2. **Visualization**:
    - Enhancing insights into datasets via graphical representation.
3. **Advanced Metrics**:
    - Insights into dataset distribution using skewness and kurtosis.

Let me know if you'd like further clarification or explanations for specific parts!

expalin word by wrod

Let's break down the code **word by word** and explain each line and concept in detail. I'll start with the code from **AIML_1** and then cover **AIML_2**.

AIML_1: Statistical Measures

Import Statements

```
python


  import pandas as pd
  import numpy as np
  import matplotlib.pyplot as plt
```

- **`import` allows the use of external libraries.**
    - **`pandas`**: Library for data manipulation and analysis.
    - **`numpy` (short for Numerical Python)**: Provides functions for numerical computations.
    - **`matplotlib.pyplot`**: Used for creating visualizations.

Seeding and Generating Random Data

```
python


  np.random.seed(5)
```

- **`np.random.seed(5)`**: Fixes the random number generator for reproducibility. This ensures the same random numbers are generated every time the code is run.

```
python


  x = np.random.randint(10, 70, 10)
  y = np.random.randint(20, 40, 10)
```

- `np.random.randint(a, b, n)`: Generates `n` random integers between `a` (inclusive) and `b` (exclusive).
  - `x`: Random integers from 10 to 69, with 10 elements.
  - `y`: Random integers from 20 to 39, with 10 elements.

```
python


  x.sort()
  y.sort()
```

- `.sort()`: Sorts the arrays `x` and `y` in ascending order for easier interpretation.

Defining Statistical Functions

1. **Mean Function**

```
python


  def calc_mean(dataset):
      return dataset.sum() / len(dataset)
```

- `def`: Keyword to define a function.
- `calc_mean`: Function name for calculating mean.
- `dataset`: Input parameter representing a dataset (array-like structure).
- `dataset.sum()`: Adds all values in the dataset.
- `len(dataset)`: Returns the number of elements in the dataset.
- **Formula for Mean**:

$$\text{Mean} = \frac{\text{Sum of all values}}{\text{Number of values}}$$

---

2. **Variance Function**

```
python


  def calc_variance(dataset, mean):
      squared_diff = np.square(dataset - mean)
      return calc_mean(squared_diff)
```

- `calc_variance`: Function for variance calculation.
- `dataset - mean`: Subtracts the mean from each value in the dataset.
- `np.square()`: Squares the differences.
- `calc_mean(squared_diff)`: Finds the average of squared differences.
- **Formula for Variance**:

$$\text{Variance} = \frac{\sum (X_i - \text{Mean})^2}{n}$$

3. **Standard Deviation**

```python
def calc_SD(variance):
    return np.sqrt(variance)
```

- `calc_SD`: Calculates the standard deviation.
- `np.sqrt()`: Finds the square root of the variance.
- **Formula for Standard Deviation**:

$$\text{SD} = \sqrt{\text{Variance}}$$

---

4. **Covariance**

```python
def calc_covariance(dataset1, dataset2):
    mean1 = calc_mean(dataset1)
    mean2 = calc_mean(dataset2)
    return np.sum(np.multiply(dataset1 - mean1, dataset2 - mean2)) / len(dataset1)
```

- `calc_covariance`: Function for finding covariance between two datasets.
- `np.multiply()`: Multiplies corresponding elements of two arrays.
- **Formula for Covariance**:

$$\text{Covariance} = \frac{\sum (X_i - \text{Mean}_X)(Y_i - \text{Mean}_Y)}{n}$$

---

5. **Correlation**

```python
def calc_correlation(dataset1, dataset2):
    mean1 = calc_mean(dataset1)
    mean2 = calc_mean(dataset2)
    num = np.sum(np.multiply(dataset1 - mean1, dataset2 - mean2))
    de = np.sqrt(np.sum(np.square(dataset1 - mean1)) * np.sum(np.square(dataset2 - mean2)))
    return num / de
```

- `calc_correlation`: Computes the correlation coefficient between two datasets.
- `num`: Numerator (covariance).
- `de`: Denominator (product of standard deviations).
- **Formula for Correlation**:

$$\text{Correlation} = \frac{\text{Covariance}}{\text{Product of SDs}}$$

---

6. **Standard Error**

```python
def calc_SE(dataset, sd):
    return sd / np.sqrt(len(dataset))
```

- `calc_SE`: Computes the standard error.
- **Formula for Standard Error**:

$$\text{SE} = \frac{\text{SD}}{\sqrt{n}}$$

Graphical Representation

```python
plt.plot(x, "mo:", label="dataset 1")
plt.plot(y, "go:", label="dataset 2")
plt.legend(loc="upper left")
plt.axhline(mean, color="b", marker="o", linestyle=":", label="Mean")
```

- `plt.plot()`: Plots the datasets `x` and `y`.
  - `mo:`: Magenta circles connected by lines for `x`.
  - `go:`: Green circles connected by lines for `y`.
- `plt.legend()`: Adds a legend at the top-left corner.
- `plt.axhline()`: Draws a horizontal line at the mean value.

AIML_2: Normal Distribution, Skewness, and Kurtosis

Reading Data

```python
df = pd.read_csv("score.csv")
x = df['math score']
x = x.to_numpy()[:15]
x.sort()
```

- `pd.read_csv()`: Reads a CSV file.
- `df['math score']`: Extracts the "math score" column.
- `x.to_numpy()[:15]`: Converts the column to a NumPy array and takes the first 15 elements.
- `x.sort()`: Sorts the scores.

Normal Distribution Plot