Neural Network Forward Propagation

In this example, we'll consider a simple neural network with the following architecture:

- **Input Layer:** 2 neurons
- **Hidden Layer:** 2 neurons with ReLU activation
- **Output Layer:** 1 neuron with Sigmoid activation

Step 1: Initialize Input

Let the input vector be:

$$\mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Step 2: Define Weights and Biases

Initialize the weights and biases for each layer.

Weights and Biases for Hidden Layer

$$\mathbf{W}^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} \end{bmatrix}, \mathbf{b}^{(1)} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \end{bmatrix}$$

Weights and Biases for Output Layer

$$\mathbf{W}^{(2)} = \begin{bmatrix} w_{11}^{(2)} & w_{12}^{(2)} \end{bmatrix}, \mathbf{b}^{(2)} = b^{(2)}$$

Step 3: Forward Propagation

3.1 Compute Weighted Sum for Hidden Layer

The weighted input to the hidden layer neurons is calculated as:

$$Z^{(1)} = \mathbf{W}^{(1)}\mathbf{X} + \mathbf{b}^{(1)}$$

3.2 Apply ReLU Activation Function

The ReLU activation function is defined as:

$$ReLU(z) = \max(0, z)$$

Applying ReLU to each element of $Z^{(1)}$:

$$A^{(1)} = ReLU(Z^{(1)})$$

3.3 Compute Weighted Sum for Output Layer

The weighted input to the output neuron is:

$$Z^{(2)} = \mathbf{W}^{(2)} A^{(1)} + b^{(2)}$$

3.4 Apply Sigmoid Activation Function

The Sigmoid activation function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Applying Sigmoid to $Z^{(2)}$:

$$A^{(2)} = \sigma(Z^{(2)})$$

Step 4: Summary of Forward Propagation

1. Input: $\mathbf{X}$
2. Hidden Layer:
   i. Weighted Sum: $Z^{(1)} = \mathbf{W}^{(1)}\mathbf{X} + \mathbf{b}^{(1)}$
   ii. Activation: $A^{(1)} = ReLU(Z^{(1)})$
3. Output Layer:
   i. Weighted Sum: $Z^{(2)} = \mathbf{W}^{(2)} A^{(1)} + b^{(2)}$
   ii. Activation: $A^{(2)} = \sigma(Z^{(2)})$

Example Calculation

Let's assign specific values to the inputs, weights, and biases for a concrete example.

Given:

- $x_1 = 1$
- $x_2 = 2$
- $w_{11}^{(1)} = 0.5$
- $w_{12}^{(1)} = -0.6$
- $w_{21}^{(1)} = 0.1$
- $w_{22}^{(1)} = 0.8$

- $b_1^{(1)} = 0.0$
- $b_2^{(1)} = 0.1$
- $w_{11}^{(2)} = 0.7$
- $w_{12}^{(2)} = -1.2$
- $b^{(2)} = 0.3$

4.1 Compute $Z^{(1)}$

$Z^{(1)} = \mathbf{W}^{(1)}\mathbf{X} + \mathbf{b}^{(1)}$

Calculating each component:

- $Z_1^{(1)} = 0.5 \times 1 + (-0.6) \times 2 + 0.0 = 0.5 - 1.2 + 0 = -0.7$
- $Z_2^{(1)} = 0.1 \times 1 + 0.8 \times 2 + 0.1 = 0.1 + 1.6 + 0.1 = 1.8$

Thus:

$$Z^{(1)} = \begin{bmatrix} -0.7 \\ 1.8 \end{bmatrix}$$

4.2 Apply ReLU to Get $A^{(1)}$

$$A^{(1)} = ReLU(Z^{(1)}) = \begin{bmatrix} ReLU(-0.7) \\ ReLU(1.8) \end{bmatrix} = \begin{bmatrix} 0 \\ 1.8 \end{bmatrix}$$

4.3 Compute $Z^{(2)}$

$Z^{(2)} = \mathbf{W}^{(2)}A^{(1)} + b^{(2)} = 0.7 \times 0 + (-1.2) \times 1.8 + 0.3 = 0 - 2.16 + 0.3 = -1.86$

4.4 Apply Sigmoid to Get $A^{(2)}$

$$A^{(2)} = \sigma(Z^{(2)}) = \frac{1}{1 + e^{-(-1.86)}} = \frac{1}{1 + e^{1.86}}$$

Calculating the value:

- $e^{1.86} \approx 6.419$
- $A^{(2)} = \frac{1}{1 + 6.419} = \frac{1}{7.419} \approx 0.135$

Thus, the output of the network is approximately 0.135.

Conclusion

Through forward propagation, the neural network processes the input $\mathbf{X} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ and produces an output $A^{(2)} \approx 0.135$ using a combination of ReLU and Sigmoid activation functions.

General Forward Propagation Equations

For a neural network with multiple layers, the forward propagation can be generalized as:

- **Layer l:** <
  - Weighted Sum: $Z^{(l)} = \mathbf{W}^{(l)} A^{(l-1)} + \mathbf{b}^{(l)}$
  - Activation: $A^{(l)} = \text{activation}(Z^{(l)})$

The final output layer applies an appropriate activation function based on the task (e.g., Sigmoid for binary classification).

Activation Functions Used

ReLU (Rectified Linear Unit)

$ReLU(z) = \max(0, z)$

ReLU introduces non-linearity by outputting zero for negative inputs and the input itself for positive inputs.

Sigmoid

$\sigma(z) = \frac{1}{1 + e^{-z}}$

Sigmoid squashes the input into a range between 0 and 1, making it suitable for probability estimation in binary classification tasks.

Final Notes

Forward propagation is a fundamental process in neural networks, enabling the transformation of input data through layers of weights, biases, and activation functions to produce meaningful outputs. Understanding each step mathematically ensures a solid foundation for building and troubleshooting neural network models.