046, 0.20829293, -1.17065629, .........................]) ----> Z-Score Values

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

****

**Task 10:**

* Calculate Q1, Q3, and IQR, then identify outliers.
* Plot data with outliers marked, using a scatter plot or boxplot.

Q1 = np.percentile(data, 25)

Q3 = np.percentile(data, 75)

IQR = Q3 - Q1

lower\_bound = Q1 - 1.5 \* IQR

upper\_bound = Q3 + 1.5 \* IQR

outliers = data[(data < lower\_bound) | (data > upper\_bound)]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_