ای

import numpy as np

def bipolar\_sigmoid(x):

return (np.exp(x) - np.exp(-x)) / (np.exp(x) + np.exp(-x))

def bipolar\_sigmoid\_derivative(x):

y = bipolar\_sigmoid(x)

return 1 - y\*\*2

def tanh(x):

return np.tanh(x)

def calculate\_outputs(input\_matrix):

rows, cols = input\_matrix.shape

output\_sigmoid = np.zeros((rows, cols))

output\_derivative = np.zeros((rows, cols))

output\_tanh = np.zeros((rows, cols))

for i in range(rows):

for j in range(cols):

output\_sigmoid[i, j] = bipolar\_sigmoid(input\_matrix[i, j])

output\_derivative[i, j] = bipolar\_sigmoid\_derivative(input\_matrix[i, j])

output\_tanh[i, j] = tanh(input\_matrix[i, j])

return output\_sigmoid, output\_derivative, output\_tanh

# Example usage with user input

input\_matrix = []

rows = int(input("Enter the number of rows: "))

cols = int(input("Enter the number of columns: "))

print("Enter the matrix elements row by row:")

for i in range(rows):

row = input().split()

row = [float(x) for x in row]

input\_matrix.append(row)

input\_matrix = np.array(input\_matrix)

output\_sigmoid, output\_derivative, output\_tanh = calculate\_outputs(input\_matrix)

print("Output Sigmoid:")

print(output\_sigmoid)

print("Output Derivative:")

print(output\_derivative)

print("Output Tanh:")

print(output\_tanh)