1. We adopt the squared error loss: $L = \frac{1}{2}(h(x_1, x_2) - y)^2$

let $z = b + w_1 x_1 + w_2 x_2 \Rightarrow h(x_1, x_2) = \sigma(z)$

Since $\sigma'(z) = \sigma(z)(1-\sigma(z))$, the gradients with respect to the parameters are:

For b: $\frac{\partial L}{\partial b} = (h-y)\sigma(z)(1-\sigma(z))$

For w_1: $\frac{\partial L}{\partial w_1} = (h-y)\sigma(z)(1-\sigma(z)) \cdot x_1$

For w_2: $\frac{\partial L}{\partial w_2} = (h-y)\sigma(z)(1-\sigma(z)) \cdot x_2$

Thus, the gradient vector is $\nabla_\theta L = (h-y)\sigma(z)(1-\sigma(z)) \cdot (1, x_1, x_2)$

The standard SGD update is $\theta' = \theta^\circ - \eta \nabla_\theta L$, where η is the learning rate

For the given data point and parameter initialization, we compute

$$z = 4 + 5(1) + 6(2) = 21, \quad h = \sigma(21)$$

Hence, the update becomes $\theta' = (4, 5, 6) - \eta \cdot \left((\sigma(21) - 3)\sigma(21)(1-\sigma(21))\right) \cdot (1, 1, 2)$

Explicitly, this gives: $b' = 4 - \eta \cdot \left((\sigma(21) - 3)\sigma(21)(1-\sigma(21))\right)$

$$w_1' = 5 - \eta \cdot \left((\sigma(21)-3)\sigma(21)(1-\sigma(21)) \cdot 1\right)$$

$$w_2' = 6 - \eta \cdot \left((\sigma(21)-3)\sigma(21)(1-\sigma(21)) \cdot 2\right)$$

2. (a) The sigmoid function is $\sigma(x) = \frac{1}{1+e^{-x}}$

For $k=1$, $\sigma'(x) = \sigma(x)(1-\sigma(x))$

For $k=2$, $\sigma''(x) = \sigma'(x)(1-2\sigma(x)) = \sigma(x)(1-\sigma(x))(1-2\sigma(x))$

For $k=3$, $\sigma'''(x) = \sigma(x)(1-\sigma(x))(1-6\sigma(x)+6\sigma(x)^2)$

(b) The sigmoid function is closely related to the hyperbolic tangent function. In fact, we have: $\sigma(x) = \frac{1}{1+e^{-x}} = \frac{1}{2}(1+\tanh(\frac{x}{2}))$.

This identity is useful because it connects the logistic function to hyperbolic functions, which often arise in the analysis of differential equations and neural network activations.

3. Alternative activation functions: Why do modern neural networks often prefer the Relu or other nonlinearities instead of the sigmoid function?

Gradient vanishing problem: Since the sigmoid function saturates at very large positive or negative inputs, how does this affect gradient propagation in deep networks? What are common solutions?