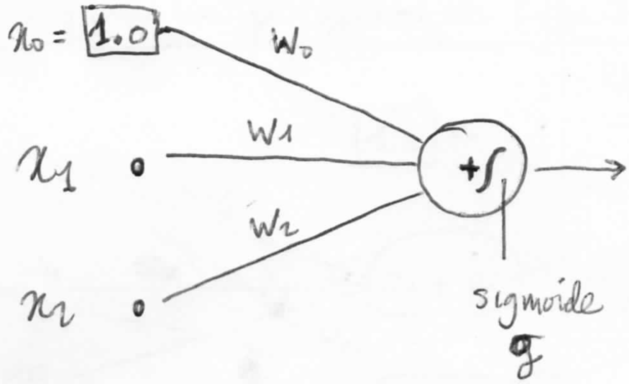
II. Rétropropagation pour un neurone seul



prédiction vrai

$$\hat{y}^{(k)} \neq y^{(k)}$$

$$\hat{y}^{(k)} = g\Big(\underbrace{w_0 x_0^{(k)} + w_1 x_1^{(k)} + w_2 x_2^{(k)}}_{X^{(k)}}\Big).$$

A) si erreur $= E(k) = \left(\hat{y}^{(k)} - y^{(k)}\right)^2$ $\qquad E_{totale} = \frac{1}{P}\sum_{k=0}^{P-1}\left(\hat{y}^{(k)} - y^{(k)}\right)^2$

$i = 0, 1, 2.$

$$\frac{\partial E^{(k)}}{\partial w_i} = 2 \cdot g'(X^{(k)}) \cdot \left[\hat{y}^{(k)} - y^{(k)}\right] \times x_i^{(k)}$$

$$= 2\, g \times (1-g) \times (\hat{y}^{(k)} - y^{(k)}) \times \begin{cases} 1 \longrightarrow \partial E^{(k)}/\partial w_0 \\ x_1 \longrightarrow \partial E^{(k)}/\partial w_1 \\ x_2 \longrightarrow \partial E^{(k)}/\partial w_2 \end{cases}$$

$$\Delta w_i = -\alpha \frac{\partial E}{\partial w_i}$$

$$= -\alpha \times \frac{2}{P} \sum_{k=0}^{P-1} g^{(k)} \times \left[1 - g^{(k)}\right] \times (\hat{y}^{(k)} - y^{(k)}) \times \begin{cases} 1 & \text{pour } \Delta w_0 \\ x_1^{(k)} & \text{pour } \Delta w_1 \\ x_2^{(k)} & \text{pour } \Delta w_2. \end{cases}$$

B) si Erreur $= C(k) = -\left[y^{(k)} \operatorname{Log} \hat{y}^{k} + (1 - y^{(k)}) \operatorname{Log}(1 - \hat{y}^{(k)})\right]$

$$\frac{\partial C^{(k)}}{\partial w_i} = -\left[y^{(k)} \frac{g'}{\hat{y}^{(k)}} x_i + (1 - y^{(k)}) \frac{-g'}{1 - \hat{y}^{(k)}} x_i\right]$$

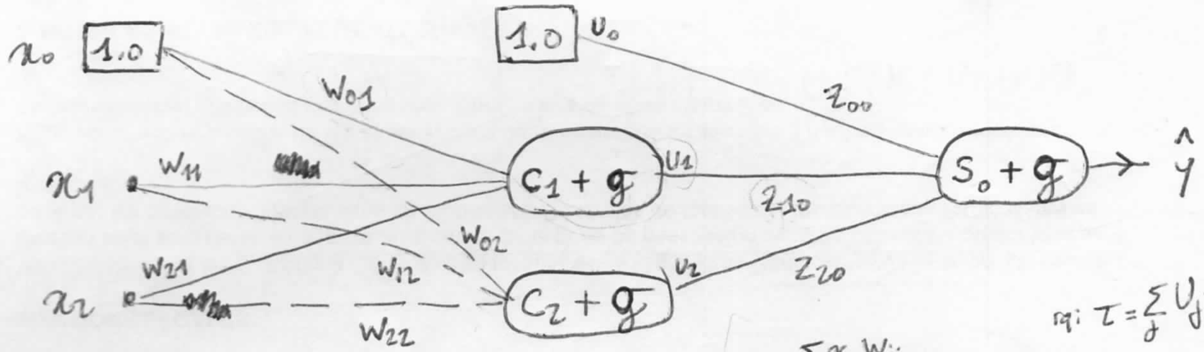
$$= -g'(X^{(k)}) \times x_i^{(k)} \left[\frac{y^{(k)}}{\hat{y}^{(k)}} - \frac{(1 - y^{(k)})}{(1 - \hat{y}^{(k)})}\right]$$

$$= -g'(X^{(k)}) \cdot x_i^{(k)} \frac{y^{(k)}(1 - \hat{y}^{(k)}) - \hat{y}^{(k)}(1 - y^{(k)})}{\hat{y}^{(k)}(1 - \hat{y}^{(k)})}$$

$$= (-1) \underbrace{\frac{g'(X^{(k)})}{\hat{y}^{(k)}(1 - \hat{y}^{(k)})}}_{= 1 \text{ si } g = \text{sigmoïde}} \times x_i^{(k)} \left(y^{(k)} - \hat{y}^{(k)}\right) = x_i^{(k)}\left[-y^{(k)} + \hat{y}^{(k)}\right].$$

$$\Delta w_i = -\alpha\, \partial C/\partial w_i = -\alpha \sum_k \left(\hat{y}^{(k)} - y^{(k)}\right) x_i^k = -\alpha \sum_k \left(\hat{y}^{(k)} - y^{(k)}\right) \times \begin{cases} 1 \rightarrow \Delta w_0 \\ x_1^k \rightarrow \Delta w_1 \\ x_2^k \rightarrow \Delta w_2 \end{cases}$$

Rétropropagation pour un perceptron de type XOR.



rq: $\sigma_i = \sum_j x_j w_{ji}$

$U_0 = 1$

$U_1 = g\left(\sum_i x_i w_{i1}\right) = g(\sigma_1)$

$U_2 = g\left(\sum_i x_i w_{i2}\right) = g(\sigma_2)$

rq: $\tau = \sum_j U_j z_{j0}$

$\hat{y} = g\left(\sum_j U_j z_{j0}\right) = g(\tau)$

g = sigmoïde

1) rétropropagation sur les z_{00}, z_{10}, z_{20} (corrections à effectuer)

A) si $E^{(k)} = (\hat{y}^{(k)} - y^{(k)})^2$, un calcul identique à celui de page 1 (rétropropagation sur un neurone seul) en substituant U_i à x_i, donne :

$$\Delta Z_{i\phi} = -\alpha \times \frac{2}{P} \sum_k \underbrace{g\left(\textstyle\sum_j U_j z_{j0}\right)\left[1 - g\left(\textstyle\sum U_j z_{j0}\right)\right]}_{\hat{y}^{(k)}(1-\hat{y}^{(k)})} U_i^{(k)} \times (\hat{y}^{(k)} - y^{(k)})$$

$$= -\frac{2\alpha}{P} \sum_k \hat{y}^{(k)} \times \left[1 - \hat{y}^{(k)}\right] U_i^{(k)} = -\frac{2\alpha}{P} \sum_k \hat{y}^{(k)}\left[1 - \hat{y}^{(k)}\right] \begin{Bmatrix} 1 \\ U_1^{(k)} \\ U_2^{(k)} \end{Bmatrix} \boxed{\times (\hat{y}^{(k)} - y^{(k)})}$$

B) si $C^{(k)}$ = logloss, un calcul identique à la page 1 donne :

$$\Delta Z_{i\phi} = -\alpha \sum_k (\hat{y}^{(k)} - y^{(k)})\, U_i^{(k)} = -\alpha \sum_k (\hat{y}^{(k)} - y^{(k)}) \times \begin{Bmatrix} 1 \\ U_1^{(k)} \\ U_2^{(k)} \end{Bmatrix}$$

2) rétropropagation de l'erreur sur U_1 et U_2 (si $C^{(k)}$ = logloss)

$$\frac{\partial C^{(k)}}{\partial U_1} = \frac{\partial C^{(k)}}{\partial \hat{y}^{(k)}} \times \frac{\partial \hat{y}^{(k)}}{\partial U_1}$$

avec : $\dfrac{\partial \hat{y}^{(k)}}{\partial U_1} = g'\left(\sum U_j z_{j0}\right) \times z_{10} = \hat{y}^{(k)} \times (1 - \hat{y}^{(k)}) . z_{10}$

$$\frac{\partial C^{(k)}}{\partial \hat{y}^{(k)}} = -y^{(k)} \frac{1}{\hat{y}^{(k)}} - (1 - y^{(k)}) \times \frac{(-1)}{1 - \hat{y}^{(k)}}$$

$$= -\left[\frac{y^{(k)}}{\hat{y}^{(k)}} - \frac{(1 - y^{(k)})}{(1 - \hat{y}^{(k)})}\right]$$

$$= -\frac{1}{\hat{y}^{(k)}(1 - \hat{y}^{(k)})}\left[y^{(k)}(1 - \hat{y}^{(k)}) - \hat{y}^{(k)}(1 - y^{(k)})\right]$$

$$= \left[\hat{y}^{(k)} - y^{(k)}\right] / \left[\hat{y}^{(k)}(1 - \hat{y}^{(k)})\right]$$

$$= \frac{\left[\hat{y}^{(k)} - y^{(k)}\right]}{\cancel{\hat{y}^{(k)}(1 - \hat{y}^{(k)})}} \times \cancel{\hat{y}^{(k)}(1 - \hat{y}^{(k)})}\, z_{10}$$

$$= \left[\hat{y}^{(k)} - y^{(k)}\right] z_{10}$$

De même :

$$\frac{\partial C^{(k)}}{\partial U_2} = \left[\hat{y}^{(k)} - y^{(k)}\right] z_{20}$$

Donc :

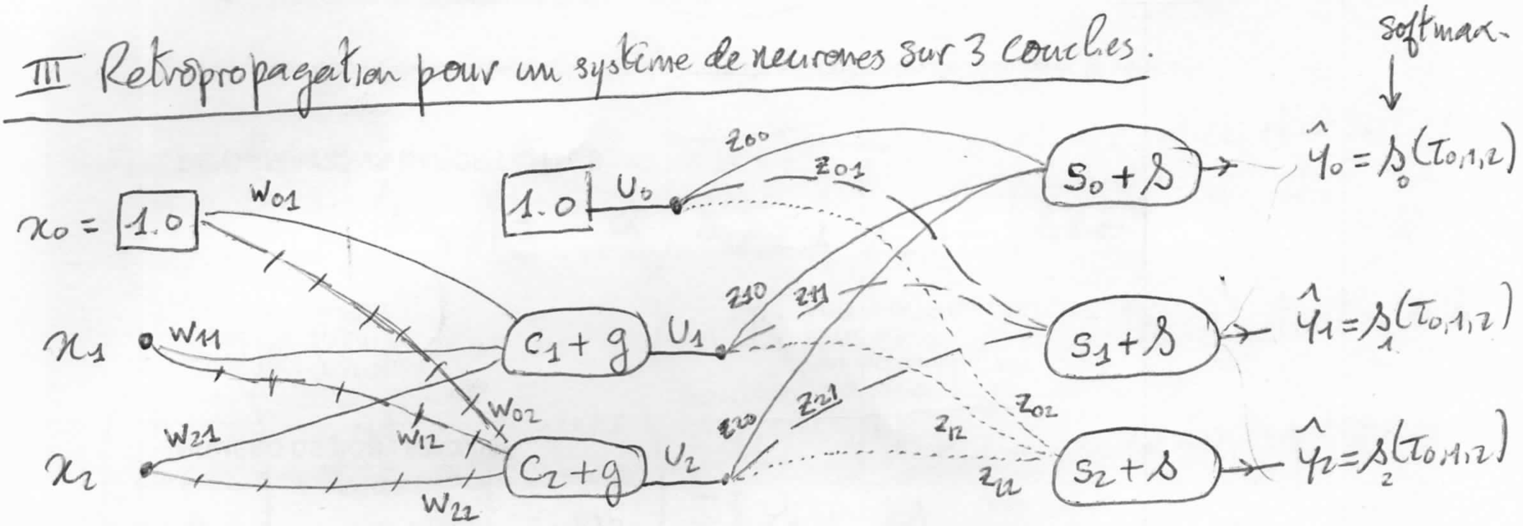
$$\frac{\partial C^{(k)}}{\partial \sigma_1} = \frac{\partial C^{(k)}}{\partial U_1} \times \frac{\partial U_1}{\partial \sigma_1} = \left[\hat{y}^{(k)} - y^{(k)}\right] z_{10} \times U_1 \times (1 - U_1)$$ car g = sigmoïde donc $g'(\sigma_1) = U_1(1 - U_1)$

$$\frac{\partial C^{(k)}}{\partial \sigma_2} = \frac{\partial C^{(k)}}{\partial U_2} \times \frac{\partial U_2}{\partial \sigma_2} = \left[\hat{y}^{(k)} - y^{(k)}\right] z_{20} \times U_2(1 - U_2)$$ " " " $g'(\sigma_2) = U_2(1 - U_2)$

Donc : $\dfrac{\partial C^{(k)}}{\partial w_{i1}} = \dfrac{\partial C^{(k)}}{\partial \sigma_1} \times \dfrac{\partial \sigma_1}{\partial w_{i1}} = [\hat{y}^{(k)} - y^{(k)}]\, z_{10} \times U_1 (1 - U_1) \times x_i^{(k)}$.

$$\frac{\partial C^{(k)}}{\partial w_{i2}} = \frac{\partial C^{(k)}}{\partial \sigma_2} \times \frac{\partial \sigma_2}{\partial w_{i2}} = [\hat{y}^{(k)} - y^{(k)}]\, z_{20} \times U_2 (1 - U_2) \times x_i^{(k)}.$$

III Retropropagation pour un système de neurones sur 3 couches.



$U_0 = 1.0$

$U_1 = g(\sigma_1)$

$\sigma_1 = \sum_i w_{i1} \times x_i$

$U_2 = g(\sigma_2)$

$\sigma_2 = \sum_i x_i w_{i2}$

$g =$ sigmoïde

$\tau_0 = \sum_j U_j z_{j0}$

$\tau_1 = \sum_j U_j z_{j1}$

$\tau_2 = \sum_j U_j z_{j2}$

$\hat{y}_0 = e^{\tau_0} / (e^{\tau_0} + e^{\tau_1} + e^{\tau_2})$.

$\hat{y}_1 = e^{\tau_1} / (e^{\tau_0} + e^{\tau_1} + e^{\tau_2})$

$\hat{y}_2 = e^{\tau_2} / (e^{\tau_0} + e^{\tau_1} + e^{\tau_2})$

rq : $\hat{y}_0 + \hat{y}_1 + \hat{y}_2 = 1$ toujours

Fonction de coût : log loss

$$C^{(k)} = -\sum_{i=0}^{2} y_i^{(k)} \operatorname{Log}(\hat{y}_i^{(k)})$$

$$= -y_0^{(k)} \operatorname{Log}(\hat{y}_0^{(k)}) - y_1^{(k)} \operatorname{Log}(\hat{y}_1^{(k)}) - y_2^{(k)} \operatorname{Log}(\hat{y}_2^{(k)})$$

rq : vraie valeur :

$(y_0, y_1, y_2) = (1, 0, 0)$ si classe ϕ

$= (0, 1, 0)$ si classe 1

$= (0, 0, 1)$ si classe 2

donc : $y_0 + y_1 + y_2 = 1$

Donc : $\dfrac{\partial C^{(k)}}{\partial \hat{y}_i} = -y_i^{(k)} \times \dfrac{1}{\hat{y}_i^{(k)}}$, $i = 0, 1, 2$.

Puis : $\dfrac{\partial C^{(k)}}{\partial \tau_i} = \dfrac{\partial C^{(k)}}{\partial \hat{y}_0} \times \dfrac{\partial \hat{y}_0}{\partial \tau_i} + \dfrac{\partial C^{(k)}}{\partial \hat{y}_1} \times \dfrac{\partial \hat{y}_1}{\partial \tau_i} + \dfrac{\partial C^{(k)}}{\partial \hat{y}_2} \times \dfrac{\partial \hat{y}_2}{\partial \tau_i}$.

et il faut calculer les $\partial \hat{y}_j / \partial \tau_i$, $j = 0, 1, 2$, $i = 0, 1, 2$.

Synthese du II (apprentissage du perceptron XOR) [cf p2 et 3]

Pour chaque echantillon "k" du jeu de données :

1) entree : $x_0^{(k)} = 1$ $\longrightarrow$ sortie : $\hat{y}_k$
   $x_1^{(k)}$ = longueur pétale (rq : y_k vraie valeur)
   $x_2^{(k)}$ = largeur pétale

2) calcul $\delta_{sortie}^{(k)} = \hat{y}_k - y_k$

3) . calcul correction sur z_{00}, z_{10}, z_{20} (pds entre couches cachee et de sortie)

$$\left(\Delta z_{i\phi}\right)^{(k)} = -\alpha\, \delta_{sortie}^{(k)} \begin{pmatrix} 1 \\ U_1^{(k)} \\ U_2^{(k)} \end{pmatrix} = \begin{pmatrix} \Delta z_{00} \\ \Delta z_{10} \\ \Delta z_{20} \end{pmatrix}$$

. calcul de l'erreur sur les σ_i (avant application de la fonction d'activation de la couche cachee).

$$\begin{pmatrix} \frac{\partial C}{\partial \sigma_1} \\ \partial C / \partial \sigma_2 \end{pmatrix} = \overrightarrow{\delta_{cache}^{(k)}} = \begin{pmatrix} \delta_{caché}^{(k)1} \\ \delta_{caché}^{(k)2} \end{pmatrix} = \begin{pmatrix} (\hat{y}^{(k)} - y^{(k)})\, z_{10}\, U_1 (1 - U_1) \\ (\hat{y}^{(k)} - y^{(k)})\, z_{20}\, U_2 (1 - U_2) \end{pmatrix}$$

4) . calcul de correction sur les poids w_{ij} (entre couche d'entree et couche cachee)

$$\frac{\partial C}{\partial w_{i1}} = \delta_{caché}^{(k)1} \times x_i^{(k)} \qquad \frac{\partial C}{\partial w_{i2}} = \delta_{caché}^{(k)2} \times x_i^{(k)} \qquad (i = 0, 1, 2)$$

$$\left(\Delta w_{i1}\right)^{(k)} = -\alpha\, \delta_{caché}^{(k)1} \times x_i^{(k)} \qquad \left(\Delta w_{i2}\right)^{(k)} = -\alpha\, \delta_{caché}^{(k)2}\, x_i^{(k)}$$

$$\frac{\partial \hat{y}_0}{\partial \tau_0} = \frac{e^{\tau_0}(e^{\tau_0}+e^{\tau_1}+e^{\tau_2}) - e^{\tau_0}e^{\tau_0}}{(\;''\quad ''\quad ''\;)^2} = \frac{e^{\tau_0}(e^{\tau_1}+e^{\tau_2})}{(\;''\quad ''\quad ''\;)^2} = \frac{e^{\tau_0}}{(e^{\tau_0}+e^{\tau_1}+e^{\tau_2})} - \left(\frac{e^{\tau_0}}{e^{\tau_0}+e^{\tau_1}+e^{\tau_2}}\right)^2$$

$$= \hat{y}_0(1-\hat{y}_0)$$

$$\frac{\partial \hat{y}_0}{\partial \tau_1} = \frac{e^{\tau_0}(-e^{\tau_1})}{(e^{\tau_0}+e^{\tau_1}+e^{\tau_2})^2} = -\hat{y}_0 \cdot \hat{y}_1 \quad \text{et} \quad \frac{\partial \hat{y}_0}{\partial \tau_2} = -\hat{y}_0\,\hat{y}_2$$

De même : $\frac{\partial \hat{y}_1}{\partial \tau_0} = -\hat{y}_1\,\hat{y}_0$, $\frac{\partial \hat{y}_1}{\partial \tau_1} = \hat{y}_1(1-\hat{y}_1)$, $\frac{\partial \hat{y}_1}{\partial \tau_2} = -\hat{y}_1 \cdot \hat{y}_2$

et $\frac{\partial \hat{y}_2}{\partial \tau_0} = -\hat{y}_2\,\hat{y}_0$, $\frac{\partial \hat{y}_2}{\partial \tau_1} = -\hat{y}_1\,\hat{y}_2$, $\frac{\partial \hat{y}_2}{\partial \tau_2} = \hat{y}_2(1-\hat{y}_2)$

En synthèse.

$$\frac{\partial \hat{y}_i}{\partial \tau_j} = \begin{cases} \hat{y}_i(1-\hat{y}_i) & \text{si } i=j \\ -\hat{y}_i \cdot \hat{y}_j & \text{si } i \neq j \end{cases}$$

Donc :
$$\frac{\partial C^{(k)}}{\partial \tau_0} = \frac{\partial C^{(k)}}{\partial \hat{y}_0}\cdot\frac{\partial \hat{y}_0}{\partial \tau_0} + \frac{\partial C^{(k)}}{\partial \hat{y}_1}\cdot\frac{\partial \hat{y}_1}{\partial \tau_0} + \frac{\partial C^{(k)}}{\partial \hat{y}_2}\cdot\frac{\partial \hat{y}_2}{\partial \tau_0}$$

$$= -\frac{y_0}{\hat{y}_0}\cdot \hat{y}_0(1-\hat{y}_0) - \frac{y_1}{\hat{y}_1}\cdot(-\hat{y}_1\cdot\hat{y}_0) - \frac{y_2}{\hat{y}_2}(-\hat{y}_2\,\hat{y}_0)$$

$$= -y_0(1-\hat{y}_0) + y_1\hat{y}_0 + y_2\hat{y}_0 = -y_0 + \hat{y}_0\underbrace{(y_0+y_1+y_2)}_{=1 \text{ car somme de probabilités}}$$

$$= \hat{y}_0 - y_0 .$$

De même : $\partial C^{(k)}/\partial \tau_1 = \hat{y}_1 - y_1$ $\qquad \frac{\partial C^{(k)}}{\partial \tau_2} = \hat{y}_2 - y_2 .$

On peut alors calculer les dérivées partielles de la fonction de coût par rapport à tous les poids w_{ij}, z_{jk}, ainsi que celles par rapport à U_0, U_1, U_2 (rétropropagation des erreurs) (↳ correction par descente de gradient)

$$\cancel{\frac{\partial C^{(p)}}{\partial z_{jk}} = \frac{\partial C^{(p)}}{\partial \hat{y}_k} \times \frac{\partial \hat{y}_k}{\partial z_{jk}}}$$ ~~car z_{jk} n'intervient que dans le calcul de $\hat{y}_k$ et pas les autres~~

$$\cancel{= \frac{\partial C^{(p)}}{\partial \hat{y}_k} \times \frac{\partial \hat{y}_k}{\partial \tau_k} \times \frac{\partial \tau_k}{\partial z_{jk}}}$$ ~~car z_{jk} n'intervient que dans le calcul de τ_k et pas les autres~~

$$\cancel{= -\frac{y_k}{\hat{y}_k} \times \hat{y}_k(1-\hat{y}_k) \times U_j}$$

$$\cancel{= -y_k(1-\hat{y}_k)\,U_j}$$

~~ce qui donne la correction à effectuer sur le poids z_{jk} ;~~

$$\cancel{\Delta z_{jk} = -\alpha \sum_p \frac{\partial C^{(p)}}{\partial z_{jk}} = -\alpha \sum_p \left[-y_k^{(p)}(1-\hat{y}_k^{(p)})\,U_j^{(p)}\right].}$$

$$\frac{\partial C^{(p)}}{\partial z_{jh}} = \frac{\partial C^{(p)}}{\partial \tau_k} \times \frac{\partial \tau_k}{\partial z_{jh}}$$ car z_{jh} n'intervient que dans le calcul de τ_k et pas les autres.

$$= (\hat{y}_k - y_k) \times U_j$$

ce qui donne la correction à effectuer sur le poids z_{jh}:

$$\Delta z_{jh} = -\alpha \sum_p \frac{\partial C^{(p)}}{\partial z_{jh}} = -\alpha \sum_p (\hat{y}_k - y_k) \times U_j.$$

La rétropropagation des erreurs commises en sortie de réseau vers les erreurs en couche cachée (sur U_1, U_2) sont: $\partial C^{(p)}/\partial U_j$, $j = 1, 2$.

$$\frac{\partial C^{(p)}}{\partial U_j} = \frac{\partial C^{(p)}}{\partial \tau_0} \times \frac{\partial \tau_0}{\partial U_j} + \frac{\partial C^{(p)}}{\partial \tau_1} \cdot \frac{\partial \tau_1}{\partial U_j} + \frac{\partial C^{(p)}}{\partial \tau_2} \times \frac{\partial \tau_2}{\partial U_j}$$

$$= (\hat{y}_0 - y_0) \times z_{j0} + (\hat{y}_1 - y_1) z_{j1} + (\hat{y}_2 - y_2) z_{j2}.$$

De plus: $$\frac{\partial U_j}{\partial w_{ij}} = \underset{\substack{\uparrow \\ \text{sigmoïde}}}{g'(\sigma_j)} \times \frac{\partial \sigma_j}{\partial w_{ij}} = U_j(1 - U_j) \times x_i$$ car sigmoïde: $g'(\sigma_j) = U_j(1-U_j)$

Donc: $$\frac{\partial C^{(p)}}{\partial w_{ij}} = \frac{\partial C^{(p)}}{\partial U_j} \times \frac{\partial U_j}{\partial w_{ij}}$$ car w_{ij} n'intervient que dans le calcul de U_j et pas les autres

$$= \left[(\hat{y}_0 - y_0) z_{j0} + (\hat{y}_1 - y_1) z_{j1} + (\hat{y}_2 - y_2) z_{j2} \right] U_j (1 - U_j)\, x_i = \frac{\partial C^{(p)}}{\partial U_j} \times U_j(1-U_j) x_i$$

La correction à effectuer sur w_{ij} est alors.

$$\Delta w_{ij} = -\alpha \sum_p \frac{\partial C^{(p)}}{\partial w_{ij}} = -\alpha \sum_p \frac{\partial C^{(p)}}{\partial U_j} \cdot U_j (1 - U_j)\, x_i$$

<u>En synthèse</u>:

a) appelons $\delta_k^{(p)\,\text{sortie}} = \hat{y}_k^{(p)} - y_k^{(p)}$ le vecteur erreur produite en couche de sortie pour l'échantillon (p). rq: f.activation = softmax.

b) la correction à appliquer aux poids z_{jk} reliant la couche de sortie et la couche cachée est:

$$\Delta z_{jk} = -\alpha \sum_{(p)} \delta_k^{(p)\,\text{sortie}} \times U_j^{(p)}$$

c) la rétropropagation des erreurs de la couche de sortie vers la couche cachée est:

$$\delta_j^{(p)\,\text{caché}} = \frac{\partial C^{(p)}}{\partial U_j} = \sum_k \delta_k^{(p)\,\text{sortie}} z_{jk}$$ ← avant correction → si $\delta_j^{(p)\,\text{caché}} = \partial C / \partial U_j$!!

d) la correction à appliquer aux poids w_{ij} reliant la couche cachée et la couche d'entrée est: rq: f.activation = sigmoïde

$$\Delta w_{ij} = -\alpha \sum_{(p)} \delta_j^{(p)\,\text{caché}} U_j (1 - U_j)\, x_i$$

Synthèse du III (apprentissage reseau 3 couches de 3 neurones) [p 3,4,5]

Pour chaque échantillon "p" du jeu de données :

1) <u>entrée</u> $x_0^{(p)} = 1$ $\longrightarrow$ sortie : $\hat{y}_0^{(p)}, \hat{y}_1^{(p)}, \hat{y}_2^{(p)}$

$x_1^{(p)}$ = longueur petale

$x_2^{(p)}$ = largeur petale

(<u>rq:</u> $y_0^{(p)}, y_1^{(p)}, y_2^{(p)}$ = vraie valeur)

2) calcul erreur de sortie :

$$\vec{\delta}^{\,0,1,2}_{\text{sortie}} = \frac{\partial C}{\partial \tau_{0,1,2}} = \begin{pmatrix} \hat{y}_0^{(p)} - y_0^{(p)} \\ \hat{y}_1^{(p)} - y_1^{(p)} \\ \hat{y}_2^{(p)} - y_2^{(p)} \end{pmatrix} = \left(\vec{\hat{y}}^{(p)} - \vec{y}^{(p)}\right) \qquad \overset{\text{sortie}}{\delta_k^{(p)}} = \frac{\partial C^{(p)}}{\partial \tau_k}$$

3). calcul correction sur les poids z_{jk} entre couche cachee et couche sortie.

$$\Delta z_{jk}^{(p)} = -\alpha \; \overset{\text{sortie}}{\delta_k^{(p)}} \times U_j^{(p)} \;/\, (\text{Nb echantillons})$$

$\to$ pour obtenir une correction <u>moyennée</u>

k $\to$ tous les neurones de sortie.

j $\to$ tous les neurones cachés.

$\longrightarrow$. calcul de l'erreur sur les σ_i (avant application de la fonction d'activation de la couche cachée)

$$\begin{pmatrix} \partial C/\partial\sigma_1 \\ \partial C/\partial\sigma_2 \end{pmatrix} = \vec{\delta}^{(p)}_{\text{caché}} = \begin{pmatrix} \overset{\text{caché}}{\delta}{}^{(p)}_{1\,\text{caché}} \\ \overset{\text{caché}}{\delta}{}^{(p)}_{2\,\text{caché}} \end{pmatrix} \quad \text{avec:} \quad \overset{\text{caché}}{\delta_j^{(p)}} = \Big(\sum_{k} \overset{\text{sortie}}{\delta_k^{(p)}} z_{jk}\Big) \times \underbrace{U_j(1-U_j)}_{\text{derivee sigmoide}}$$

k $\to$ neurones couche sortie

4). calcul correction sur les poids w_{ij} (entre couche entree et couche cachee)

$$\Delta w_{ij}^{(p)} = -\alpha \; \overset{\text{caché}}{\delta_j^{(p)}} \times x_i^{(p)} \;/\, (\text{Nb echantillons})$$

rq: si l'on avait défini $\overset{\text{caché}}{\delta_j^{(p)}} = \dfrac{\partial C}{\partial U_j}$ on aurait obtenu : $\overset{\text{caché}}{\delta_j} = \sum_k \delta_k^{(p)} z_{jk}$

(car : $\dfrac{\partial C}{\partial \sigma_j} = \dfrac{\partial C}{\partial U_j} \times \dfrac{\partial U_j}{\partial \sigma_j} = \dfrac{\partial C}{\partial U_j} \times g'(\sigma_j) = \dfrac{\partial C}{\partial U_j} \times U_j(1-U_j)$)

mais le facteur $U_j(1-U_j)$ serait réapparu dans $\Delta w_{ij}^{(p)} = -\alpha \; \overset{\text{caché}}{\delta_j^{(p)}} \times U_j(1-U_j)\, x_i^{(p)}$

Les calculs se généralisent au cas où il y a plusieurs couches cachées

A) entre couche de sortie et dernière couche cachée :

a) • $\overset{\text{sortie}}{\delta_k^{(p)}} = \hat{y}_k^{(p)} - y_k^{(p)}$, $k = 0, \ldots$ tous les neurones de sortie

$\hat{y}_k^{(p)}$: prédiction ; $y_k^{(p)}$: vraie valeur

c) • $\Delta z_{jk} = \dfrac{-\alpha}{\text{Nb échantillon}} \sum_{(p)} \overset{\text{sortie}}{\delta_k^{(p)}} \times U_j^{(p)}$, j = tous les neurones de la dernière couche cachée, k = tous les neurones de la couche de sortie

$U_j^{(p)}$: production couche cachée

b) • $\overset{\text{cachée}}{\delta_j^{(p)}} = \sum_{\substack{k \\ \text{couche sortie}}} \overset{\text{sortie}}{\delta_k^{(p)}} z_{jk}$

z_{jk} : poids entre couche de sortie et dernière couche cachée (valeur à prendre <u>avant</u> correction)

B) entre deux couches cachées, si $\overset{\text{caché}}{\delta_j}$ est défini par : $\overset{\text{caché}}{\delta_j^{(p)}} = \dfrac{\partial C^{(p)}}{\partial U_j}$

U_j → <u>sortie de la couche</u> ~~précédente~~ <u>après application de sigmoïde</u>

a) • $\overset{\text{caché amont}}{\delta_j^{(p)}} = \sum_{\substack{k \\ \text{couche cachée aval}}} \overset{\text{caché aval}}{\delta_k^{(p)}} z_{jk}$

z_{jk} : poids entre les couches cachées amont et aval (valeur à prendre avant correction)

b) • $\Delta z_{jk} = \dfrac{-\alpha}{\text{Nb échantillon}} \sum_{(p)} \overset{\text{caché aval}}{\delta_k^{(p)}} \underbrace{U_k (1 - U_k)}_{\text{production neurones } \underline{\text{aval}}} x_j$

x_j : production neurones <u>amont</u>

* Si $\overset{\text{caché}}{\delta_j}$ est défini par : $\overset{\text{caché}}{\delta_j^{(p)}} = \dfrac{\partial C^{(p)}}{\partial \sigma_j} = \dfrac{\partial C^{(p)}}{\partial U_j} \times \dfrac{\partial U_j}{\partial \sigma_j} = U_j (1 - U_j) \times \dfrac{\partial C^{(p)}}{\partial U_j}$

σ_j → <u>sortie de la couche</u> ~~précédente aval~~ <u>avant application de la sigmoïde</u>

a) $\overset{\text{caché amont}}{\delta_j^{(p)}} = \sum_{\substack{k \\ \text{couche cachée aval}}} \overset{\text{caché aval}}{\delta_k^{(p)}} \times z_{jk} \times U_j (1 - U_j)$

U_j : production

b) $\Delta z_{jk} = \dfrac{-\alpha}{\text{Nb échantillon}} \sum_{(p)} \overset{\text{couche aval}}{\delta_k^{(p)}} \times x_j$

x_j : production neurones <u>amont</u>