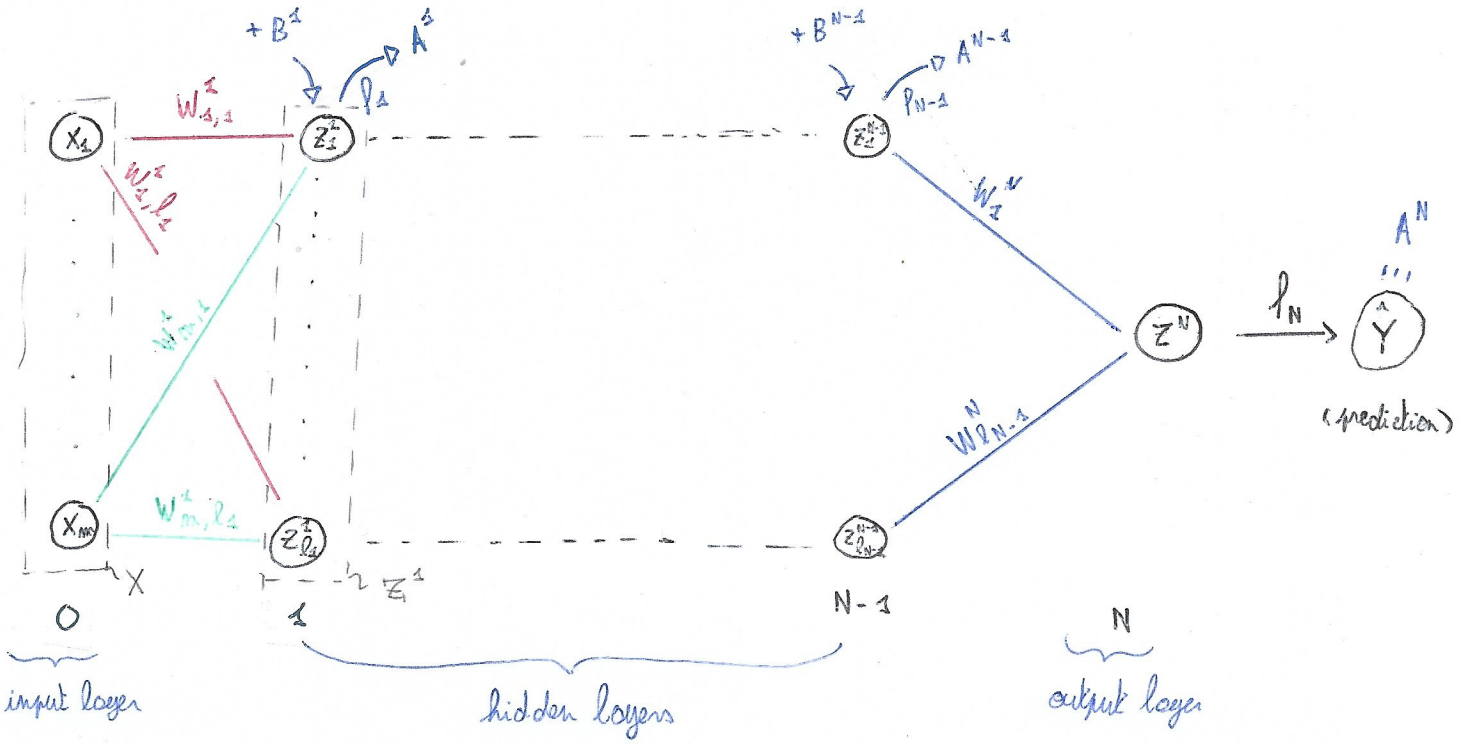
Neural Network



Notation

$X \equiv$ training input (dataset without output) $\equiv A_0$

$Y \equiv$ training output (dataset with the output only).

$\hat{Y} \equiv$ predicted output (activated output A^N associated with the N-th layer).

$W^i \equiv$ weight matrix associated with the i-th layer.

$B^i \equiv$ bias matrix associated with the i-th layer.

$$Z^i = (A^{i-1} \cdot W^i) + B^i$$

$\equiv$ output matrix associated with the i-th layer. (Linear)

$$A^i = f_i(Z^i)$$

$\equiv$ activated output associated with the i-th layer. (Non-linear)

$f_i \equiv$ activation function associated with the i-th layer.

$L \equiv$ loss function (used to optimize parameters W^i and B^i)

Examples of activation function:

- $\text{sigmoid}(x) = \dfrac{1}{1+e^{-x}}$

- $\text{ReLU}(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$

Examples of loss function:

- Mean Squared Error (MSE): $L(Y,\hat{Y}) = \dfrac{1}{2}\sum_{i=1}^{m}(Y_i - \hat{Y}_i)^2$

- Cross-Entropy: $L(Y,\hat{Y}) = -\sum_{i=1}^{m} y_i \ln(\hat{y}_i)$

Dimensions:

$0, \dots, N$: number of layers.

m : number of entries in the training dataset.

n : number of categories in the training dataset. (excepted "output category")

$\equiv$ number of inputs

$l_i \equiv$ number of nodes for the i-th layer ; $i = 1, \dots, N$

1st layer:

$\dim(X) \equiv \dim(A^0) = m \times n$

$\dim(W^1) = n \times l_1$

$\dim(X \cdot W^1) = m \times l_1 \Rightarrow \dim(B^1) = m \times l_1$

$\dim(Z^1) = m \times l_1 \Rightarrow \dim(A^1) = m \times l_1$

Any i-th layer

$\dim(A^{i-1}) = m \times l_{i-1}$ $\quad \dim(B^i) = m \times l_i$

$\dim(W^i) = l_{i-1} \times l_i \Rightarrow \dim(Z^i) = m \times l_i \Rightarrow \dim(A^i) = m \times l_i$

Neural Network

Feed Forward : $x \rightsquigarrow \cdot \rightsquigarrow \cdot \rightsquigarrow \dots \rightsquigarrow \hat{Y}$

$$Z^i = (A^{i-1} \cdot W^i) + B^i$$

$$A^i = f_i(Z^i)$$

with : $A^0 \equiv X$; $A^N \equiv \hat{Y}$.

Backpropagation : $x \dots \curvearrowleft \cdot \curvearrowleft \hat{Y}$

$$L(Y, \hat{Y}) \equiv L(Y, A^N) \in \mathbb{R}$$

intuitively : $\begin{cases} \text{if } L(Y, A^N) \text{ is small} \Rightarrow \text{good perf.} \\ \text{if } L(Y, A^N) \text{ is big} \Rightarrow \text{bad perf.} \end{cases}$

To optimize $L(Y, A^N)$ we have to compute $dL(Y, A^N) = 0$.

Error on a layer

$$\frac{\Delta L}{\Delta Z^i} \approx \frac{\partial L}{\partial Z^i}$$

$$= \frac{\partial L}{\partial A^N} \cdot \frac{\partial A^N}{\partial Z^N} = \frac{\partial L}{\partial A^N} \cdot f'(Z^N)$$

Error on bias and weight

$$\frac{\partial L}{\partial B^i} = \frac{\partial L}{\partial Z^i}$$

$$\frac{\partial L}{\partial W^i} = \frac{\partial L}{\partial Z^i} \cdot \frac{\partial Z^i}{\partial W^i}$$

$$= \frac{\partial L}{\partial Z^i} \cdot [A^{i-1}]^T$$

$$\frac{\partial L}{\partial W^N} = \frac{\partial L}{\partial A^N} \cdot \frac{\partial A^N}{\partial Z^N} \cdot \frac{\partial Z^N}{\partial W^N}$$

$$= \frac{\partial L}{\partial A^N} \cdot f_N'(Z^N) \cdot [A^{N-1}]^T$$

$$\frac{\partial L}{\partial W^{N-1}} = \left(\frac{\partial L}{\partial A^N} \cdot \frac{\partial A^N}{\partial Z^N}\right) \cdot \left(\frac{\partial Z^N}{\partial A^{N-1}} \cdot \frac{\partial A^{N-1}}{\partial Z^{N-1}}\right) \cdot \frac{\partial Z^{N-1}}{\partial W^{N-1}}$$

$$= \left(\frac{\partial L}{\partial A^N} f_N'(Z^N)\right) \cdot \left([W^N]^T \cdot f_{N-1}'(Z^{N-1})\right) \cdot [A^{N-2}]^T$$

$$\frac{\partial L}{\partial W^i} = \underbrace{\left(\frac{\partial L}{\partial A^N} f_N'(Z^N)\right) \cdot \left([W^N]^T \cdot f_{N-1}'(Z^{N-1})\right) \cdots \left([W^{i+1}] f_i'(Z^i)\right)}_{= \partial L / \partial Z^{i+1}} [A^{i-1}]^T$$

$$= \frac{\partial L}{\partial Z^i}$$

$$\frac{\partial L}{\partial B^i} = \frac{\partial L}{\partial Z^i}$$

$\odot$: Hadamar product

Summary

Error: $\Delta Z = \frac{\partial L}{\partial Z^i}$

$$= \frac{\partial L}{\partial A^N} \odot \frac{\partial A^N}{\partial Z^i}$$

$$= \frac{\partial L}{\partial A^N} \odot f_i'(Z^i)$$

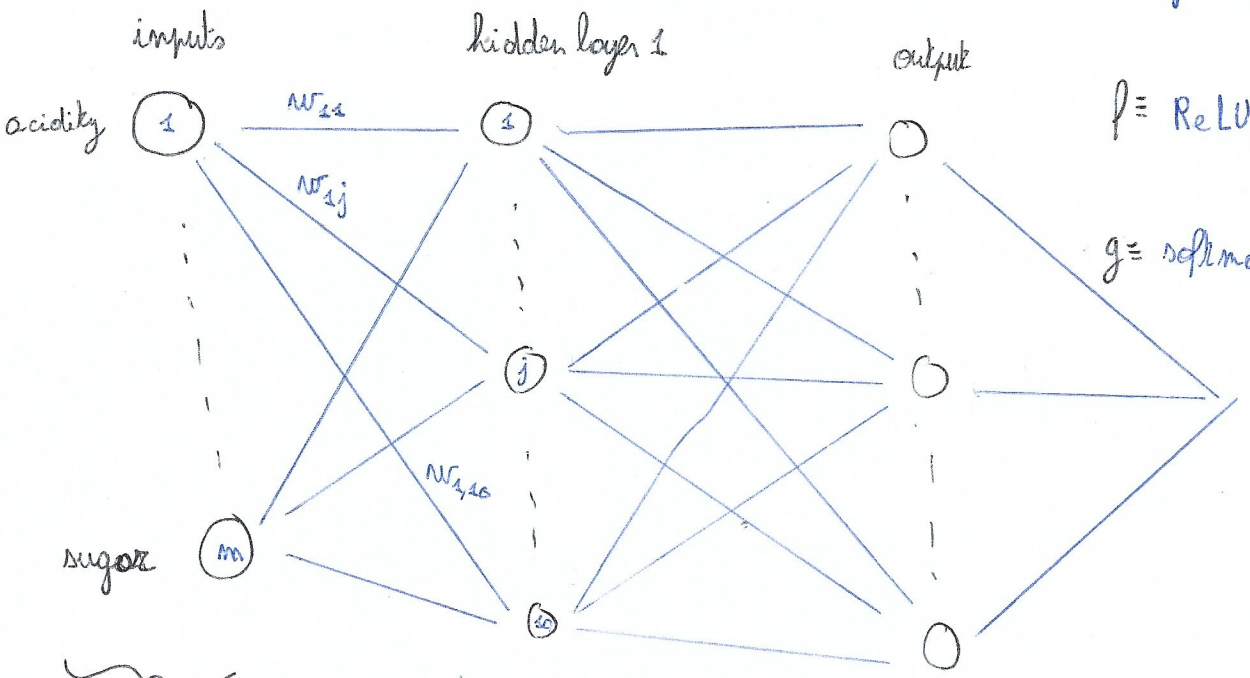
Error on parameters:

- $\frac{\partial L}{\partial W^i} = \frac{\partial L}{\partial Z^i} \cdot \frac{\partial Z^i}{\partial W^i}$

$$= \frac{\partial L}{\partial Z^i} \cdot [A^{i-1}]^T = \frac{\partial L}{\partial Z^{i+1}} \cdot [W^{i+1}] \odot f_i'(Z^i) [A^{i-1}]^T$$

- $\frac{\partial L}{\partial B^i} = \frac{\partial L}{\partial Z^i}$

Neurone



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

$$f \equiv \text{ReLU}(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

$$g \equiv \text{softmax}(x) = \frac{e^{x}}{\sum_{k} e^{x_k}}$$

11 catégories caractérisent le vin

10 noeuds (arbitraire)

10 noeuds (arbitraire)

Prédiction

training_dataset ↖ ↗ testing_dataset

⇒ n = nombre de lignes du dataset (préalablement divisé en 2)

⇒ m = nombre de catégories = nombre d'inputs

⇒ l = nombre de noeuds arbitraires

Feed Forward :

Chaque couche est calculée comme suit,

np.random.uniform

↑

random

⇓ ↘

NEXT_LAYER = PREVIOUS_LAYER . WEIGHTS + BIAS

$$\underbrace{Z^{(1)}}_{\text{hidden\_layer\_1}} = \underbrace{X}_{\text{inputs}} \cdot \underbrace{W^{(1)}}_{\text{weights}} + \underbrace{B^{(1)}}_{\text{bias}}$$

$\underbrace{X}_{\text{inputs}} \in \mathbb{R}^{n \times m}$, $\underbrace{W^{(1)}}_{\text{weights}} \in \mathbb{R}^{m \times l}$, $\underbrace{B^{(1)}}_{\text{bias}} \in \mathbb{R}^{n \times l}$, $Z^{(1)} \in \mathbb{R}^{n \times l}$

linéaire ⟶ non-linéaire

⟹ $f(\text{hidden\_layer\_1}) \equiv f(Z^{(1)})$

↳ sigmoïd for hidden layers.
ReLU

$$\underbrace{Z^{(2)}}_{\text{output\_layer}} = \underbrace{f(Z^{(1)})}_{\text{hidden\_layer\_1}} \cdot \underbrace{W^{(2)}}_{\text{weights}} + \underbrace{B^{(2)}}_{\text{bias}}$$

$f(Z^{(1)}) \in \mathbb{R}^{n \times l}$, $W^{(2)} \in \mathbb{R}^{l \times l}$, $B^{(2)} \in \mathbb{R}^{n \times l}$

⟹ $g(\text{output\_layer}) \equiv g(Z^{(2)})$

↳ softmax

$\hat{y}$

Backpropagation :

on cherche à minimiser l'erreur de la valeur prédite ($\hat{y}$) par rapport à la vraie valeur (y).

On définit pour cela une "loss-function",

$$L(y, \hat{y}) := -\sum_i y_i \cdot \ln(\hat{y}_i)$$

La ressemble à l'entropie statistique,

$$S[p] = -k_B \sum_i p_i \ln(p_i)$$

Elle permet de calculer l'erreur de la prédiction ($\hat{y}$) par rapport à la vraie valeur (y).
Minimiser cette erreur consiste en un problème d'optimisation,

$$dL = \left(\frac{\partial L}{\partial W} \quad \frac{\partial L}{\partial B} \right) = 0$$

On sait que $\hat{y} = X \cdot W + B$

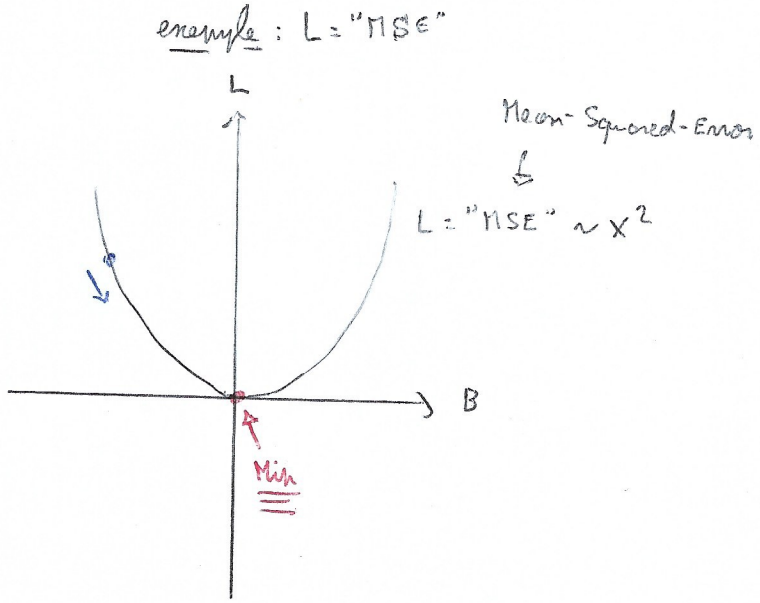
$$\begin{cases} \frac{\partial L}{\partial W} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial W} = -\sum_i y_i \cdot \frac{1}{\hat{y}_i} \cdot X_i = 0 \\ \frac{\partial L}{\partial B} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial B} = -\sum_i y_i \cdot \frac{1}{\hat{y}_i} \cdot 1 = 0 \end{cases}$$

On peut aussi utiliser MSE : $L(y_i, \hat{y}_i) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$

$= \frac{1}{n}$

Gradient descent

Afin de trouver la valeur minimale de la "fonction de Loss", de manière non-analytique. On utilise un algorithme appelé la descente de gradient.



On va petit à petit tweaker les paramètres B et W afin d'atteindre le minimum.

$$\begin{cases} W = W - \frac{\partial L}{\partial W} \cdot \text{"learning-rate"} \\ \qquad\qquad \text{↳ "vitesse / précision de l'algorithme"} \\ B = B - \frac{\partial L}{\partial B} \cdot \text{"learning-rate"} \end{cases}$$

$$\triangleright \frac{\partial L}{\partial W^{(2)}} = -\sum_i \frac{\partial}{\partial W^{(2)}} \left(y_i \ln(g \circ z^{(2)}) \right) \quad \text{avec } g \circ z^{(2)} = g \circ ((f \circ z^{(1)}) \cdot W^{(2)} + B^{(2)})$$

$$= -\sum_i \frac{\partial}{\partial W^{(2)}} \left(y_i \ln(g \circ ((f \circ z^{(1)}) \cdot W^{(2)} + B^{(2)})) \right)$$

$$= -\sum_i y_i \cdot \frac{1}{g \circ z^{(2)}} \cdot f \circ z^{(1)} = -Y \cdot \frac{1}{g(z^{(2)})} \cdot f(z^{(1)})$$

▷ $\frac{\partial L}{\partial B^{(2)}} = -Y \cdot \frac{1}{g(Z^{(2)})}$

▷ $\frac{\partial L}{\partial W^{(1)}} = -Y \cdot \frac{1}{g(Z^{(2)})} \cdot \frac{\partial (g \circ Z^{(2)})}{\partial W^{(1)}}$

Or, $\frac{\partial (g \circ Z^{(2)})}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left(g\left(f(Z^{(1)}) \cdot W^{(2)} + B^{(2)} \right) \right)$

$= g'\left(f(Z^{(1)}) \cdot W^{(2)} + B^{(2)} \right) \cdot W^{(2)} f'(Z^{(1)}) \cdot \frac{\partial Z^{(1)}}{\partial W^{(1)}}$

$= g'(Z^{(2)}) \cdot W^{(2)} f'(Z^{(1)}) \cdot X$

$\Rightarrow$▷ $\frac{\partial L}{\partial W^{(1)}} = -Y \cdot \frac{1}{g(Z^{(2)})} \cdot g'(Z^{(2)}) \cdot W^{(2)} f'(Z^{(1)}) \cdot X$

▷ $\frac{\partial L}{\partial B^{(1)}} = -Y \cdot \frac{1}{g(Z^{(2)})} \cdot g'(Z^{(2)}) \cdot W^{(2)} f'(Z^{(1)}) \cdot 1$

- $g'(x) = \frac{e^x \cdot \sum_k e^{x_k} - e^x \cdot \sum_k e^{x_k} \cdot d(x_k)/dx}{\left(\sum_k e^{x_k} \right)^2}$

- $f'(x) = \begin{cases} 1, & x > 0 \\ 0, & x \le 0 \end{cases}$