```
#Importando a biblioteca
import numpy as np

#Função do cáculo da sigmóide
def sigmoid(x):
    result = (1+np.exp(-x)) # 1 + (e^ -x)
    return 1/result         # 1 / (1 + (e^ -x))

#Arquitetura da MPL
N_input = 3
N_hidden = 4
N_output = 2

#Vetor dos valores de entrada
x = np.array([0.5, 0.1, -0.2])
target =np.array([0.3, 0.8])
learnrate = 0.5

#Pesos da Camada Oculta
weights_in_hidden = np.array([[-0.07,  0.04, -0.05, 0.07],
                              [ 0.04,  0.10,  0.02, 0.01],
                              [-0.03,  0.04, -0.11, 0.06]])

#Pesos da Camada de Saída
weights_hidden_out = np.array([[-0.10,  0.09],
                               [-0.04,  0.12],
                               [-0.02,  0.04],
                               [-0.01,  0.09]])


#Camada oculta

#Calcule a combinação linear de entradas e pesos sinápticos
hidden_layer_input = np.dot(x, weights_in_hidden)

#Aplicado a função de ativação
hidden_layer_output = sigmoid(hidden_layer_input)

#Camada de Saída

#Calcule a combinação linear de entradas e pesos sinápticos
output_layer_in = np.dot(hidden_layer_output, weights_hidden_out)

#Aplicado a função de ativação
output = sigmoid(output_layer_in)

print('As saídas da rede são',output)

## TODO: Cálculo do Erro
error = target - output
#print('Erro da Rede: ',error)

# TODO: Calcule o termo de erro de saída (Gradiente da Camada de Saída)
output_error_term = error * output * (1 - output)

# TODO: Calcule a contribuição da camada oculta para o erro
hidden_error = np.dot(weights_hidden_out,output_error_term)

# TODO: Calcule o termo de erro da camada oculta (Gradiente da Camada Oculta)
hidden_error_term = hidden_error * hidden_layer_output * (1 - hidden_layer_output)

# TODO: Calcule a variação do peso da camada de saída
delta_w_h_o = learnrate * output_error_term*hidden_layer_output[:, None]
print('Variação do peso da camada de saída (delta_w_h_o): ',delta_w_h_o)

# TODO: Calcule a variação do peso da camada oculta
delta_w_i_h = learnrate * hidden_error_term * x[:, None]
print('Variação do peso da camada oculta (delta_w_i_h): ',delta_w_i_h)
```

```
##Atualização dos Pesos
weights_input_hidden = learnrate * delta_w_i_h
print('weights_input_hidden: ',weights_input_hidden)
weights_hidden_output = learnrate * delta_w_h_o
print('weights_hidden_output: ',weights_hidden_output)
```