

Name- Aarya Thorat  
D15A - 63

## Experiment – 0

### 1. Dataset Source

The dataset used for this experiment is a **synthetic student performance dataset** commonly used for demonstrating data preprocessing and exploratory data analysis techniques in machine learning.

#### **Dataset Source:**

[student\\_performance.csv](#)

### 2. Dataset Description

The dataset contains academic performance records of students and is used to analyze relationships between study habits, attendance, and examination scores.

#### **Dataset Characteristics**

- **Number of Records:** ~100–500 students (varies by version)
- **Type:** Structured, tabular data
- **Domain:** Education / Academic Analytics

#### **Features (Independent Variables)**

Feature Name	Description
Hours_Studied	Number of hours spent studying
Attendance	Attendance percentage of the student
Assignment_Score	Score obtained in assignments
Midterm_Score	Score obtained in midterm examination

#### **Target / Analysis Variable**

- Midterm\_Score (used for statistical analysis and visualization)

## Derived Categorical Feature

- Performance – classified based on **Attendance**:
  - High ( $\geq 85\%$ )
  - Medium (70–84%)
  - Low (< 70%)

## 3. Mathematical Formulation of the Algorithm

This experiment does **not use a predictive machine learning algorithm**.

Instead, it focuses on **statistical analysis and data normalization**, which are foundational steps in ML pipelines.

## Statistical Measures

- **Mean**

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i$$

- **Median**

The middle value of the sorted dataset.

- **Standard Deviation**

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}$$

## Min-Max Normalization

Used to scale values between 0 and 1:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

This technique is widely used in machine learning preprocessing to ensure uniform feature scaling.

## 4. Algorithm Limitations

Since this experiment relies on **Exploratory Data Analysis (EDA)** techniques rather than predictive models, the following limitations apply:

- No predictions or classifications are performed
- Cannot be used to infer causality
- Visual interpretations may vary depending on dataset size
- Sensitive to outliers in normalization and correlation analysis
- Not suitable for automated decision-making without a trained ML model

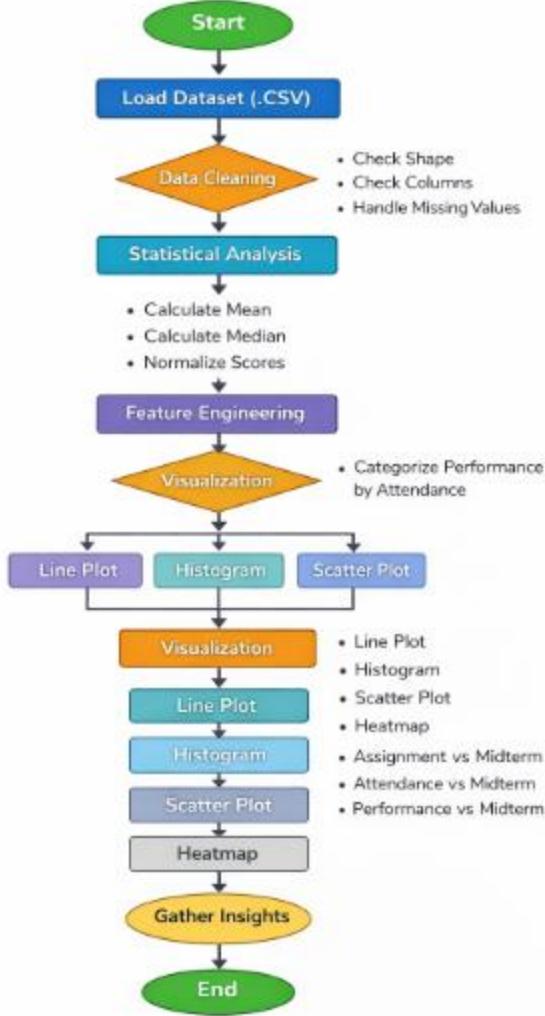
EDA is **descriptive**, not **prescriptive or predictive**.

## **5. Methodology / Workflow**

### **Step-by-Step Workflow**

- 1. Dataset Loading**
  - Loaded CSV file using Pandas
- 2. Data Inspection**
  - Checked shape, columns, and missing values
- 3. NumPy Operations**
  - Converted Midterm\_Score into NumPy array
  - Calculated mean, median, standard deviation
  - Applied Min-Max normalization
- 4. Feature Engineering**
  - Created Performance category based on attendance
- 5. Data Visualization**
  - Line plot: Assignment Score vs Midterm Score
  - Histogram: Distribution of Midterm Scores
  - Scatter plot: Attendance vs Midterm Score
  - Heatmap: Correlation between numerical features
  - Boxplot: Performance category vs Midterm Score

### ***Workflow Diagram***



## 6. Performance Analysis

Since no predictive model is trained, **traditional ML metrics** such as accuracy, precision, recall, or RMSE are **not applicable**.

### Analytical Observations

- Strong positive correlation observed between:
  - Assignment\_Score and Midterm\_Score
  - Attendance and academic performance
- Students with **High attendance** tend to score higher in midterm exams
- Normalized scores enable fair comparison across students
- Heatmap reveals inter-feature dependencies useful for future model selection

This analysis helps determine **feature importance** for downstream machine learning tasks.

## **7. Hyperparameter Tuning**

Hyperparameter tuning is **not applicable** in this experiment because:

- No machine learning model is implemented
- No training or optimization process is involved

However, this experiment serves as a **preparatory step** for ML pipelines where:

- Feature scaling parameters
- Model hyperparameters
- Feature selection thresholds

would be tuned in subsequent experiments.

## **Conclusion**

This experiment successfully demonstrated the use of **NumPy, Pandas, Matplotlib, and Seaborn** for data handling, normalization, and exploratory data analysis. By modifying the parameters and analysis focus, deeper insights into student academic behavior were obtained. The results emphasize the importance of EDA as a foundational step before applying machine learning algorithms.

## **Exercise 1 –**

```
import numpy as np

df = pd.read_csv('/content/student_performance.csv')

final_scores = df['Final_Score'].to_numpy()

print(final_scores)

mean_score = np.mean(final_scores)
median_score = np.median(final_scores)
std_score = np.std(final_scores)

print("Mean:", mean_score)
print("Median:", median_score)
print("Standard Deviation:", std_score)

min_score = np.min(final_scores)
max_score = np.max(final_scores)

normalized_scores = (final_scores - min_score) / (max_score - min_score)

normalized_scores
```

*Output -*

```
[52 57 60 64 68 71 74 77 79 83 63 70 75 56 69 73 80 58 72 78]
Mean: 68.95
Median: 70.5
Standard Deviation: 8.71478628538876
array([0.          , 0.16129032, 0.25806452, 0.38709677, 0.51612903,
       0.61290323, 0.70967742, 0.80645161, 0.87096774, 1.          ,
       0.35483871, 0.58064516, 0.74193548, 0.12903226, 0.5483871 ,
       0.67741935, 0.90322581, 0.19354839, 0.64516129, 0.83870968])
```

**Exercise 2 –**

```

import pandas as pd
df = pd.read_csv('/content/student_performance.csv')
df.head()
print("Shape:", df.shape)

print("Columns:", df.columns)

print("\nMissing Values:\n", df.isnull().sum())
def performance_label(score):
    if score >= 75:
        return "Excellent"
    elif score >= 60:
        return "Good"
    else:
        return "Average"

df['Performance'] = df['Final_Score'].apply(performance_label)

df.head()

```

*Output -*

```

Shape: (20, 5)
Columns: Index(['Hours_Studied', 'Attendance', 'Assignment_Score', 'Midterm_Score',
       'Final_Score'],
       dtype='object')

```

Missing Values:

```

Hours_Studied      0
Attendance         0
Assignment_Score   0
Midterm_Score      0
Final_Score        0
dtype: int64

```

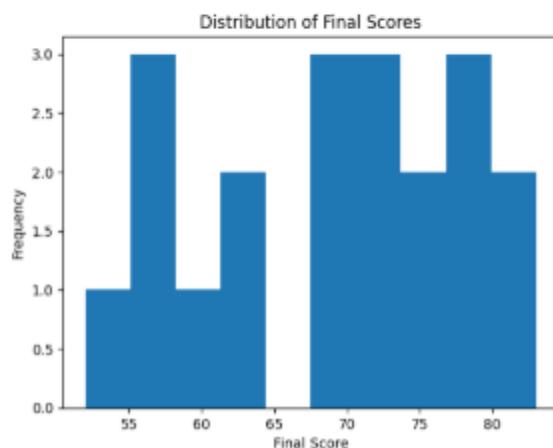
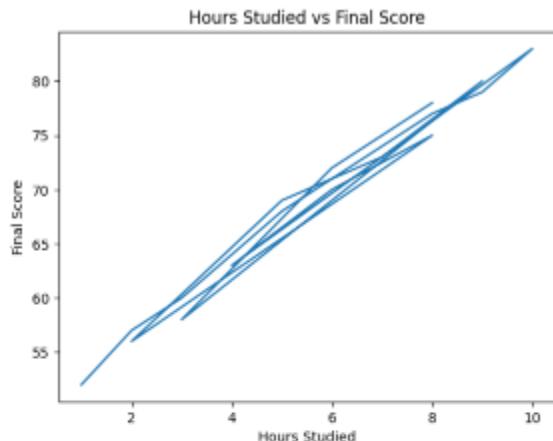
	Hours_Studied	Attendance	Assignment_Score	Midterm_Score	Final_Score	Performance	grid icon
0	1	60	55	50	52	Average	
1	2	65	58	55	57	Average	
2	3	70	60	58	60	Good	
3	4	75	65	62	64	Good	
4	5	80	68	65	68	Good	

### Exercise 3 -

```
import matplotlib.pyplot as plt
plt.figure()
plt.plot(df['Hours_Studied'], df['Final_Score'])
plt.xlabel('Hours Studied')
plt.ylabel('Final Score')
plt.title('Hours Studied vs Final Score')
plt.show()

plt.figure()
plt.hist(df['Final_Score'])
plt.xlabel('Final Score')
plt.ylabel('Frequency')
plt.title('Distribution of Final Scores')
plt.show()
```

*Output -*



#### Exercise 4 -

```
import seaborn as sns

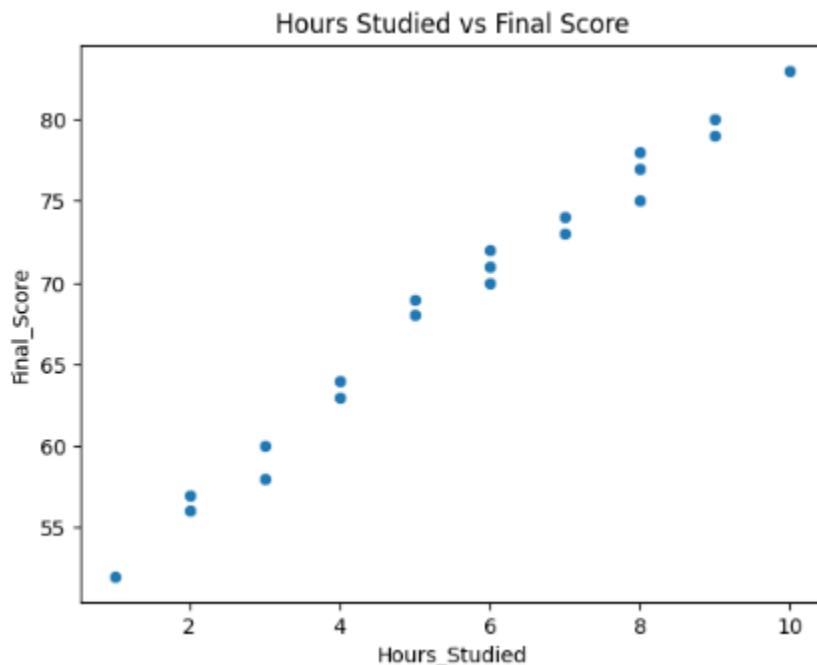
sns.scatterplot(
    x='Hours_Studied',
    y='Final_Score',
    data=df
)
plt.title('Hours Studied vs Final Score')
plt.show()

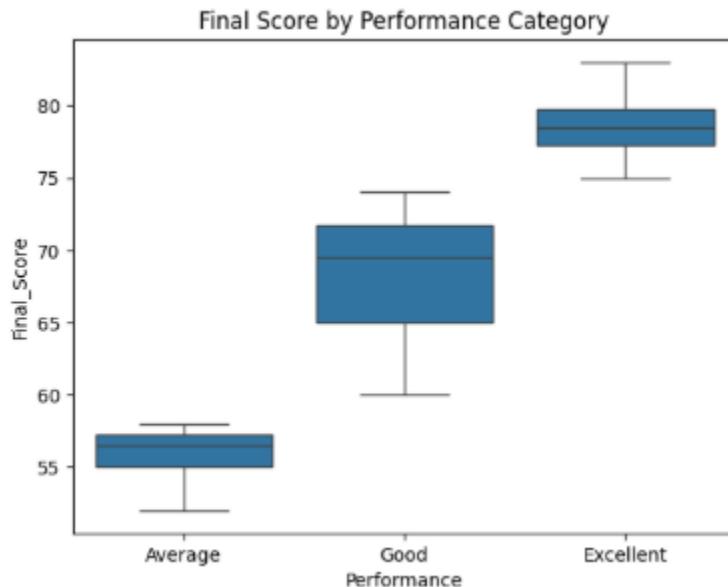
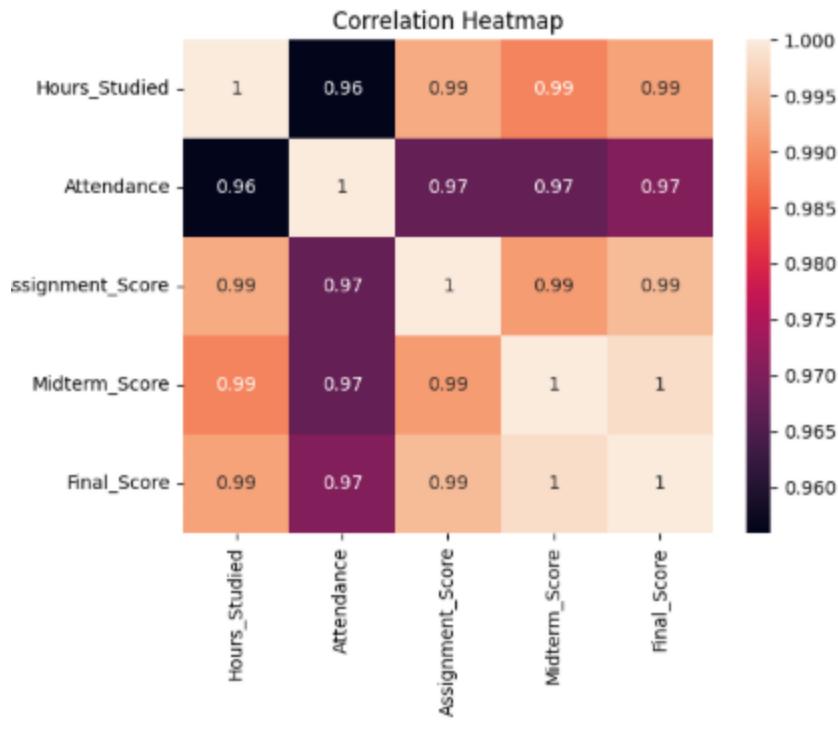
correlation_matrix = df.corr(numeric_only=True)

sns.heatmap(
    correlation_matrix,
    annot=True
)
plt.title('Correlation Heatmap')
plt.show()

sns.boxplot(
    x='Performance',
    y='Final_Score',
    data=df
)
plt.title('Final Score by Performance Category')
plt.show()
```

Output -





## **Conclusion:**

This experiment performed exploratory data analysis on a student performance dataset to understand relationships between attendance, study habits, and exam scores. Statistical measures and Min-Max normalization were applied to summarize and scale the data. Visualizations revealed strong positive correlations between attendance, assignment scores, and midterm performance. The experiment highlights the importance of EDA as a foundational step before applying machine learning models.