Data Analysis with Python

ISE4132 : AI Application System

AMIN AL (아민알)
Integrated System Engineering (ISE)
010-6853-6648
alaminanik@inha.ac.kr

Understanding the Dataset

Brief description of the dataset (source, type of data: numerical, categorical, text, images, etc.)

Number of samples and features (rows × columns)

Purpose of the dataset (prediction, classification, regression, etc.)

Feature Types: Numerical features (continuous or discrete), Categorical features (nominal, ordinal), Datetime features

Descriptive Analytics

Descriptive statistics provide a summary of data features

Common statistics:
Minimum (Min) – smallest value
Maximum (Max) – largest value
Mean – average value
Median – middle value
Standard Deviation (SD) – spread or variability
Range – difference between Max and Min

Minimum (Min), Maximum (Max)

Definition: The smallest and largest value in the dataset.

Purpose:
Understand lower and higher bounds.
Identify outliers or extreme low values.

Example:
If dataset = [5, 8, 12, 20], Min = 5, Max = 20

Mean and Median

Mean: Sum of all values divided by total number of values.

Median: Middle value when data is sorted.

Formula:

$$\text{Mean} = \frac{\sum x_i}{n}$$

Mean Example:
Dataset = [5, 8, 12, 20], Mean = (5+8+12+20)/4 = 11.25

Mean and Median

Arrange the data in ascending order: $x_1 \leq x_2 \leq x_3 \leq \ldots \leq x_n$

If $n$is odd (number of data points is odd): $\text{Median} = x_{\frac{n+1}{2}}$

Example:
Dataset = [3, 5, 8] → Sorted: [3, 5, 8]
n=3, Median = $\mathrm{x}_{(3+1)/2} = \mathrm{x}_2 = 5$

If n is even (number of data points is even): $\text{Median} = \dfrac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2}$

Example:
Dataset = [3, 5, 8, 10] → Sorted: [3, 5, 8, 10]
n =4, so Median = $\frac{x_{4/2}+x_{4/2+1}}{2} = \frac{x_2+x_3}{2} = \frac{5+8}{2} = 6.5$

Standard Deviation (SD)

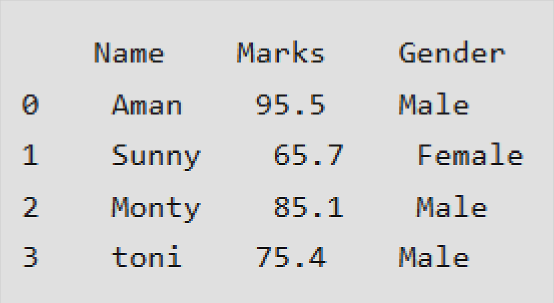
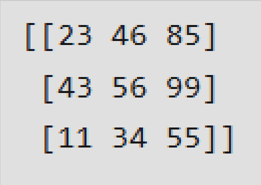
Definition: Measures how much values deviate from the mean. It is a statistical measure of how spread out data points are from their average (mean).

Formula: $$SD = \sqrt{\frac{\sum(x_i - \bar{x})^2}{n}}$$

Purpose: Indicates variability or spread of data

Example: Dataset = [5, 8, 12, 20], SD ≈ 6.05

Pandas and NumPy

Pandas	NumPy
When we have to work on Tabular data, we prefer the pandas module.	When we have to work on Numerical data, we prefer the NumPy module.
The powerful tools of pandas are DataFrame and Series.	Whereas the powerful tool of NumPy is Arrays.
	

```
    Name     Marks     Gender
0     Aman     95.5     Male
1     Sunny     65.7     Female
2     Monty     85.1     Male
3     toni     75.4     Male
```

```
[[23 46 85]
 [43 56 99]
 [11 34 55]]
```

Code

```
import numpy as np
import pandas as pd

data = [5, 8, 12, 12, 20]

series = pd.Series(data)

minimum = series.min()
print(f"Minimum: {minimum}")

maximum = series.max()
print(f"Maximum: {maximum}")

mean = series.mean()
print(f"Mean: {mean}")

median = series.median()
print(f"Median: {median}")

std_dev = series.std()
print(f"Standard Deviation: {std_dev}")

data_range = maximum - minimum
print(f"Range: {data_range}")
```

Data Distribution

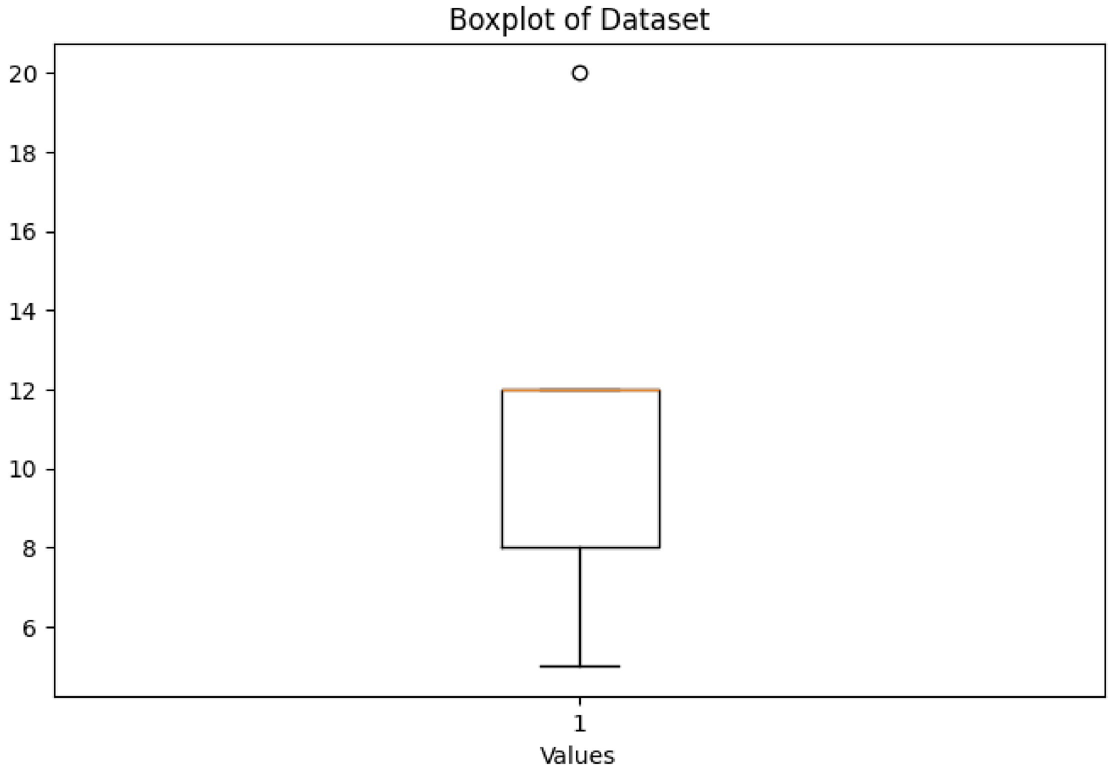
Definition: How data values are spread across possible ranges.

Why important?
- Detect skewness (left/right skewed)
- Identify normal vs. abnormal patterns
- Helps decide preprocessing steps (e.g., normalization, transformation)

Example:
Histogram → shows frequency of values
Line Chart → Shows the probability distribution of a single variable.
Boxplot → highlights median, quartiles, and outliers (*an outlier is a data point that differs significantly from other observations*)

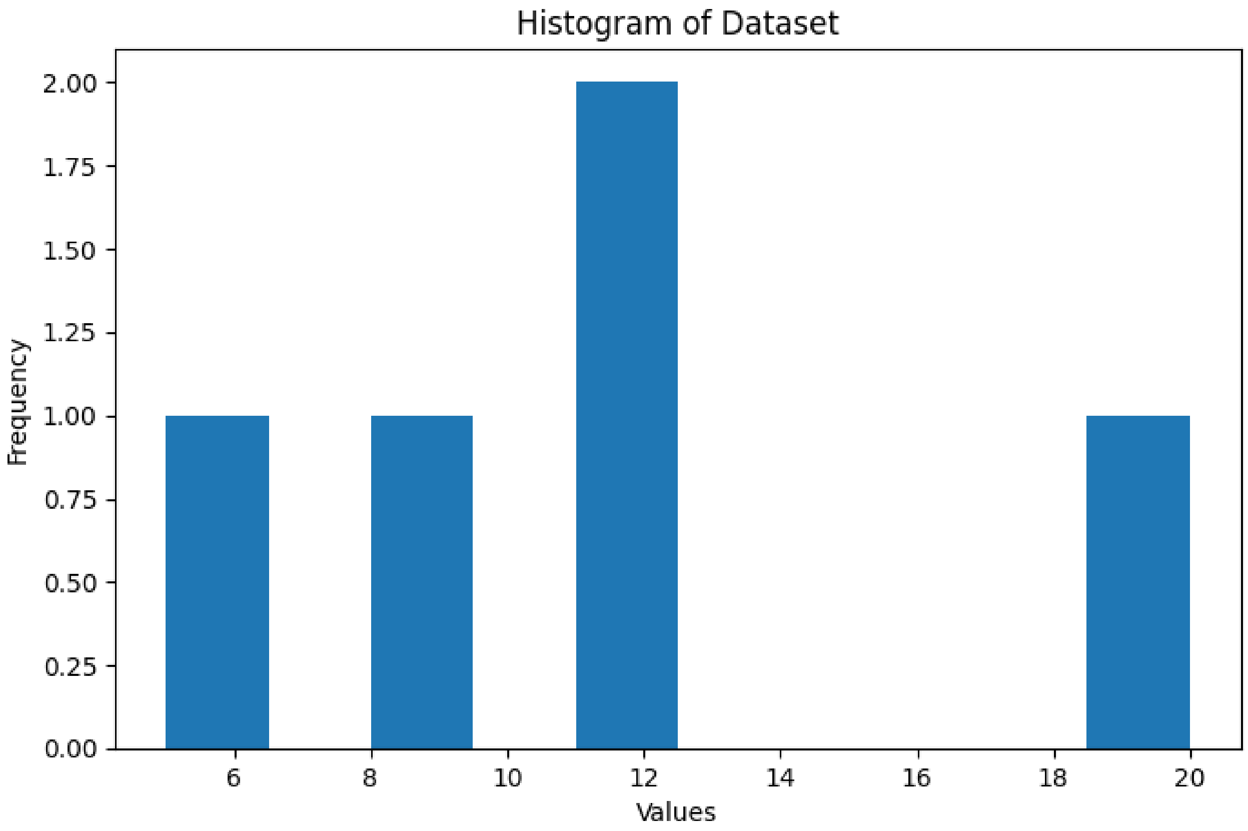
Boxplot

data = [5, 8, 12, 12, 20]



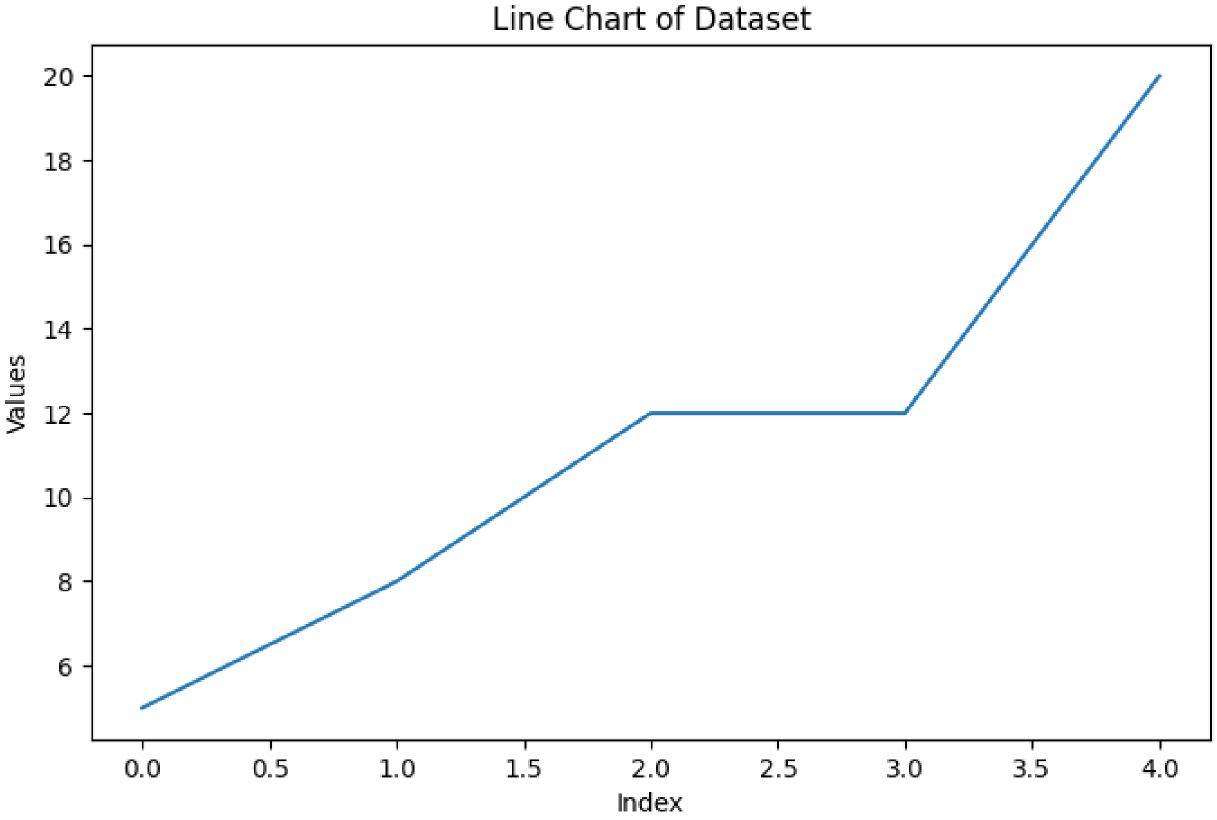
Histogram

data = [5, 8, 12, 12, 20]



Line Chart

data = [5, 8, 12, 12, 20]



Code

```
import matplotlib.pyplot as plt

# Boxplot
plt.figure(figsize=(8,5))
plt.boxplot(data)
plt.title("Boxplot of Dataset")
plt.xlabel("Values")
plt.show()

# Histogram
plt.figure(figsize=(8,5))
plt.hist(data, bins=5)
plt.title("Histogram of Dataset")
plt.xlabel("Values")
plt.ylabel("Frequency")
plt.show()

# Line chart
plt.figure(figsize=(8,5))
plt.plot(data)
plt.title("Line Chart of Dataset")
plt.xlabel("Index")
plt.ylabel("Values")
plt.show()
```

matplotlib.pyplot is a module within the Matplotlib library in Python. It provides a state-based interface for creating visualizations.

```
# Boxplot
plt.figure(figsize=(8,5))
plt.boxplot(data, vert=False)
plt.title("Boxplot of Dataset")
plt.xlabel("Values")
plt.show()

# Histogram
plt.figure(figsize=(8,5))
plt.hist(data, bins=5, color='skyblue', edgecolor='black')
plt.title("Histogram of Dataset")
plt.xlabel("Values")
plt.ylabel("Frequency")
plt.show()

# Line chart
plt.figure(figsize=(8,5))
plt.plot(data, marker='o', linestyle='-', color='green')
plt.title("Line Chart of Dataset")
plt.xlabel("Index")
plt.ylabel("Values")
plt.show()
```

Data Correlation

Definition: A statistical measure that shows the relationship between two variables.

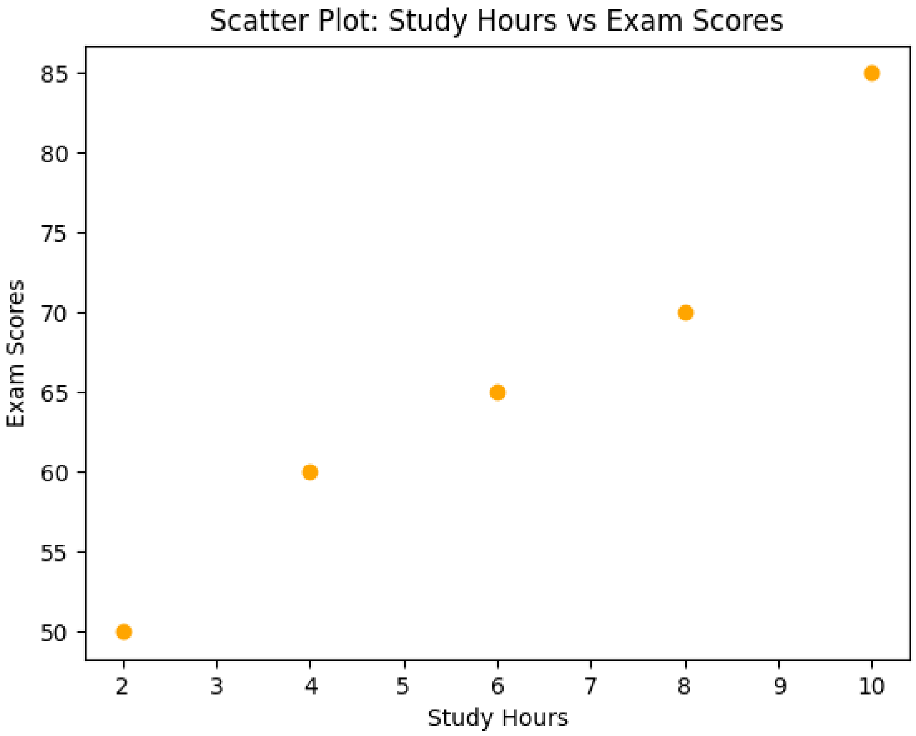
Range:
+1 → Strong positive relationship
0 → No relationship
–1 → Strong negative relationship

Why important?
Detect multicollinearity (when features are too similar)
Identify most relevant features for ML
Reduce redundancy in the dataset

Example:
Correlation matrix (numeric values)
Heatmap (color-coded for easy interpretation)
Line Chart → See if variables move together (positive correlation) or in opposite directions (negative correlation).
Example: "Height & Weight → strong positive correlation"

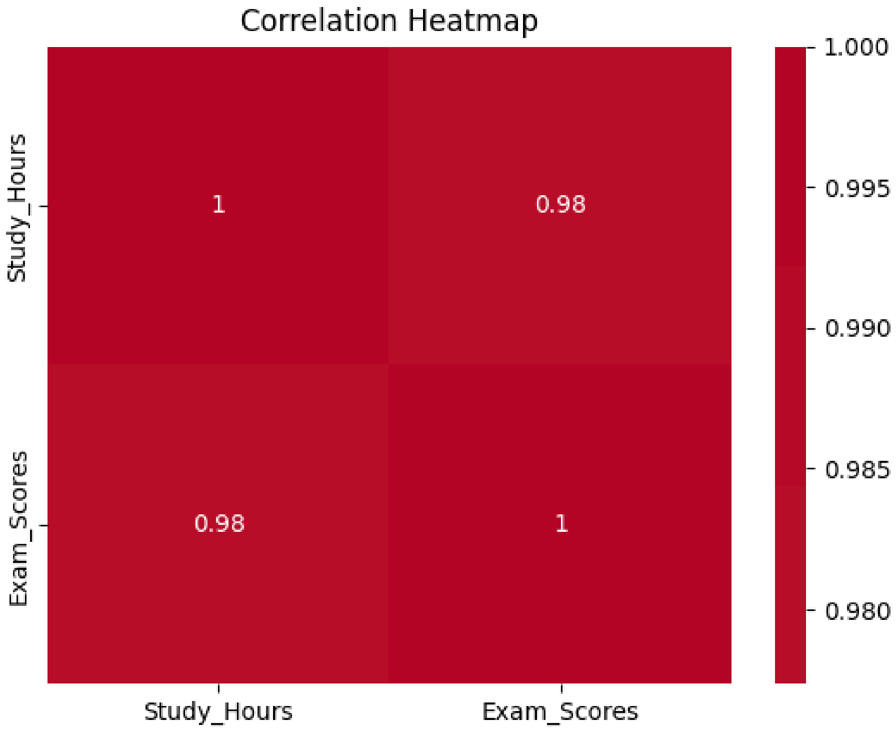
Scatter Plot

Study_Hours = [2, 4, 6, 8, 10]
Exam_Scores = [50, 60, 65, 70, 85]



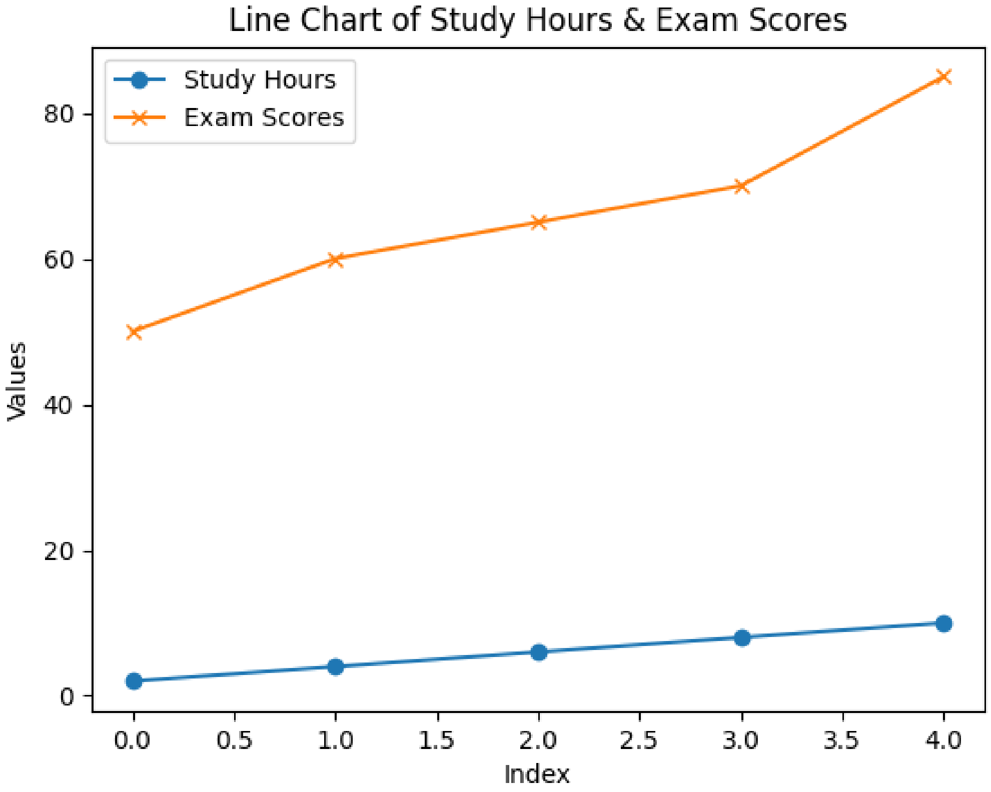
Correlation Heatmap

Study_Hours = [2, 4, 6, 8, 10]
Exam_Scores = [50, 60, 65, 70, 85]



Line Chart

Study_Hours = [2, 4, 6, 8, 10]
Exam_Scores = [50, 60, 65, 70, 85]



Code

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

data = pd.DataFrame({
    "Study_Hours": [2, 4, 6, 8, 10],
    "Exam_Scores": [50, 60, 65, 70, 85]
})

# Scatter Plot (Correlation)
plt.scatter(data["Study_Hours"], data["Exam_Scores"],
color="orange")
plt.title("Scatter Plot: Study Hours vs Exam Scores")
plt.xlabel("Study Hours")
plt.ylabel("Exam Scores")
plt.show()
```

```
# Correlation Heatmap
sns.heatmap(data.corr(), annot=True, cmap="coolwarm",
center=0)
plt.title("Correlation Heatmap")
plt.show()

# Line Chart (Correlation over sequence)
plt.plot(data["Study_Hours"], label="Study Hours", marker="o")
plt.plot(data["Exam_Scores"], label="Exam Scores", marker="x")
plt.title("Line Chart of Study Hours & Exam Scores")
plt.xlabel("Index")
plt.ylabel("Values")
plt.legend()
plt.show()
```

Handle Duplicates and Missing Values

For Numeric Columns

Drop rows/columns
If only a few rows are missing, drop them.
If an entire column has too many missing values, drop the column.

Fill with statistical values
Mean: Good if data is normally distributed (bell-shaped).
Median: Better if data has outliers.
Mode: Works if numeric values are discrete (e.g., bathroom count).

Predict missing values (ML Imputation)
Use machine learning models (like regression or KNN) to predict missing values based on other columns.

Handle Duplicates and Missing Values

For Categorical Columns

Drop rows/columns
If only a few rows are missing, drop them.
If an entire column has too many missing values, drop the column.

Mode imputation: Fill missing categories with the most frequent value.

Predict missing values (Classification): Train a model to predict missing category based on other features (similar to ML imputation).

Code

```
import pandas as pd

file_path = "House_Rent_Dataset.csv”
df = pd.read_csv(file_path)

print("Initial shape:", df.shape)
print("\nMissing values:\n", df.isnull().sum())
```

```
df = df.drop_duplicates()

num_cols =
df.select_dtypes(include=['int64','float64']).c
olumns
df[num_cols] =
df[num_cols].fillna(df[num_cols].median())

cat_cols =
df.select_dtypes(include=['object']).columns
for col in cat_cols:
 df[col] = df[col].fillna(df[col].mode()[0])

print("Missing values after cleaning:\n",
df.isnull().sum())
```

Class Activity 1

Dataset: [3, 7, 9, 15, 21, 21, 30]

1. Convert the dataset into a pandas Series.

2. Calculate **minimum, maximum, mean, median, standard deviation, and range**.

3. Compare mean and median. What does this tell you about the dataset's distribution?

Class Activity 2

Dataset: [4, 7, 7, 10, 12, 15, 18]

1. Create a **boxplot** and identify any outliers.

2. Create a **histogram** with **3 bins**.

3. Plot a **line chart** of the dataset.

4. Change the histogram bins to **5** and observe how the visualization changes.

Class Activity 3

Plot a scatter plot of Hours_Sleep vs Productivity_Score.
Calculate the correlation coefficient between the two variables.
Create a line chart for both variables on the same plot.

Hours_Sleep	Productivity_Score
4	50
5	55
6	60
7	65
8	80

Class Activity 4

Check for missing values in the dataset.
Fill missing numeric values with median.
Fill missing categorical values with mode.
Verify that all missing values have been handled.

Name	Age	Grade	Score
Alice	20	A	90
Bob		B	85
Carol	19		78
Dave	21	B	
Eve	22	A	88