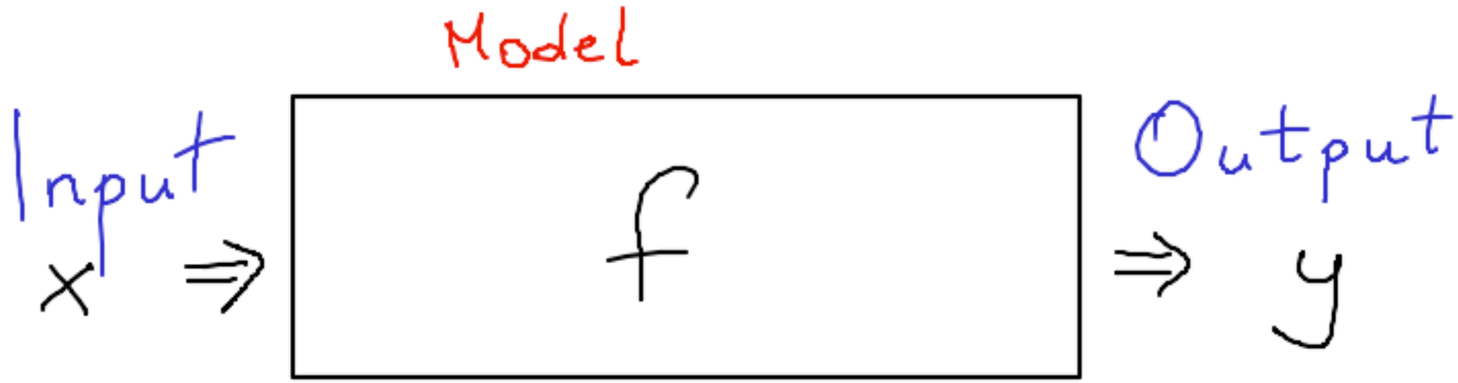
Neural Networks

Calculus

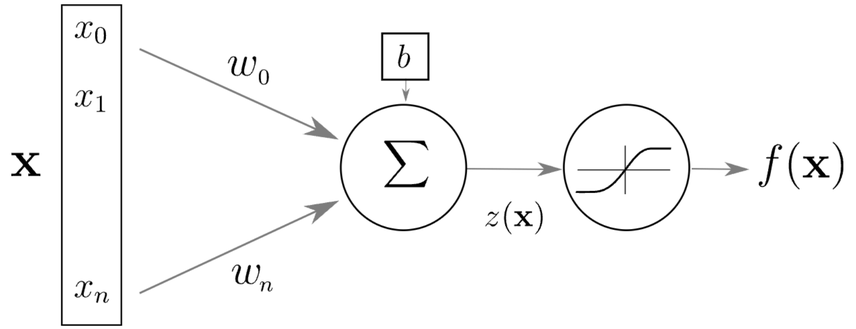
Rafał Nowak March 7, 2023

Computation graph (recap)

- neural network = parametrized, non-linear function



Artificial neuron



$$\mathrm{z}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b}$$

$$\mathrm{f}(\mathbf{x}) = \mathrm{g}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b})$$

- $\mathbf{x}, \mathrm{f}(\mathbf{x})$ input and output
- $\mathrm{z}(\mathbf{x})$ pre-activation
- $\mathbf{w}, \mathrm{b}$ weights and bias
- g activation function

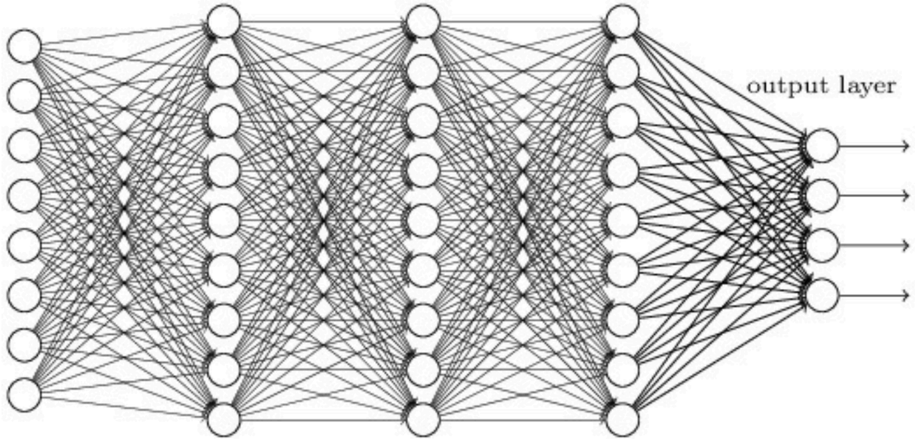
source: m2dsupsdlclass.github.io

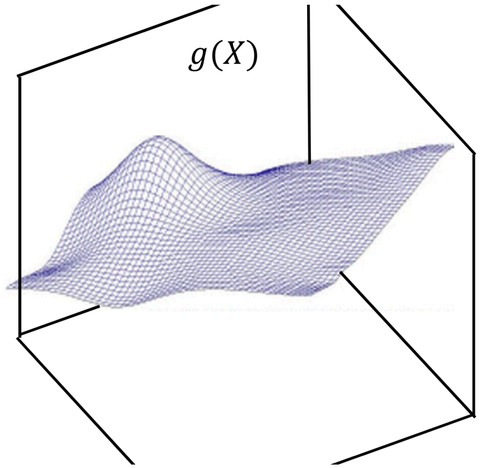
Supervised **learning**

- regression
- classification
- …

What we learn?

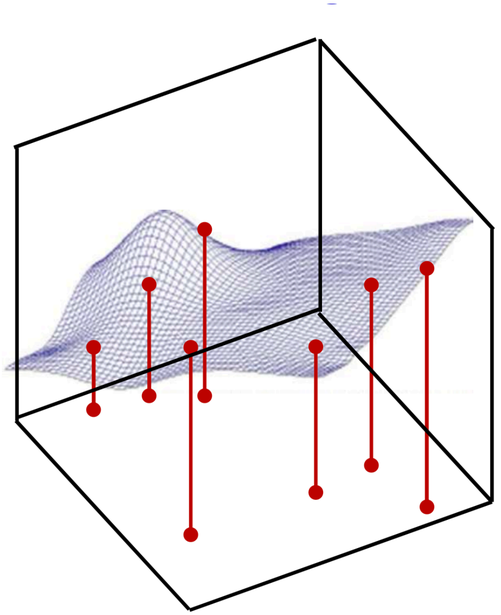
- network parameters W
- $f(X, W) \approx g(X)$

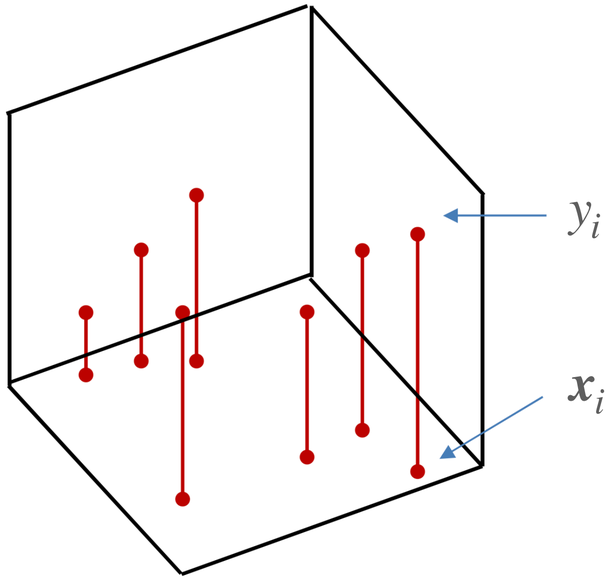




Learning by sampling

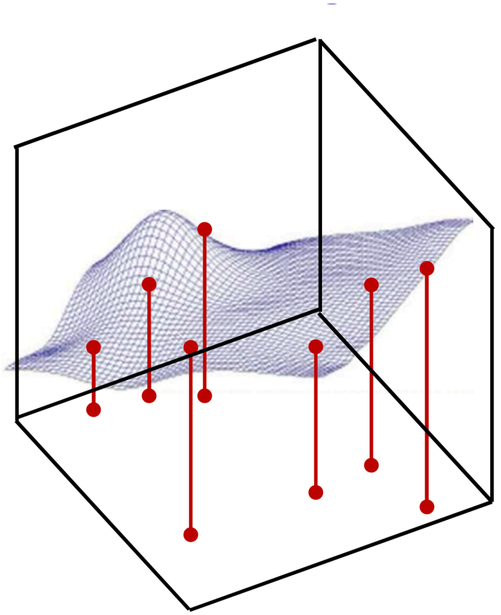
- training set $X_{\text{train}} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times n}$
- $f(X, W) \approx g(X)$
- Questions:
  - good sampling?
- Examples:
  - images + labels
  - speech recordings and their transcription

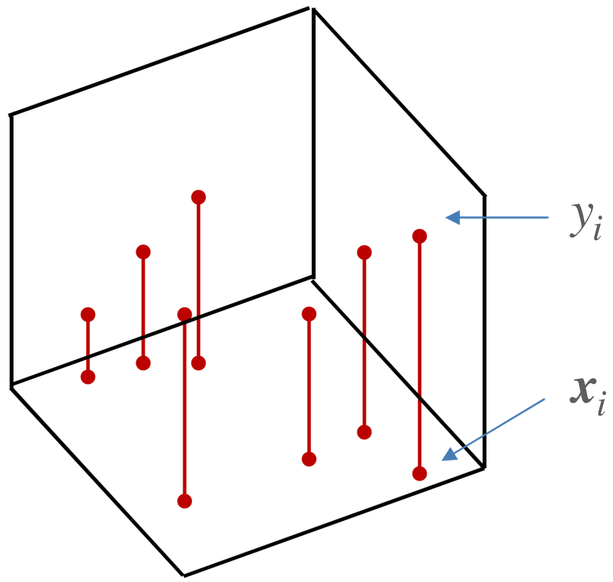




Learning the function

- Estimate the training parameters to *fit* the training data points
- Questions:
  - network architecture? basis functions?
  - approximation error?



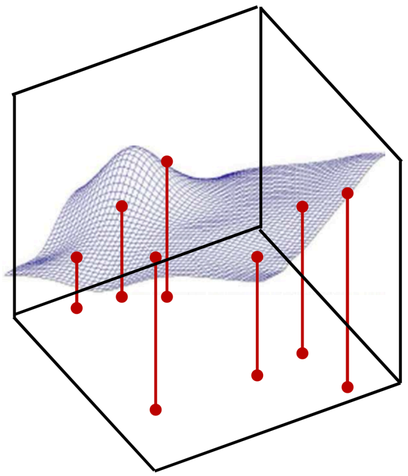


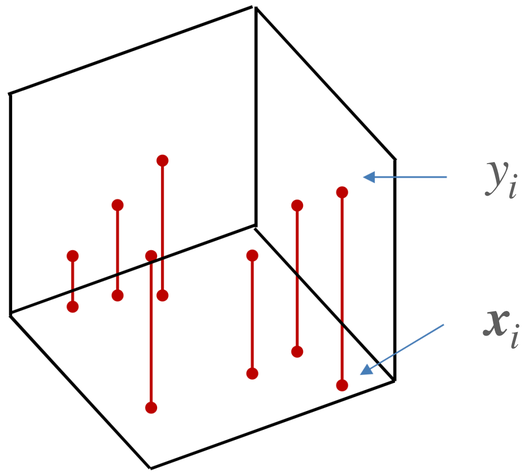
Learning the function

Error function (loss function)

- Given the training set $X_{\text{train}} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times n}$

$$\text{Err}(W) = \frac{1}{N} \sum_{i=1}^{N} \text{div}(f(\boldsymbol{x}_i; W), y_i)$$





Logistic regression

Binary classification

- input data ($i = 1,2,\ldots,N$)

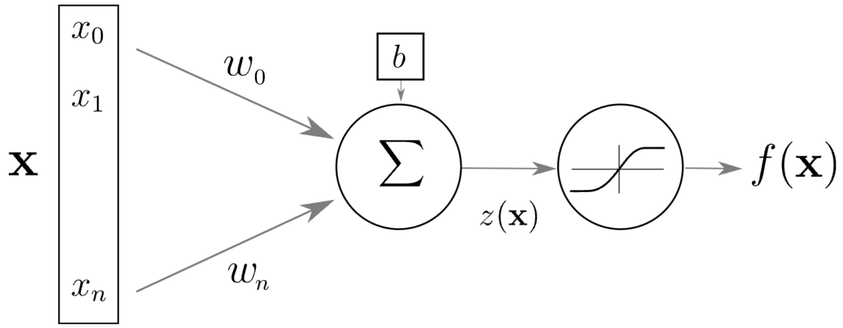
  $\boldsymbol{x}_i \in \mathbb{R}^n, \qquad y_i \in 0,1$

- parameters

  $\boldsymbol{w} \in \mathbb{R}^n, \quad b \in \mathbb{R}$

- learning function (for given sample point $(\boldsymbol{x}, y)$)

  $f(\boldsymbol{x};\, \boldsymbol{w}, b) = p(y = 1 \,|\, \boldsymbol{x})$

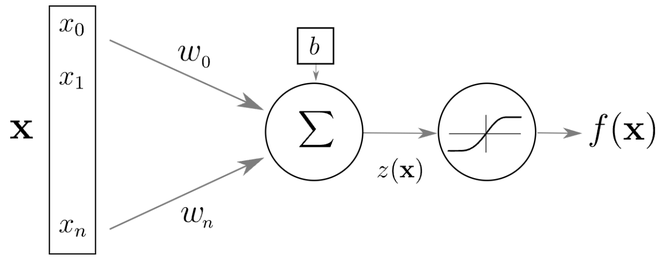


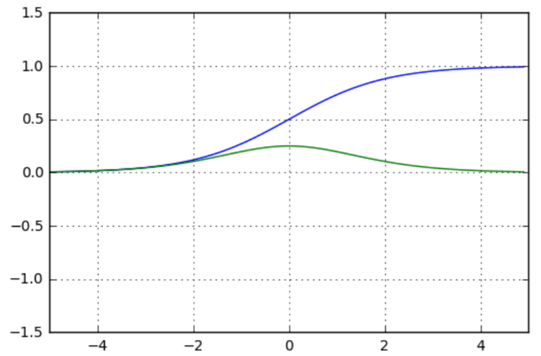
Logistic regression

Binary classification

- sigmoid function

$$g(x) = \sigma(x)$$

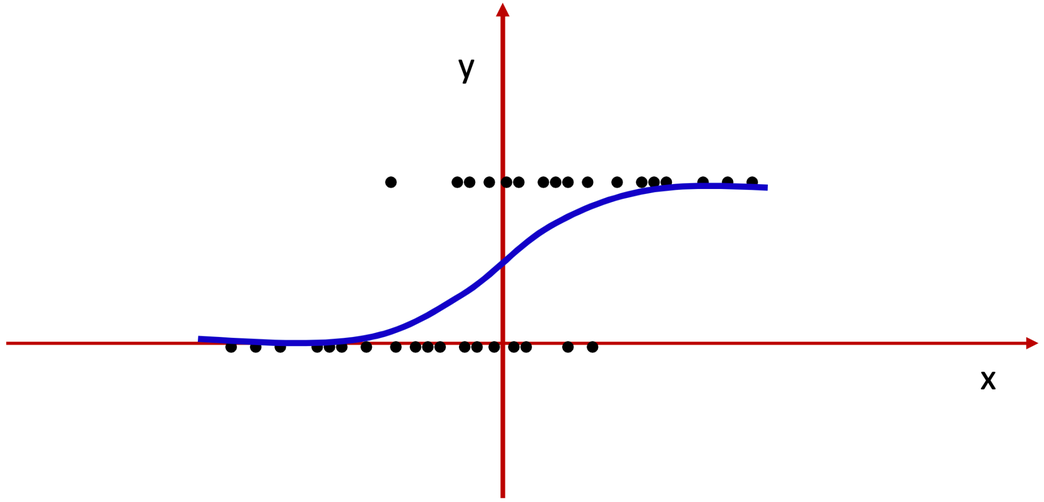




$$\text{sigm}(x) = \frac{1}{1 + e^{-x}}$$

$$\text{sigm}'(x) = \text{sigm}(x)(1 - \text{sigm}(x))$$

The *probability* of y=1



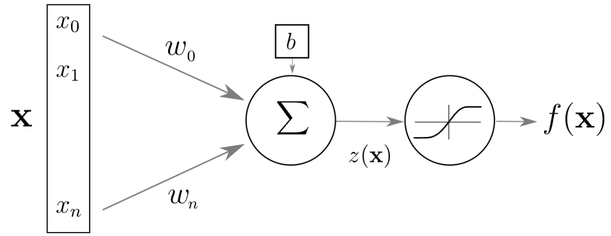
Logistic regression

Binary classification - loss function

- binary **cross entropy**

$$f(\boldsymbol{x};\ \boldsymbol{w}, b) = \hat{y} = p(y = 1 \,|\, \boldsymbol{x}) \in (0,1)$$

$$\text{div}(\hat{y}, y) = -\, y \log \hat{y} - (1 - y)\log(1 - \hat{y})$$



$$\mathbf{z}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b}$$

$$\mathrm{f}(\mathbf{x}) = \mathrm{g}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b})$$

- $\mathbf{x}, \mathrm{f}(\mathbf{x})$ input and output
- $\mathbf{z}(\mathbf{x})$ pre-activation
- $\mathbf{w}, \mathrm{b}$ weights and bias

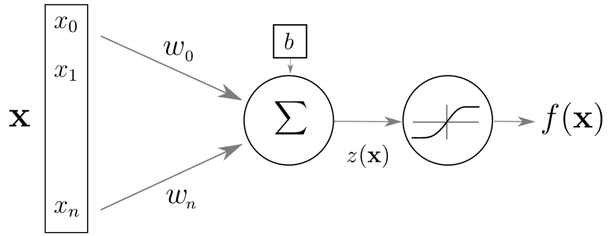
Linear regression (recap)

Loss function

- (scaled) L_2 error (squared euclidean distance)

$$f(\boldsymbol{x};\, \boldsymbol{w}, b) = \hat{y}$$

$$\text{div}(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$$



$$\mathbf{z}(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b}$$

$$\mathrm{f}(\mathbf{x}) = \mathrm{g}(\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b})$$

- $\mathbf{x}, \mathrm{f}(\mathbf{x})$ input and output
- $\mathbf{z}(\mathbf{x})$ pre-activation
- $\mathbf{w}, \mathrm{b}$ weights and bias

Logistic regression

Derivative of loss function

$$z = \boldsymbol{w}^T \boldsymbol{x} + b, \qquad \hat{y} = \sigma(z), \qquad \frac{\partial \hat{y}}{\partial z} = \sigma(z)(1 - \sigma(z))$$

$$L(\boldsymbol{w}, b) = -\, y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

$$\frac{\partial L}{\partial \hat{y}} = ?\,, \qquad \frac{\partial L}{\partial z} = ?\,, \qquad \frac{\partial L}{\partial \boldsymbol{w}} = ?\,, \qquad \frac{\partial L}{\partial b} = ?$$

Matrix calculus

Notation

- $x \in \mathbb{R}$
- $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n$
- $f : \mathbb{R}^n \to \mathbb{R},\ y = f(\boldsymbol{x})$
- $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m,\ \boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$

Matrix calculus

(Scalar) derivative

- $x \in \mathbb{R}$

  $f : \mathbb{R} \to \mathbb{R}$

  $y = f(x), \qquad \dfrac{\partial f}{\partial x} = f'(x)$

Matrix calculus

Gradient (horizontal vector)

- $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n$

  $f : \mathbb{R}^n \to \mathbb{R}$

  $$y = f(\boldsymbol{x}), \qquad \nabla_{\boldsymbol{x}} y = \nabla y = \frac{\partial y}{\partial \boldsymbol{x}} = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right]$$

Matrix calculus

Gradient

- $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n$
  $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \\ \vdots \\ f_m(\boldsymbol{x}) \end{bmatrix}, \qquad \nabla_{\boldsymbol{x}} \boldsymbol{y} = \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}, \frac{\partial f_1}{\partial x_2}, \ldots, \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1}, \frac{\partial f_2}{\partial x_2}, \ldots, \frac{\partial f_2}{\partial x_n} \\ \vdots \\ \frac{\partial f_m}{\partial x_1}, \frac{\partial f_m}{\partial x_2}, \ldots, \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Examples

- $\frac{\partial y}{\partial x}(c) = 0$ (c - constant)
- $\frac{\partial y}{\partial x}(x) = 1$
- $\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}(\boldsymbol{x}) = ?$ ($\boldsymbol{x} \in \mathbb{R}^n$)

Examples

$\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$

- $\dfrac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{x} + \boldsymbol{w}) = ?$
- $\dfrac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{x} * \boldsymbol{w}) = ?$
- $\dfrac{\partial}{\partial \boldsymbol{x}}(\boldsymbol{x} * \boldsymbol{x}) = ?$
- $\dfrac{\partial}{\partial \boldsymbol{x}}(A\boldsymbol{x}) = ?$

Examples

$f : \mathbb{R}^n \to \mathbb{R}$

- $\frac{\partial}{\partial \boldsymbol{x}} \text{sum}(\boldsymbol{x}) = ?$
- $\frac{\partial}{\partial \boldsymbol{x}} (\boldsymbol{x}^T \boldsymbol{w}) = ?$
- $\frac{\partial}{\partial \boldsymbol{x}} (\boldsymbol{x}^T \boldsymbol{x}) = ?$
- $\frac{\partial}{\partial \boldsymbol{x}} (\boldsymbol{x}^T A \boldsymbol{x}) = ?$

Total derivative chain rule

- $$\frac{\partial f(\boldsymbol{u})}{\partial x} = \frac{\partial f(u_1, u_2, \ldots, u_n)}{\partial x} = \sum_i \frac{\partial f}{\partial u_i}\frac{\partial u_i}{\partial x} \qquad (u_i = u_i(x), \quad x \in \mathbb{R})$$

Chain rule

$x \in \mathbb{R}$

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{g}(x)) = \boldsymbol{f} \circ \boldsymbol{g}\,(x) = \begin{bmatrix} f_1(\boldsymbol{g}(x)) \\ f_2(\boldsymbol{g}(x)) \\ \vdots \\ f_n(\boldsymbol{g}(x)) \end{bmatrix}$$

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{g}(x))}{\partial x} \\ \frac{\partial f_2(\boldsymbol{g}(x))}{\partial x} \\ \vdots \\ \frac{\partial f_n(\boldsymbol{g}(x))}{\partial x} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_1}{\partial g_2}\frac{\partial g_2}{\partial x} + \cdots + \frac{\partial f_1}{\partial g_n}\frac{\partial g_n}{\partial x} \\ \frac{\partial f_2}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_2}{\partial g_2}\frac{\partial g_2}{\partial x} + \cdots + \frac{\partial f_2}{\partial g_n}\frac{\partial g_n}{\partial x} \\ \vdots \\ \frac{\partial f_n}{\partial g_1}\frac{\partial g_1}{\partial x} + \frac{\partial f_n}{\partial g_2}\frac{\partial g_2}{\partial x} + \cdots + \frac{\partial f_n}{\partial g_n}\frac{\partial g_n}{\partial x} \end{bmatrix} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{g}} \cdot \frac{\partial \boldsymbol{g}}{\partial x} \qquad \text{(matrix} \cdot \text{vector)}$$

Chain rule

$\boldsymbol{x} \in \mathbb{R}^n$

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{g}(\boldsymbol{x})) = \boldsymbol{f} \circ \boldsymbol{g}\,(\boldsymbol{x}) = \begin{bmatrix} f_1(\boldsymbol{g}(\boldsymbol{x})) \\ f_2(\boldsymbol{g}(\boldsymbol{x})) \\ \vdots \\ f_n(\boldsymbol{g}(\boldsymbol{x})) \end{bmatrix}$$

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1(\boldsymbol{g}(x))}{\partial \boldsymbol{x}} \\ \frac{\partial f_2(\boldsymbol{g}(x))}{\partial \boldsymbol{x}} \\ \vdots \\ \frac{\partial f_n(\boldsymbol{g}(x))}{\partial \boldsymbol{x}} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial g_1}\frac{\partial g_1}{\partial \boldsymbol{x}} + \frac{\partial f_1}{\partial g_2}\frac{\partial g_2}{\partial \boldsymbol{x}} + \cdots + \frac{\partial f_1}{\partial g_n}\frac{\partial g_n}{\partial \boldsymbol{x}} \\ \frac{\partial f_2}{\partial g_1}\frac{\partial g_1}{\partial \boldsymbol{x}} + \frac{\partial f_2}{\partial g_2}\frac{\partial g_2}{\partial \boldsymbol{x}} + \cdots + \frac{\partial f_2}{\partial g_n}\frac{\partial g_n}{\partial \boldsymbol{x}} \\ \vdots \\ \frac{\partial f_n}{\partial g_1}\frac{\partial g_1}{\partial \boldsymbol{x}} + \frac{\partial f_n}{\partial g_2}\frac{\partial g_2}{\partial \boldsymbol{x}} + \cdots + \frac{\partial f_n}{\partial g_n}\frac{\partial g_n}{\partial \boldsymbol{x}} \end{bmatrix} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{g}} \cdot \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}} \qquad \text{(matrix} \cdot \text{matrix)}$$

Logistic regression

Derivative of loss function

$$z = \boldsymbol{w}^T \boldsymbol{x} + b, \qquad \hat{y} = \sigma(z), \qquad \frac{\partial \hat{y}}{\partial z} = \sigma(z)(1 - \sigma'(z))$$

$$L(\boldsymbol{w}, b) = -\, y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

$$\frac{\partial L}{\partial \hat{y}} = ?\,, \qquad \frac{\partial L}{\partial z} = ?\,, \qquad \frac{\partial L}{\partial \boldsymbol{w}} = ?\,, \qquad \frac{\partial L}{\partial b} = ?$$

Logistic regression

Derivative of loss function

$$z = \boldsymbol{w}^T \boldsymbol{x} + b, \qquad \hat{y} = \sigma(z), \qquad \frac{\partial \hat{y}}{\partial z} = \sigma(z)(1 - \sigma'(z))$$

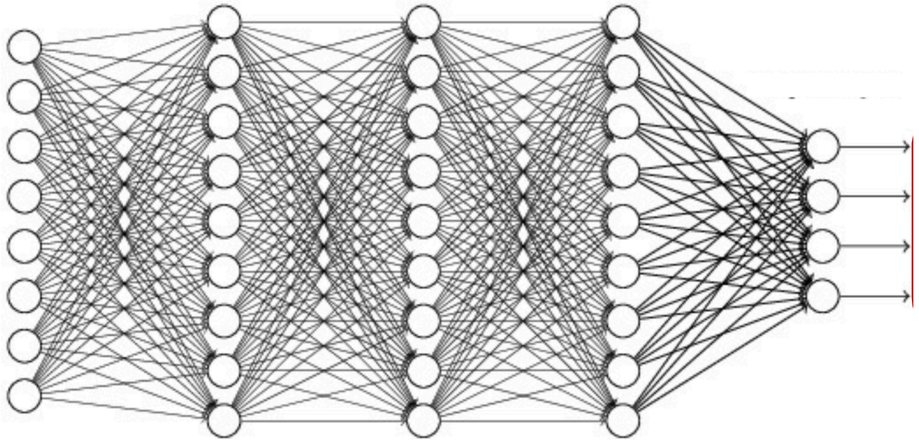
$$L(\boldsymbol{w}, b) = -\, y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

$$\frac{\partial L}{\partial \hat{y}} = -\frac{y}{\hat{y}} + \frac{1 - y}{1 - \hat{y}}\,, \quad \frac{\partial L}{\partial z} = \hat{y} - y\,, \quad \frac{\partial L}{\partial \boldsymbol{w}} = (\hat{y} - y)\,\boldsymbol{x}^T\,, \quad \frac{\partial L}{\partial b} = \hat{y} - y$$

Multiclass classification

k - classes

- Label (of class c) $\boldsymbol{y} = [0,0,\ldots,0,1,0,0,\ldots,0]$ is one hot vector with value 1 at c-th position
- Network output $f(\boldsymbol{x}) = \hat{\boldsymbol{y}} \in \mathbb{R}^k$ is probability distribution over k classes
  $\hat{\boldsymbol{y}} = \mathrm{softmax}(\boldsymbol{z})$



$$\hat{y}_i = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)}$$

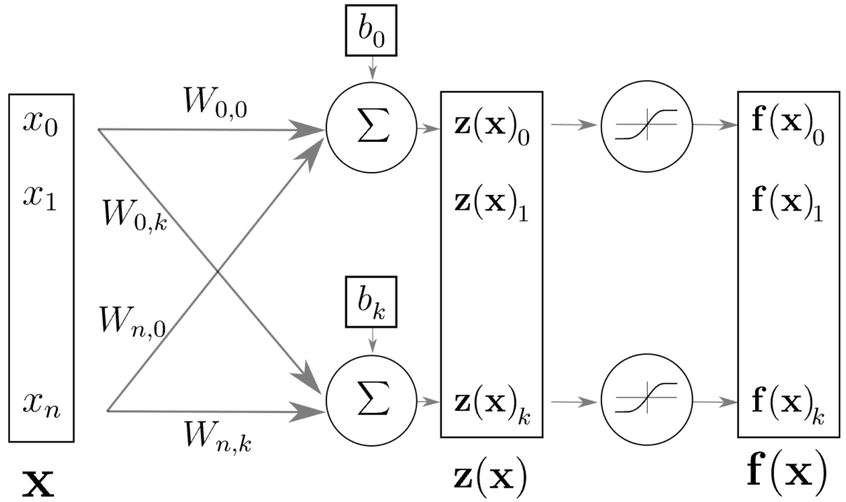
Deep Learning calculus

$$\hat{y} = \text{softmax}(z), \qquad \frac{\partial \hat{y}}{\partial z} = \text{diag}(\hat{y}) - \hat{y}\hat{y}^T$$

… to be continued next week

Forward propagation

Layer of Neurons

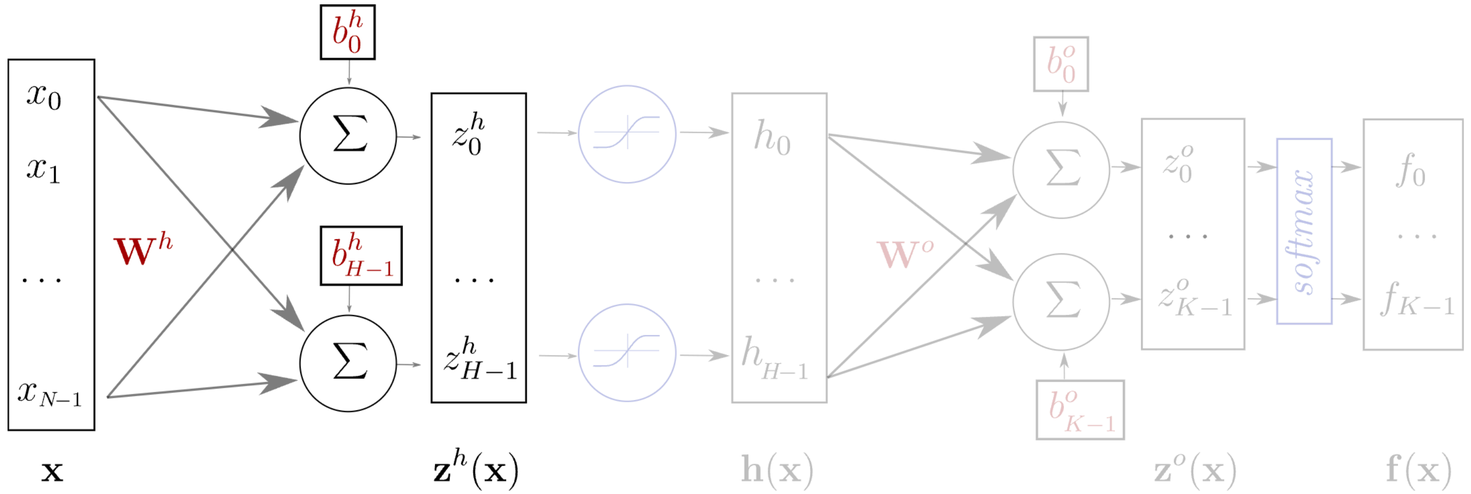


$$\mathbf{f(x)} = \mathrm{g}(\mathbf{z(x)}) = \mathrm{g}(\mathbf{Wx} + \mathbf{b})$$

- $\mathbf{W}, \mathbf{b}$ now matrix and vector

source: m2dsupsdlclass.github.io

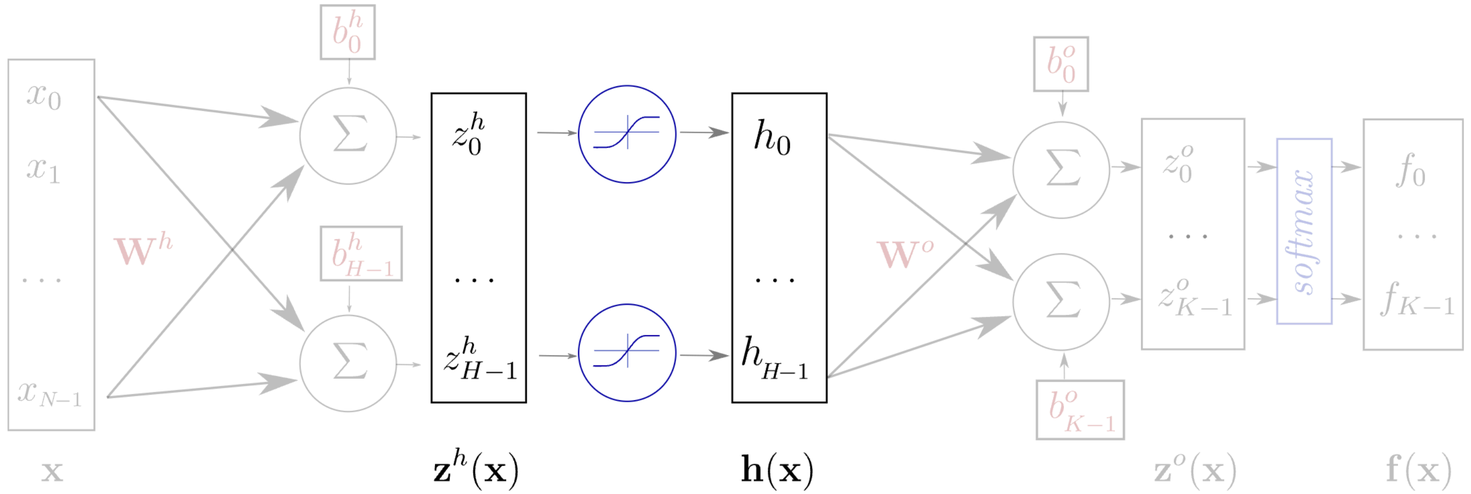
One Hidden Layer Network



- $\mathbf{z}^h(\mathbf{x}) = \mathbf{W}^h\mathbf{x} + \mathbf{b}^h$
- $\mathbf{h}(\mathbf{x}) = g(\mathbf{z}^h(\mathbf{x})) = g(\mathbf{W}^h\mathbf{x} + \mathbf{b}^h)$
- $\mathbf{z}^o(\mathbf{x}) = \mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o$
- $\mathbf{f}(\mathbf{x}) = \mathrm{softmax}(\mathbf{z}^o) = \mathrm{softmax}(\mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o)$

source: m2dsupsdlclass.github.io

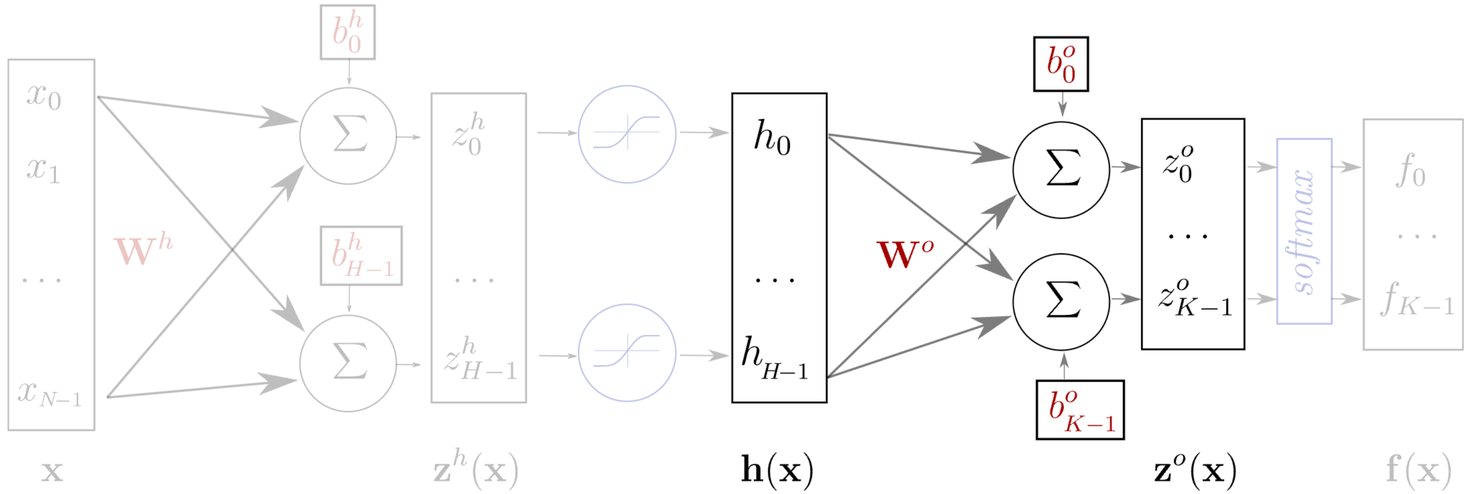
One Hidden Layer Network



- $\mathbf{z}^h(\mathbf{x}) = \mathbf{W}^h\mathbf{x} + \mathbf{b}^h$
- $\mathbf{h}(\mathbf{x}) = g(\mathbf{z}^h(\mathbf{x})) = g(\mathbf{W}^h\mathbf{x} + \mathbf{b}^h)$
- $\mathbf{z}^o(\mathbf{x}) = \mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o$
- $\mathbf{f}(\mathbf{x}) = \mathrm{softmax}(\mathbf{z}^o) = \mathrm{softmax}(\mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o)$

source: m2dsupsdlclass.github.io

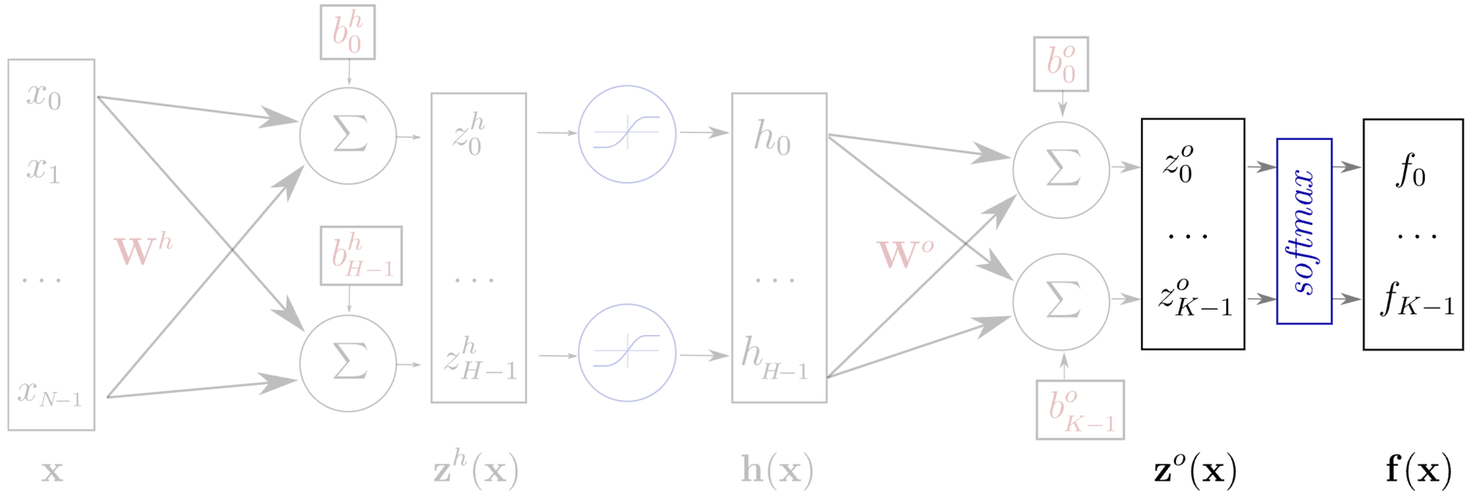
One Hidden Layer Network



- $\mathbf{z}^h(\mathbf{x}) = \mathbf{W}^h\mathbf{x} + \mathbf{b}^h$
- $\mathbf{h}(\mathbf{x}) = g(\mathbf{z}^h(\mathbf{x})) = g(\mathbf{W}^h\mathbf{x} + \mathbf{b}^h)$
- $\mathbf{z}^o(\mathbf{x}) = \mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o$
- $\mathbf{f}(\mathbf{x}) = \mathrm{softmax}(\mathbf{z}^o) = \mathrm{softmax}(\mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o)$

source: m2dsupsdlclass.github.io

One Hidden Layer Network



- $\mathbf{z}^h(\mathbf{x}) = \mathbf{W}^h\mathbf{x} + \mathbf{b}^h$
- $\mathbf{h}(\mathbf{x}) = g(\mathbf{z}^h(\mathbf{x})) = g(\mathbf{W}^h\mathbf{x} + \mathbf{b}^h)$
- $\mathbf{z}^o(\mathbf{x}) = \mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o$
- $\mathbf{f}(\mathbf{x}) = \mathrm{softmax}(\mathbf{z}^o) = \mathrm{softmax}(\mathbf{W}^o\mathbf{h}(\mathbf{x}) + \mathbf{b}^o)$

source: m2dsupsdlclass.github.io