Neural networks :

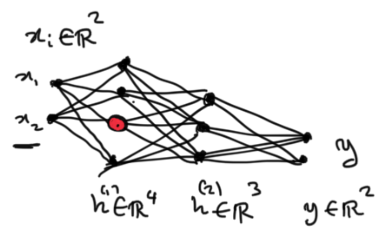
Linear models :

$$y = f\left(\sum_{i=1}^{d} \underline{w_i}\, \varphi_i(x)\right) \quad \begin{cases} \underline{f: \text{linear}}, \ \varphi: \text{identity} \to \text{Linear regression} \\ \underline{f: \text{linear}}, \ \varphi: \text{nonlinear} \to \text{with basis f.} \\ \underline{f: \text{logistic}}, \ \varphi: \text{identity} \to \text{logistic regression} \end{cases}$$

<u>Goal</u> : Make $\varphi_i(x)$ depend on trainable parameters .

Multi-layer perceptron (MLP) :

$$\boxed{y = w^T \varphi(x;\theta)}$$



$$\underbrace{h_q^{(1)}}_{\text{activations}} = f^{(1)}\Big(\underbrace{\sum_{i=1}^{d} w_{iq}^{(1)} + b_q^{(1)}}_{\text{linear}}\Big) \qquad q = 1, \dots, Q^{(1)} = 4$$

$\xRightarrow{\text{vectorize}}$

$$\mathcal{D} := \{ \underset{n\times d}{X}, \underset{n\times S}{y} \}, \quad X = \begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & & \\ x_{n1} & \cdots & x_{nd} \end{bmatrix}, \quad y = \begin{bmatrix} y_{11} & \cdots & y_{1S} \\ \vdots & & \\ y_{n1} & \cdots & y_{nS} \end{bmatrix}$$

$$\underset{n\times Q^{(1)}}{H^{(1)}} = \begin{bmatrix} h_1^{(1)} \\ \vdots \\ h_Q^{(1)} \end{bmatrix} = f^{(1)}\Big(\underbrace{\underset{n\times d}{X}\ \underset{d\times Q^{(1)}}{W^{(1)}}}_{n\times Q^{(1)}} + \underset{1\times Q^{(1)}}{b^{(1)}}\Big)$$

$$\underset{n\times Q^{(2)}}{H^{(2)}} = f^{(2)}\Big(\underset{n\times Q^{(1)}}{H^{(1)}}\ \underset{Q^{(1)}\times Q^{(2)}}{W^{(2)}} + \underset{1\times Q^{(2)}}{b^{(2)}}\Big)$$

$$\vdots \qquad\qquad \vdots$$

$$y = H^{(L)} W^{(0)} + b^{(0)}$$

X : input
$\downarrow$
$H^{(1)}$
$\downarrow$
$H^{(2)}$
$\downarrow$
$\vdots$
$H^{(L)}$
$\downarrow$
y : output

output

Trainable parameters:

$$\theta := \{ \underbrace{W^{(1)}}_{2\times 4}, \underbrace{b^{(1)}}_{1\times 4}, \underbrace{W^{(2)}}_{4\times 3}, \underbrace{b^{(2)}}_{1\times 3}, \dots, \underbrace{W^{(o)}}_{3\times 2}, \underbrace{b^{(o)}}_{1\times 2} \}$$

Network hyper-parameters:

1.) Number of hidden layers : L

2.) Number of neurons / dimensionality : $\underline{Q^{(1)}}, \dots, \underline{Q^{(L)}}$

3.) Choice of activation functions : sigmoid, tanh, ReLU, etc.

Training:

$$y = f_\theta(x) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

$$\Rightarrow \quad p(y \mid x, \theta, \sigma^2) = \mathcal{N}(y \mid f_\theta(x), \sigma^2)$$

$$\mathcal{L}(\theta, \sigma^2) := -\log p(y \mid x, \theta, \sigma^2) = \frac{1}{2\sigma^2} \sum_{i=1}^{n} \left[f_\theta(x_i) - y_i \right]^2 + \frac{n}{2} \log(2\pi\sigma^2)$$

$\downarrow$

$$\mathcal{L}(\theta) := \frac{1}{n} \sum_{i=1}^{n} \left[f_\theta(x_i) - y_i \right]^2$$

(mean squared loss)

(regression)

$$\mathcal{L}(\theta) := -\sum_{i=1}^{n} y_i \log \sigma(f_\theta(x_i)) + (1-y_i) \log(1 - \sigma(f_\theta(x_i)))$$

(classification)

θ^* ... $\mathcal{L}(\theta)$ S.G.D. ... $n+1$

$$\theta^* = \arg\min_{\theta \in \Theta} \mathcal{L}(\theta) \Longrightarrow \theta^{n+1} = \theta^n - \eta \nabla_\theta \mathcal{L}(\theta^n)$$

Automatic differentiation

$F : \mathbb{R}^d \longrightarrow \underline{\underline{\mathbb{R}}}$ (e.g. $\mathcal{L}(\theta)$)

$F : [\ ] \longrightarrow \square$, $x \in \mathbb{R}^d$, $y \in \mathbb{R}$

$$F = D \circ C \circ B \circ A$$

$$y = F(x) = D(C(B(A(x))))$$

$\hookrightarrow$ $y = D(c)$, $c = C(b)$, $b = B(a)$, $a = A(x)$

Jacobian : $$\underset{d \times 1}{\nabla_x F} = \frac{dF}{dx} = \frac{dy}{dx} = \left[\frac{\partial y}{\partial x_1}, \ldots, \frac{\partial y}{\partial x_d}\right]^T$$

Chain rule : $$\nabla_x F = \boxed{\frac{\partial y}{\partial c} \cdot \frac{\partial c}{\partial b}} \cdot \boxed{\frac{\partial b}{\partial a} \cdot \frac{\partial a}{\partial x}}$$ reverse mode $\longrightarrow$

$$\underset{1 \times d_c}{\frac{\partial y}{\partial c} = D'(c)} \qquad \frac{\partial c}{\partial b} = C'(b) \qquad \frac{\partial a}{\partial x} = A'(x)$$

$$\boxed{d_c \times d_b} \qquad d_a \times d$$

- Forward accumulation :

$$\nabla_x F = \frac{\partial y}{\partial c}\left(\left(\frac{\partial c}{\partial b}\left(\frac{\partial b}{\partial a}\left(\frac{\partial a}{\partial x}\right)\right)\right)\right)$$

use when

$\dim\{x\} << \dim\{y\}$

- <u>Reverse mode</u> :

$$\nabla_x F = \left(\left(\frac{\partial y}{\partial c} \cdot \frac{\partial c}{\partial b}\right) \frac{\partial b}{\partial a}\right) \frac{\partial a}{\partial x}$$

use $\dim\{x\} >> \dim\{y\}$

- FW - AD $\Longrightarrow$ JVPs (Jacobian-vector product)

Constructs the Jacobian one column at a time

$$\nabla_x F : \left[\frac{\partial y}{\partial x_1}, \ldots, \frac{\partial y}{\partial x_d}\right] \underset{[0,1,\ldots,0]}{\vec{v}} = \frac{\partial y}{\partial c}\left(\frac{\partial c}{\partial b}\left(\left(\frac{\partial b}{\partial a}\left(\frac{\partial a}{\partial x}\, \underline{\vec{v}}\right)\right)\right)\right)$$

- RV - AD $\Longrightarrow$ VJPs (Vector-Jacobian product)

Construct the Jacobian one row at a time :

$$\nabla_x F : \left[\frac{\partial y}{\partial x_1}, \ldots, \frac{\partial y}{\partial x_d}\right] = \left(\left(\left(v^T \frac{\partial y}{\partial c}\right) \frac{\partial c}{\partial b}\right) \frac{\partial b}{\partial a}\right) \frac{\partial a}{\partial x}$$

$\mathbb{R}^d \rightarrow \mathbb{R}$