Question 1:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

file\_path = 'C:/Users/2022503035/Documents/machine\_learning\_2022503035/back propagation/INPUT\_VALUES.xlsx'

df = pd.read\_excel(file\_path)

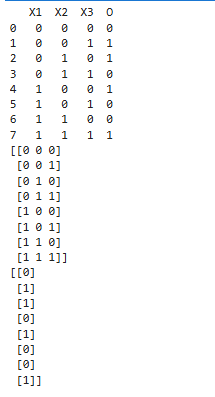
print(df)

X = df[['X1', 'X2', 'X3']].values

print(X)

y = df['O'].values.reshape(-1, 1)

print(y)



W14 = 0.2

W15 = -0.3

W24 = 0.4

W25 = 0.1

W34 = -0.5

W35 = 0.2

W46 = -0.3

W56 = -0.2

b4 = -0.4

b5 = 0.2

b6 = 0.1

W1 = np.array([[W14, W15], [W24, W25], [W34, W35]])

b1 = np.array([b4, b5])

W2 = np.array([[W46], [W56]])

b2 = np.array([b6])

def sigmoid(x):

return 1 / (1 + np.exp(-x))

def feedforward\_propagation(X, W1, b1, W2, b2):

Z1 = np.dot(X, W1) + b1

A1 = sigmoid(Z1)

Z2 = np.dot(A1, W2) + b2

A2 = sigmoid(Z2)

return A1, A2

def compute\_error(y\_pred, y):

return np.mean((y\_pred - y) \*\* 2)

def backpropagate(X, y, A1, A2, W1, b1, W2, b2, learning\_rate):

dA2 = A2 - y

dZ2 = dA2 \* A2 \* (1 - A2)

dW2 = np.dot(A1.T, dZ2)

db2 = np.sum(dZ2, axis=0)

dA1 = np.dot(dZ2, W2.T)

dZ1 = dA1 \* A1 \* (1 - A1)

dW1 = np.dot(X.T, dZ1)

db1 = np.sum(dZ1, axis=0)

W1 -= learning\_rate \* dW1

b1 -= learning\_rate \* db1

W2 -= learning\_rate \* dW2

b2 -= learning\_rate \* db2

return W1, b1, W2, b2

def train\_network(X, y, W1, b1, W2, b2, learning\_rate, iterations=1000):

errors = []

for i in range(iterations):

A1, A2 = forward\_propagation(X, W1, b1, W2, b2)

error = compute\_error(A2, y)

errors.append(error)

W1, b1, W2, b2 = backpropagate(X, y, A1, A2, W1, b1, W2, b2, learning\_rate)

return errors, W1, b1, W2, b2

learning\_rates = [0.01, 0.1, 0.5]

all\_errors = {}

for lr in learning\_rates:

errors, \_, \_, \_, \_ = train\_network(X, y, W1, b1, W2, b2, lr)

all\_errors[lr] = errors

plt.figure(figsize=(10, 6))

plt.plot(range(1000), errors, label=f"Learning Rate: {lr}")

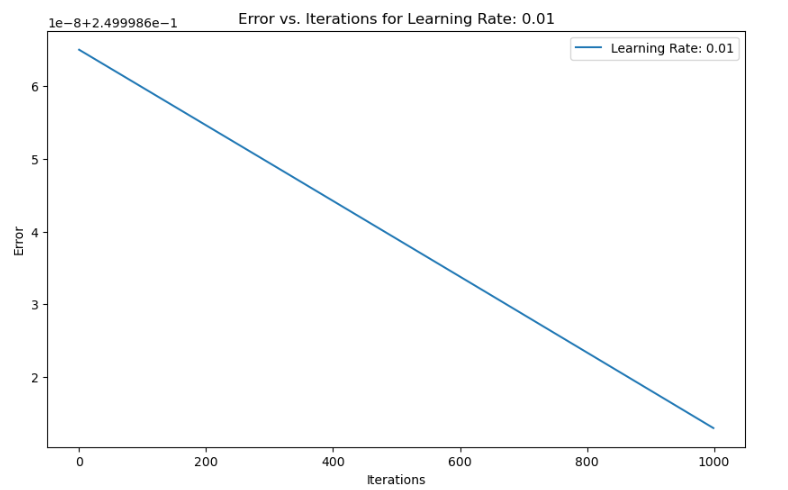
plt.xlabel('Iterations')

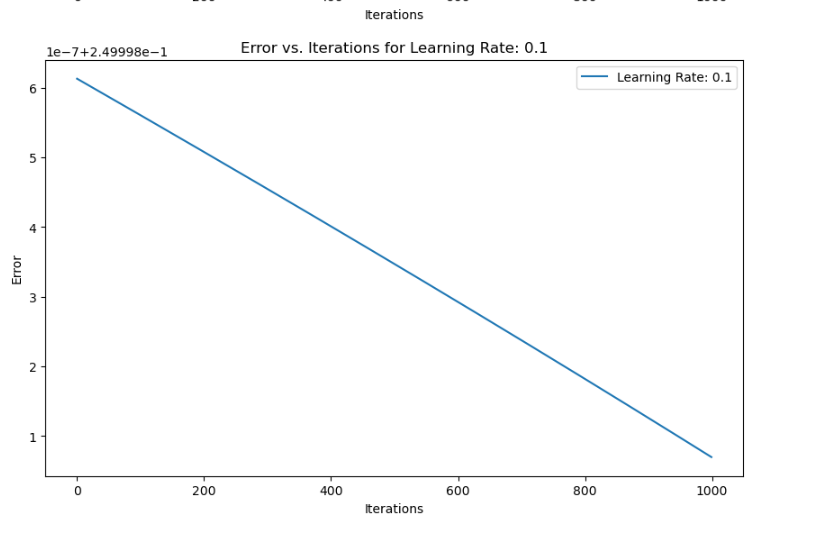
plt.ylabel('Error')

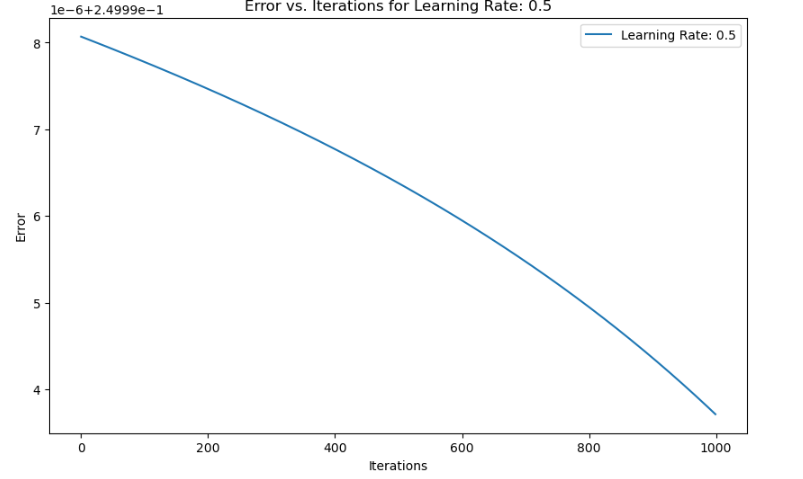
plt.title(f'Error vs. Iterations for Learning Rate: {lr}')

plt.legend()

plt.show()







for lr in learning\_rates:

errors, \_, \_, \_, \_ = train\_network(X, y, W1, b1, W2, b2, lr)

all\_errors[lr] = errors

plt.figure(figsize=(10, 6))

for lr in learning\_rates:

plt.plot(range(1000), all\_errors[lr], label=f"Learning Rate: {lr}")

plt.xlabel('Iterations')

plt.ylabel('Error')

plt.title('Error vs. Iterations for Different Learning Rates')

plt.legend()

plt.show()

