```
import numpy as np
import matplotlib.pyplot as plt

def sigmoide(x):
    return 1 / (1 + np.exp(-x))

def derivada_sigmoide(x):
    return x * (1 - x)

def inicializar_pesos(num_entradas):
    np.random.seed(1)
    return 2 * np.random.random((num_entradas, 1)) - 1

def treinar_rede(entradas, saidas, num_iteracoes, taxa_aprendizado):
    num_amostras, num_entradas = entradas.shape
    pesos_sinapticos = inicializar_pesos(num_entradas)

    erros = []

    for iteracao in range(num_iteracoes):
        saida_camadas_1 = sigmoide(np.dot(entradas, pesos_sinapticos))

        erro = saidas - saida_camadas_1

        erro_quadratico_medio = np.mean(erro**2)
        erros.append(erro_quadratico_medio)

        if iteracao % 1000 == 0:
            print(f"Erro quadrático médio na iteração {iteracao}: {erro_quadratico_medio}")

        # Ajustes nos pesos
        ajustes = erro * derivada_sigmoide(saida_camadas_1)
        pesos_sinapticos += taxa_aprendizado * np.dot(entradas.T, ajustes)

    return pesos_sinapticos, erros

def avaliar_rede(entradas, pesos_sinapticos):
    return sigmoide(np.dot(entradas, pesos_sinapticos))

entradas = np.array([[0, 0, 1],
                     [1, 1, 1],
                     [1, 0, 1],
                     [0, 1, 1]])

saidas = np.array([[0],
                   [1],
                   [1],
                   [0]])

num_iteracoes = 10000
taxa_aprendizado = 0.1

pesos_sinapticos, erros = treinar_rede(entradas, saidas, num_iteracoes, taxa_aprendizado)

saida_camada_1 = avaliar_rede(entradas, pesos_sinapticos)

print("Pesos finais da sinapse após o treinamento: ")
print(pesos_sinapticos)
print("\nSaídas da camada 1 após o treinamento: ")
print(saida_camada_1)

plt.plot(erros)
plt.xlabel("Número de iterações ")
plt.xlabel("Erro Quadrático Médio ")
plt.title("Erro durante o treinamento ")
plt.show()
```