La fonction Softmax

February 9, 2024

La fonction softmax, est une généralisation de la fonction logistique. Elle convertit un vecteur de K nombres réels en une distribution de probabilités sur K choix. Le cadre d'utilisation de softmax est celui du **problème de classification multi-classe**. Dans un tel problème, on demande en général que le modèle prédictif renvoie :

- Soit **la classe prédite** (identifiée par un numéro par exemple).
- Soit **la liste des probabilités d'appartenance à chaque classe**. Le modèle renvoie alors **un vecteur de probabilité**, c'est-à-dire un vecteur de nombres entre 0 et 1, dont la somme vaut 1.

Si le modèle est un réseau de neurones, alors on est obligé d'être dans le deuxième cas, car c'est un modèle qui doit être dérivable si l'on veut utiliser la méthode du gradient.

La fonction sigmoïde, qui est l'activation la plus standard, ne renvoie pas de vecteur de probabilités, car il n'est pas garanti que la somme des valeurs valent 1. On pourrait quand même utiliser cette activation, mais dans ce cas il y a les inconvénients suivants :

- L'utilisateur du réseau doit à la main normaliser les valeurs pour obtenir un vecteur de probabilité.
- Lors de l'apprentissage, le réseau doit lui-même apprendre que les valeurs cibles sont des vecteurs de probabilités.
- Certaines métriques nécessitent que la prédiction donnée par le réseau soit un vecteur de probabilité, comme par exemple l'entropie croisée (ce n'est pas le cas pour la métrique MeanSquareError).

Ainsi, la fonction d'activation softmax sert à **forcer le réseau de neurones à renvoyer des vecteurs de probabilités**.

Définition de la fonction softmax :

Le principe de la fonction softmax est simple : on part d'un vecteur $(x_0, x_1, ...)$ avec les coefficients quelconques, et on veut obtenir un vecteur avec des coefficients positifs de somme 1.

Il suffit alors :

- de remplacer chaque x_i par $\exp(x_i)$ pour obtenir un nombre positif.
- puis de diviser par la somme des coefficients pour rendre une probabilité.

Ce qui donne comme formule $\text{softmax}(x_0, x_1, ...) = (y_0, y_1, ...)$ où les y_i valent :

$$y_i = \frac{\exp(x_i)}{\sum_{p \geq 0} \exp(x_p)}$$

Exemple :
$x = (x_1, x_2, x_3) = (1, 4, -2)$
Alors pour calculer $\text{softmax}(x)$ il faut calculer :

$$\exp(x_1) = \exp(1) = 2.71...$$

$$\exp(x_2) = \exp(2) = 54.59...$$

$$\exp(x_3) = \exp(3) = 0.13...$$

Ensuite on divise chacun de ces nombres par la somme $s = \exp(x_1) + \exp(x_2) + \exp(x_3) = 74.81...$ ce qui donne :

$$\text{softmax}(x)_1 = \frac{\exp(x_1)}{s}$$

$$\text{softmax}(x_2)_2 = \frac{\exp(x_2)}{s}$$

$$\text{softmax}(x_3)_3 = \frac{\exp(x_3)}{s}$$

Ainsi $\text{softmax}(x) = (0.047..., 0.95, 0.002...)$ et on peut vérifier que la somme des coefficients fait bien 1.

Lien avec la fonction argmax :

Mais alors d'où vient le nom softmax ? Il vient du fait que c'est une approximation de la fonction $argmax$ en une fonction lisse C^∞.

La version encodée One hot de $argmax$ renvoie un vecteur de probabilité de type Dirac, c'est-à-dire une valeur 1 et que des 0 ailleurs. Par exemple sur le vecteur $x = (1, 4, -2)$, la version encodée One Hot de son argmax est le vecteur (0,1,0) où la position du 1 est la position du plus grand coefficient de x. L'inconvénient de cette fonction n'est pas dérivable... On introduit alors cette nouvelle fonction pour pallier à ça.

En effet, on peut voir que $\text{softmax}(x) = (0.047..., 0.95, 0.002...)$ est une assez bonne approximation de (0,1,0), ainsi softmax est une approximation dérivable de la version encodée One hot de $argmax$, d'où le nom "soft".

Proposition :

La fonction softmax S est différentiable et $\frac{\partial S_i}{\partial x_j} = S_i(\delta_{i,j} - S_j)$ pour tout $i, j \in [\![1; n]\!]$, où $\delta_{i,j}$ est la fonction delta de Kronecker définie par $\delta_{i,j} = 1$ si $i = j$ et 0 sinon.

Preuve :

On considère la fonction softmax définie par :

$$S : R^n \to [0,1]^n$$

$$(x_1, x_2, ..., x_n) \to (S_1(x_1, ..., x_n), ..., S_n(x_1, ..., x_n))$$

Ce que nous cherchons ce sont les dérivées de la i-ème sortie par rapport à la j-ème entrée, c'est-à-dire $\frac{\partial S_i}{\partial x_j}$.

Et on a :

$$\frac{\partial S_i}{\partial x_j}(x_1, \ldots, x_n) = \frac{\partial}{\partial x_j}\left(\frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}}\right)$$

Si $i = j$:

$$\begin{aligned}\frac{\partial}{\partial x_j}\left(\frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}}\right) &= \frac{e^{x_i}(\sum_{k=1}^n e^{x_k}) - e^{x_i}e^{x_j}}{(\sum_{k=1}^n e^{x_k})^2} \\ &= e^{x_i}\frac{\sum_{k=1}^n e^{x_k}}{(\sum_{k=1}^n e^{x_k})^2} - \left(\frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}}\right)^2 \\ &= S_i(1 - S_i).\end{aligned}$$

Si $i \neq j$:

$$\begin{aligned}\frac{\partial}{\partial x_j}\left(\frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}}\right) &= \frac{0 - e^{x_j}e^{x_i}}{(\sum_{k=1}^n e^{x_k})^2} \\ &= -S_i S_j.\end{aligned}$$

Finalement, on a :

$$\frac{\partial S_i}{\partial x_j}(x_1, \ldots, x_n) = \begin{cases} S_i(1 - S_j) & \text{si } i = j \\ -S_i S_j & \text{si } i \neq j \end{cases} = S_i(\delta_{i,j} - S_j).$$

Interprétation probabiliste :

Les propriétés de softmax (toutes les valeurs de sorties sont dans $[0, 1]$ et s'additionnent à 1) rendent la fonction adaptée à une interprétation probabiliste très utile dans l'apprentissage automatique, en particulier dans les tâches de classification multiclasse.

On peut interpréter softmax comme la probabilité prédite pour la jème classe étant donné un vecteur d'échantillon x et un vecteur de pondération w d'où

$$P(y = j|\mathbf{x}) = \frac{e^{\mathbf{x}^\intercal \mathbf{w}_j}}{\sum_{k=1}^K e^{\mathbf{x}^\intercal \mathbf{w}_k}}$$