Report about Neural Networks

Vasilev Vadim gr. 5130203/20102

01.12.2024

1. Perceptron

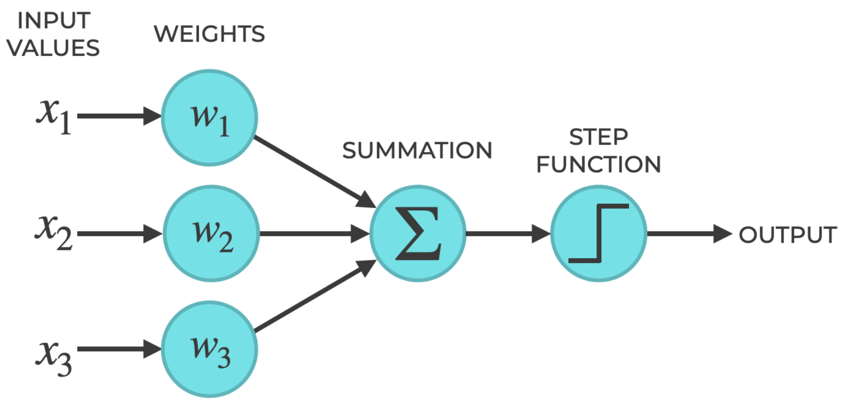

Figure 1: Perceptron

Vector Representation of Data

Inputs: $\mathbf{X} = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^n$,
Weights: $\mathbf{W} = [w_1, w_2, \ldots, w_n]^\top \in \mathbb{R}^n$,
Bias: $b \in \mathbb{R}$,
Output: $y \in \{-1, 1\}$.

Mathematical Formulation

Linear Combination:

$$z = \mathbf{X}^\top \mathbf{W} + b$$

Activation Function (Step Function):

$$\hat{y} = f(z) = \begin{cases} 1 & \text{if } z \geq 0, \\ -1 & \text{if } z < 0. \end{cases}$$

Loss Function:

$$L = \begin{cases} 0 & \text{if } y\hat{y} > 0, \\ -yz & \text{otherwise.} \end{cases}$$

Gradient Descent Algorithm

Weights and bias are updated using the perceptron learning rule:

$$w_i \leftarrow w_i + \eta(y - \hat{y})x_i, \quad b \leftarrow b + \eta(y - \hat{y}),$$

where η is the learning rate.

Explanation of gradient descendent algorithm

Gradient descent is an optimization algorithm used to minimize the loss function. For the perceptron, the algorithm iteratively adjusts the weights and bias to reduce errors.

2. Logistic Regression

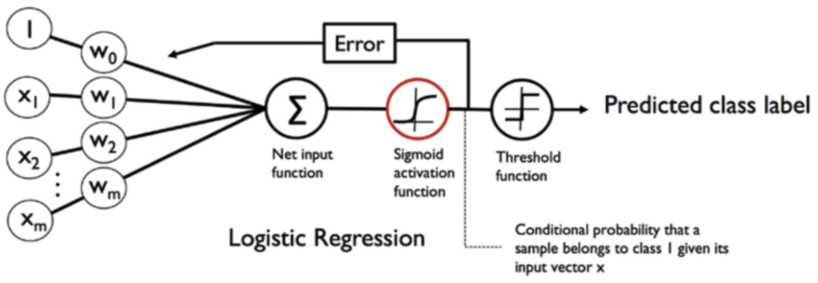
1
x1
x2
xm
w0
w1
w2
wm
Error
Σ
Net input function
Sigmoid activation function
Threshold function
Predicted class label
Logistic Regression
Conditional probability that a sample belongs to class 1 given its input vector x

Figure 2: Logistic Regression

Vector Representation of Data

Inputs: $\mathbf{X} = [x_1, x_2, \ldots, x_n]^\top \in \mathbb{R}^n$,
Weights: $\mathbf{W} = [w_1, w_2, \ldots, w_n]^\top \in \mathbb{R}^n$,
Bias: $b \in \mathbb{R}$,
Output: $y \in \{0, 1\}$.

Mathematical Formulation

Linear Combination:

$$z = \mathbf{W}^\top \mathbf{X} + b$$

Activation Function (Sigmoid):

$$\hat{y} = f(z) = \frac{1}{1 + e^{-z}}$$

Loss Function (Binary Cross-Entropy):

$$L = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

Gradient Descent Algorithm

Gradients:

$$\frac{\partial L}{\partial w_i} = (\hat{y} - y)x_i, \quad \frac{\partial L}{\partial b} = (\hat{y} - y)$$

Weight and Bias Updates:

$$w_i \leftarrow w_i - \eta \frac{\partial L}{\partial w_i}, \quad b \leftarrow b - \eta \frac{\partial L}{\partial b}.$$

Explanation of gradient descendent algorithm

Gradient descent minimizes the binary cross-entropy loss by updating weights and biases iteratively.

3. Multilayer Perceptron (MLP)

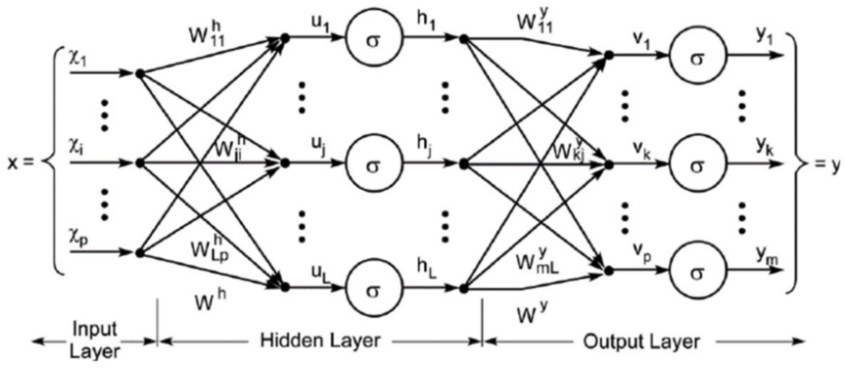

Figure 3: Multilayer Perceptron

Vector Representation of Data

Inputs: $\mathbf{x} \in \mathbb{R}^n$,
Weights: $\mathbf{W}^{(l)}$ for layer l,

Biases: $\mathbf{b}^{(l)}$ for layer l,
Outputs: $\hat{y} \in \mathbb{R}^m$ (for multi-class).

Mathematical Formulation

Forward Pass:
For layer l, the pre-activation and activation are:

$$z^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}, \quad \hat{y} = \mathbf{a}^{(l)} = f(z^{(l)}),$$

where f is a nonlinear activation function (e.g., ReLU or Sigmoid).

Loss Function: For multi-class classification:

$$L = -\sum_{i=1}^{m} y_i \log(\hat{y}_i),$$

where $\hat{y}_i = \text{Softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$.

Backpropagation and Gradient Descent

Gradients:

$$\delta^{(l)} = \frac{\partial L}{\partial z^{(l)}} = \mathbf{W}^{(l+1)\top}\delta^{(l+1)} \odot f'(z^{(l)})$$

Weight and Bias Updates:

$$\mathbf{W}^{(l)} \leftarrow \mathbf{W}^{(l)} - \eta\frac{\partial L}{\partial \mathbf{W}^{(l)}}, \quad \mathbf{b}^{(l)} \leftarrow \mathbf{b}^{(l)} - \eta\frac{\partial L}{\partial \mathbf{b}^{(l)}}.$$

Explanation of gradient descendent algorithm

Backpropagation is used to compute gradients layer by layer. The gradient descent algorithm updates weights and biases to minimize loss.