Escape room 1

**Room 1: Sensor Data and Basic Operations**

temperatures = [23.5, 25.0, 22.0, 26.5, 24.5]

average\_temp = sum(temperatures) / len(temperatures)

print(f"The average temperature is: {average\_temp}")

**Room 2: Control System Simulation**

pressure = int(input("Enter pressure: "))

if pressure < 20:

print("Pressure too low. Increase it.")

elif pressure > 80:

print("Pressure too high. Decrease it.")

else:

print("Pressure is within the acceptable range.")

**Room 3: Train-Test Split From Scratch**

data = [2.3, 3.1, 4.5, 5.6, 7.2]

size\_60 = int(0.6 \* len(data))

data\_60 = data[:size\_60]

data\_40 = data[size\_60:]

print(f"Data\_60: {data\_60}")

print(f"Data\_40: {data\_40}")

**Room 4: Supervised Learning - Machine Failure Prediction**

def classify\_failure(temp, pressure):

if temp > 75 or pressure > 120:

return "Failure"

else:

return "Normal"

temperature = 80 # Test value

pressure = 110 # Test value

result = classify\_failure(temperature, pressure)

print(f"The machine’s status is: {result}")

**Room 5: Unsupervised Learning - Clustering**

data = [3.1, 2.9, 5.4, 6.2, 1.9, 5.7]

threshold = 3.5

cluster\_1 = [x for x in data if x < threshold]

cluster\_2 = [x for x in data if x >= threshold]

print(f"Cluster 1: {cluster\_1}")

print(f"Cluster 2: {cluster\_2}")

**Room 6: Classification - Student Performance Prediction**

def classify\_student(homework\_score, exam\_score):

if homework\_score >= 85 and exam\_score >= 85:

return "High Achiever"

elif homework\_score >= 60 and exam\_score >= 60:

return "Pass"

else:

return "Fail"

print(classify\_student(90, 92)) # Output: High Achiever

print(classify\_student(65, 70)) # Output: Pass

print(classify\_student(55, 58)) # Output: Fail

**Room 7: Feature Engineering - Student Performance Prediction**

data = {'homework\_score': [40, 60, 80], 'exam\_score': [70, 85, 90]}

performance\_index = [hw \* exam for hw, exam in zip(data['homework\_score'], data['exam\_score'])]

print(f"Performance Index (Homework \* Exam): {performance\_index}")

**Room 8: Regression - Predicting ChatGPT Usage**

def predict\_usage(usage\_values):

diff = usage\_values[-1] - usage\_values[0]

slope = diff / (len(usage\_values) - 1)

return usage\_values[-1] + slope

usage\_values = [50, 60, 70, 80]

next\_week\_usage = predict\_usage(usage\_values)

print(f"Predicted ChatGPT usage for next week: {next\_week\_usage}")

**Room 9: Data Normalization - Prepare Sensor Data for Comparison**

def transform(data):

min\_val = min(data)

max\_val = max(data)

return [(x - min\_val) / (max\_val - min\_val) for x in data]

sensor\_data = [500, 1200, 3000, 8000]

normalized\_data = transform(sensor\_data)

print(f"Normalized data: {normalized\_data}")

**Room 10: Discover the Hidden Algorithm - Classify the New Machine**

import numpy as np

data = np.array([[2, 4], [4, 6], [5, 9], [7, 10]])

labels = ['Normal', 'Warning', 'Maintenance Needed', 'Fault']

new\_machine = np.array([5, 8])

distances = np.linalg.norm(data - new\_machine, axis=1)

nearest\_index = np.argmin(distances)

classification = labels[nearest\_index]

print(f"The new machine is classified as: {classification}")