import numpy as np  
  
def sigmoid(z):  
 return 1 / (1 + np.exp(-z))  
def h(x, w):  
 return sigmoid(np.dot(x, w))  
  
def compute\_gradient(X, Y, w):  
 *"""  
 计算梯度  
 :param X: 输入[m,n]：m个样本，维度为n  
 :param Y: 标签，维度[n,1]  
 :param w: 可学习参数，维度[n,1]  
 :return: 梯度，维度[n,1]  
 """* m, n = X.shape  
 gradient = np.zeros((n, 1))  
  
 for i in range(m):  
 xi = X[i].reshape(-1, 1) # 转置  
 h\_i = h(xi.T, w)  
 error = h\_i - Y[i]  
 gradient += error \* xi  
  
 gradient /= m  
 return gradient  
  
  
X = np.array([[1, 2], [3, 4], [5, 6]])  
Y = np.array([0, 1, 1]).reshape(-1, 1)  
w = np.zeros((2, 1))  
  
grad = compute\_gradient(X, Y, w)  
print("梯度:", grad.ravel())