Une brève introduction à l'Intelligence Artificielle

De l'apprentissage automatique à l'apprentissage profond - Première intuition

September 6, 2023

CentraleSupélec

De l'apprentissage automatique à l'apprentissage profond

Les réseaux de neurones

Pouvoir construire des fonctions de prédictions non-linéaires.

Linear predictors:
$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x),\ \phi(x) = [1, x]$

Non-linear (quadratic) predictors:
$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \phi(x),\ \phi(x) = [1, x, x^2]$

Non-linear neural networks:
$f_{\mathbf{w}}(x) = \mathbf{w} \cdot \sigma(\mathbf{V}\phi(x)),\ \phi(x) = [1, x]$

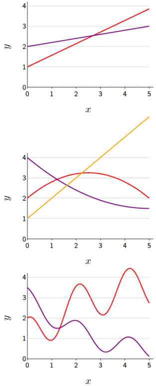


Figure: Source : CS221 Stanford

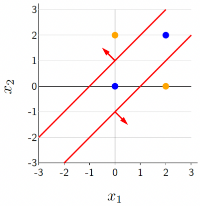
Intuition avec un exemple simple

Prédire la collision de deux voitures

- **Entrées** : positions des deux voitures (en terme de distance d'un côté de la route) : x_1 et x_2.
- **Sorties** : si les voitures sont en sécurité $(y = 1)$ ou si elles sont en collision $(y = -1)$. On considère que les voitures sont en sécurité si elles sont séparées par une distance d'au moins 1.

$$y = \text{sign}(|x_1 - x_2| - 1)$$

x_1	x_2	y
0	2	1
2	0	1
0	0	-1
2	2	-1



Frontière de décision: tous les points situés à l'intérieur de la région entre les deux lignes rouges sont négatifs et tous les points situés à l'extérieur (de part et d'autre) sont positifs.

Intuition avec un exemple simple

Décomposition du problème

Test if car 1 is far right of car 2:

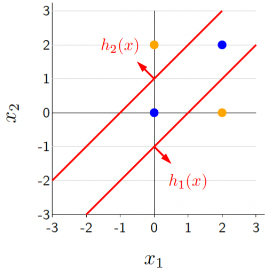
$$h_1(x) = \mathbf{1}[x_1 - x_2 \geq 1]$$

Test if car 2 is far right of car 1:

$$h_2(x) = \mathbf{1}[x_2 - x_1 \geq 1]$$

Safe if at least one is true:

$$f(x) = \text{sign}(h_1(x) + h_2(x))$$



x	$h_1(x)$	$h_2(x)$	$f(x)$
$[0,2]$	0	1	$+1$
$[2,0]$	1	0	$+1$
$[0,0]$	0	0	-1
$[2,2]$	0	0	-1

Intuition avec un exemple simple

Réécriture avec une notation vectorielle

Intermediate subproblems:

$$h_1(x) = \mathbf{1}[x_1 - x_2 \geq 1] = \mathbf{1}[[-1, +1, -1] \cdot [1, x_1, x_2] \geq 0]$$

$$h_2(x) = \mathbf{1}[x_2 - x_1 \geq 1] = \mathbf{1}[[-1, -1, +1] \cdot [1, x_1, x_2] \geq 0]$$

$$\mathbf{h}(x) = \mathbf{1}\left[\begin{bmatrix} -1 & +1 & -1 \\ -1 & -1 & +1 \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \geq 0\right]$$

Predictor:

$$f(x) = \text{sign}(h_1(x) + h_2(x)) = \text{sign}([1, 1] \cdot \mathbf{h}(x))$$

L'objectif est d'apprendre les poids en rouge.

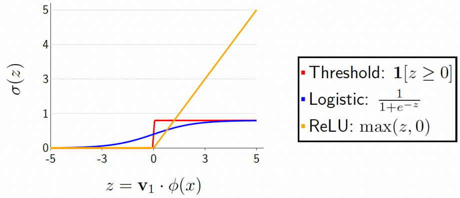
Intuition avec un exemple simple

Eviter les gradients nuls

- Problème: Le gradient de $h_1(x)$ par rapport à $\boldsymbol{v}_1$ est 0.

$$h1(x) = 1[v_1.\phi(x) \geq 0]$$

- Solution : remplacer la fonction indicatrice par une **fonction d'activation** σ avec des gradients non nuls.



$h_1(x) = \sigma(\mathbf{v}_1 \cdot \phi(x))$

Voir `https://en.wikipedia.org/wiki/Activation_function` pour différents types de fonctions d'activation

Intuition avec un exemple simple

Réseaux de neurones à 2 couches.

Intermediate subproblems:

$$\mathbf{h}(x) = \sigma(\mathbf{V}\,\phi(x))$$

Predictor (classification):

$$f_{\mathbf{V},\mathbf{w}}(x) = \mathrm{sign}(\mathbf{w} \cdot \mathbf{h}(x))$$

Interpret $\mathbf{h}(x)$ as a learned feature representation!

Hypothesis class:

$$\mathcal{F} = \{f_{\mathbf{V},\mathbf{w}} : \mathbf{V} \in \mathbb{R}^{k \times d}, \mathbf{w} \in \mathbb{R}^{k}\}$$

matrice de poids $\boldsymbol{V}$: vecteurs de poids des k sous-problèmes.

Réseaux de neurones profonds

1-layer neural network:

$$\text{score} = \mathbf{w} \cdot \phi(x)$$

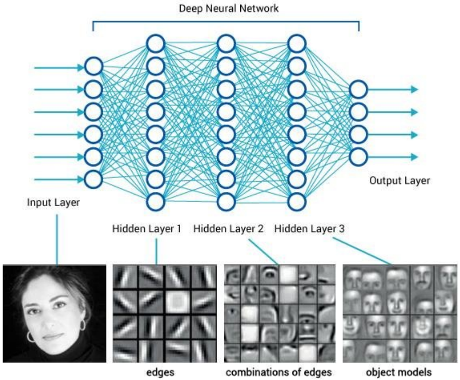
2-layer neural network:

$$\text{score} = \mathbf{w} \cdot \sigma\left(\mathbf{V} \, \phi(x) \right)$$

3-layer neural network:

$$\text{score} = \mathbf{w} \cdot \sigma\left(\mathbf{V}_2 \, \sigma\left(\mathbf{V}_1 \, \phi(x) \right) \right)$$

Réseaux de neurones profonds



Réseaux de neurones profonds

Un zoo de modèles (`https://www.asimovinstitute.org/neural-network-zoo/`)