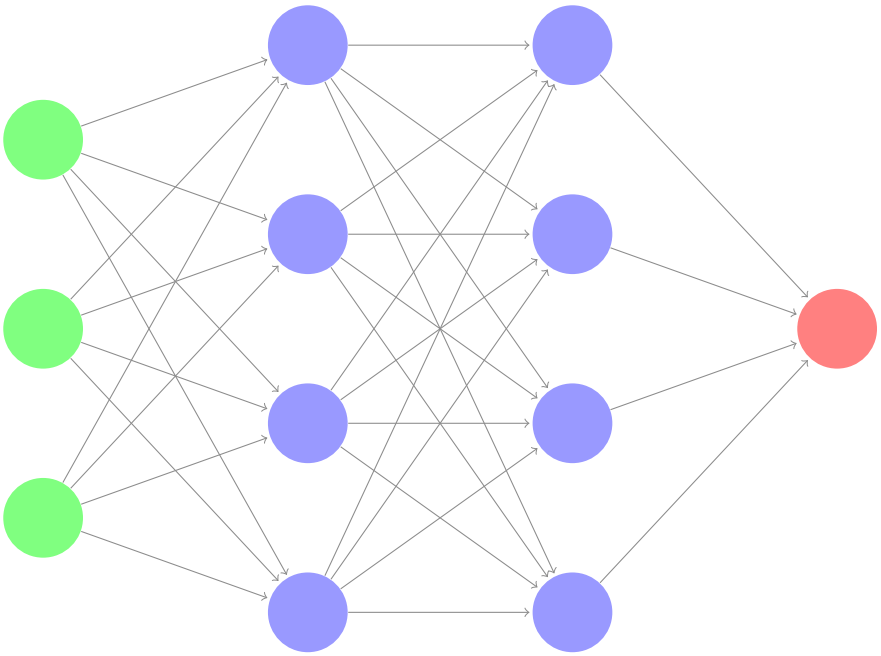
les maths derrière un réseau neuronal



olivia

hugo lageneste
hugo@olivia-ai.org

Contents

Les maths derrière un réseau de neurones artificiels

Introduction

1 Exemple d'application

Prenons un exemple pour voir comment un ANN[1] fonctionne.

Obésité	Exercice	Fume	Diabétique
1	0	0	1
0	1	0	0
0	0	1	0
1	1	0	1

Figure 1: Exemple d'un ensemble de personnes avec de l'obésité, qui font de l'exercice, fument and sont diabétiques

Dans la figure précédente, les 1 représentent vrai et les 0 faux. Nous pouvons observer que dans cet exemple, une personne diabétique est inévitablement obèse. Et si nous voulons un programme qui prends seulement en paramètre ces 4 exemples et peut prédire avec différents exemples si une personne est diabétique ou non?

Nous pouvons modéliser ce programme comme un ANN avec 3 neurones d'entrée qui représentent les colonnes Obésité, Exercice et Tabagisme et un seul neurone de sortie qui représente la colonne des diabétiques.

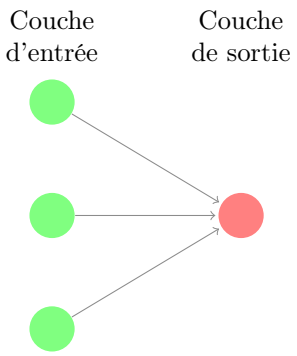


Figure 2: Diagramme de l'ANN pour modéliser l'exemple

2 Fonctionnement du réseau neuronal

Les valeurs d'entrée seront insérées dans les neurones d'entrée et le réseau fonctionnera comme une "fonction" pour dire si la personne est diabétique ou non. Chaque liaison entre chaque neurone est une valeur appelée poids et est unique. Et chaque neurone possède une valeur spécifique, calculée avec les valeurs des neurones précédents et le poids. L'objectif du réseau de neurones est de trouver les poids corrects pour faire les prédictions finales droite.

[1] Réseau de neurones artificiels, Artificial Neural Network en anglais.

Pour que les poids soient corrects, nous procéderons en plusieurs étapes.Tout d'abord, le réseau de neurones calculera la valeur des neurones de la couche de sortie appelée la prédiction. Ensuite, le réseau fera la différence entre la prédiction et la sortie réelle et ajustera les poids pour rendre la prédiction plus précise. Ce processus sera répété jusqu'à ce que le taux de réussite soit satisfaisant.

Après la construction du réseau de neurones, nous pourrons envoyer des valeurs comme 1, 1 et 1 (cela signifie donc que la personne est obèse, fait de l'exercice et fume) et obtenir une réponse du réseau de neurones qui nous indique que la personne est diabétique.

A Propagation en avant

Comme nous l'avons vu dans l'introduction, chaque neurone possède une valeur comprise entre 0 et 1, calculée avec les valeurs des neurones précédents, les poids et un biais. Nous allons voir comment ces valeurs sont calculées.

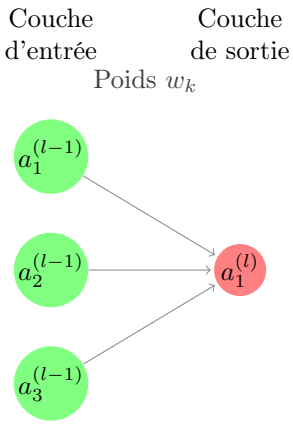


Figure 3: Schéma d'un réseau de neurones, l'exposant représente l'index de la couche et l'indice l'index du neurone puis a est la valeur détenue dans le neurone.

Créons une notation, $z_1^{(l)}$ qui est le biais ajouté au produit ponctuel des poids w_k et des valeurs des neurones précédents $a_k^{(l-1)}$.

$$z_1^{(l)} = b_1^{(l)} + \sum_{k=1}^{n_{(l-1)}} a_k^{(l-1)} w_k$$

Afin de maintenir les valeurs dans les neurones entre 0 et 1, nous allons utiliser la fonction sigmoïde qui est définie sur $]0, 1[$, noté :

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

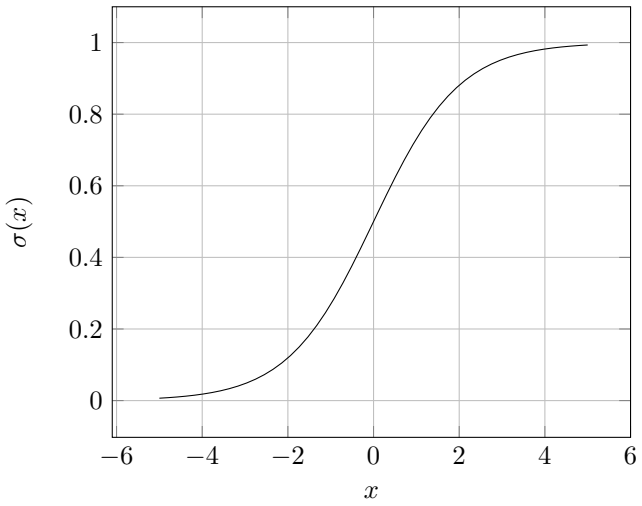


Figure 4: Représentation graphique de la fonction sigmoïde

Thus,

$$a_1^{(l)} = \sigma\left(z_1^{(l)}\right)$$

$$a_1^{(l)} \in]0, 1[$$

Ce calcul peut aussi être visualisé avec des matrices

$$a_1^{(l)} = \sigma\left(\begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix} \begin{bmatrix} a_1^{(l-1)} \\ a_2^{(l-1)} \\ a_3^{(l-1)} \end{bmatrix} + \begin{bmatrix} b_1^{(l)} \end{bmatrix}\right)$$

B Propagation en arrière

1 Fonction de coût

Prenons un exemple simple de réseau neural à une couche d'entrée et une couche de sortie

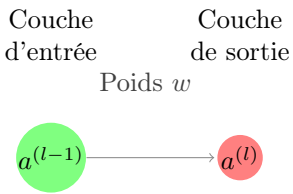


Figure 5: Diagramme de l'ANN pour modéliser l'exemple de la fonction de coût

Une fois de plus,

$$z^{(l)} = b + a^{(l-1)}w$$

$$a^{(l)} = \sigma\left(z^{(l)}\right)$$

L'objectif est maintenant d'ajuster les poids pour rendre la prévision plus précise. Introduisons la fonction de coût qui calcule la différence carrée entre la prédiction et la production réelle y.

$$C_1\left(a^{(l)}, y\right) = \left(a^{(l)} - y\right)^2$$

Plus la fonction de coût est petite, plus les prévisions sont précises. Mathématiquement, l'objectif est de minimiser la fonction de coût.

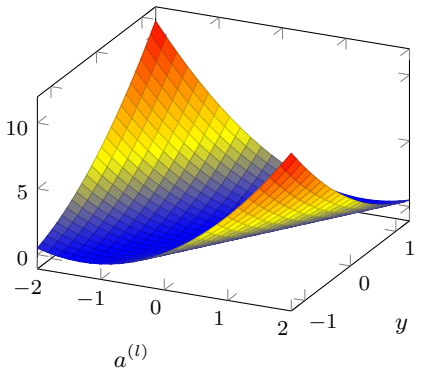


Figure 6: Représentation graphique de la fonction de coût du réseau de neurones, figure 5

2 Descente en pente

Il nous faudra maintenant comprendre à quel point la fonction de coût est sensible aux petites variations de w_k car souvenez-vous de 2, le but est d'ajuster les poids. Ainsi, nous déterminerons la dérivée partielle de C par rapport à w en utilisant la règle de la chaîne.

$$\frac{\partial C_1}{\partial w} = \frac{\partial C_1}{\partial a^{(l)}} \frac{\partial a^{(l)}}{\partial z^{(l)}} \frac{\partial z^{(l)}}{\partial w}$$

En effet,

$$\frac{\partial C_1}{\partial a^{(l)}} = 2\left(a^{(l)} - y\right)$$

$$\frac{\partial a^{(l)}}{\partial z^{(l)}} = \frac{\partial \sigma\left(z^{(l)}\right)}{\partial z^{(l)}} = \sigma\prime\left(z^{(l)}\right)$$

$$\frac{\partial z^{(l)}}{\partial w} = \frac{\partial\left(b + a^{(l-1)}w\right)}{\partial w} = a^{(l-1)}$$

Tous ensemble, cela nous donne

$$\frac{\partial C_1}{\partial w} = 2\left(a^{(l)} - y\right)\sigma\prime\left(z^{(l)}\right)a^{(l-1)}$$

Nous utiliserons cette formule pour calculer les ajustements à apporter aux poids plusieurs fois jusqu'à ce que les prédictions soient exactes.

$$w = w + \alpha\frac{\partial C_1}{\partial w}$$

C Un exemple complexe

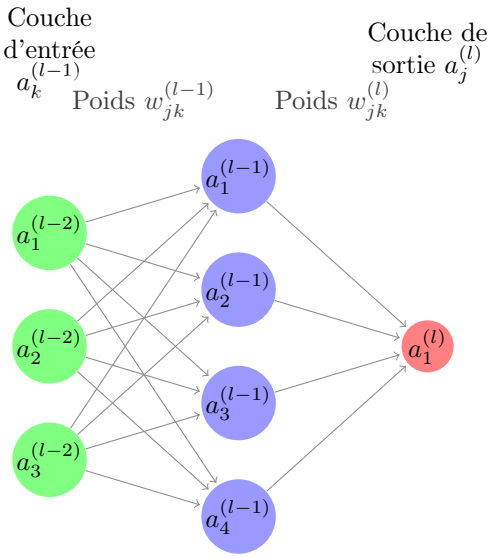


Figure 7: Diagramme d'un ANN plus complexe

Nous allons modéliser ce problème à l'aide de matrices.

1 Propagation en avant

Tout d'abord, nous allons créer une matrice pour chaque couche de poids

$$w^{(l-2)} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} \\ w_{21} & w_{22} & w_{23} & w_{24} \\ w_{31} & w_{32} & w_{33} & w_{34} \end{bmatrix}$$

$$w^{(l-1)} = \begin{bmatrix} w_{11} \\ w_{21} \\ w_{31} \\ w_{41} \end{bmatrix}$$

Une couche est représentée par une matrice t par $k^{(l)}$, où t est le nombre d'exemples de formation et k le nombre de nœuds dans une couche

$$a^{(l-2)} = \begin{bmatrix} a_1^{(l-2)} & a_2^{(l-2)} & a_3^{(l-2)} \end{bmatrix}$$

Ensuite, pour calculer les valeurs de la couche suivante, nous devons exprimer z pour calculer a.

$$z^{(l)} = a^{(l-1)} \cdot w^{(l)} + b^{(l-1)}$$

$$a = \sigma(z)$$

2 Propagation en arrière

Comme nous avons maintenant une couche cachée, nous devrons exprimer la dérivée partielle de C par rapport à $w^{(l)}$ couché en termes de l.
Les ajustements prendront la forme suivante

$$w^{(l)} = w^{(l)} + \alpha \frac{\partial C}{\partial w^{(l)}}$$

Ici, α représente le taux d'apprentissage.
Nous devons donc calculer cettte dérivée $\frac{\partial C}{\partial w^{(l)}}$. Utilisation de la règle de la chaîne,

$$\frac{\partial C}{\partial w^{(l)}} = \frac{\partial C}{\partial z^{(l)}} \frac{\partial z^{(l)}}{\partial w^{(l)}}$$

Tout d'abord, calculons $\frac{\partial z^{(l)}}{\partial w^{(l)}}$

$$\frac{\partial z^{(l)}}{\partial w^{(l)}} = \frac{\partial}{\partial w^{(l)}} \left(a^{(l-1)} w^{(l)} + b^{(l-1)} \right) = a^{(l-1)}$$

Ensuite, nous calculerons $\frac{C}{z^{(l)}}$ couché en termes de $z^{(l+1)}$ afin de modéliser la rétropropagation en programmation.

$$\frac{\partial C}{\partial z^{(l)}} = \frac{\partial C}{\partial z^{(l+1)}} \frac{\partial z^{(l+1)}}{\partial a^{(l)}} \frac{\partial a^{(l)}}{\partial z^{(l)}}$$

$$\frac{\partial z^{(l+1)}}{\partial a^{(l)}} = \frac{\partial}{\partial a^{(l)}} \left(w^{(l+1)} a^{(l)} + b^{(l)} \right) = w^{(l+1)}$$

$$\frac{\partial a^{(l)}}{\partial z^{(l)}} = \frac{\partial}{\partial z^{(l)}} \sigma \left(z^{(l)} \right) = \sigma\prime \left(z^{(l)} \right)$$

Ainsi, après avoir ajusté les termes pour faire fonctionner les produits à points, nous avons

$$\frac{\partial C}{\partial z^{(l)}} = \left(w^{(l+1)^T} \cdot \frac{\partial C}{\partial z^{(l+1)}} \right) \times \sigma\prime \left(z^{(l)} \right)$$

Et,

$$\frac{\partial C}{\partial w^{(l)}} = \left(w^{(l+1)^T} \cdot \frac{\partial C}{\partial z^{(l+1)}} \right) \times \sigma\prime \left(z^{(l)} \right) \times a^{(l-1)}$$

Comme vous pouvez le voir ici, pour calculer les ajustements des poids, vous avez besoin de la dérivée de C par rapport aux z suivants. Nous devrons donc calculer la dérivée de C par rapport à z pour la dernière couche afin de ne jamais sortir de l'intervalle. Ici, L est la dernière couche

$$\frac{\partial C}{\partial z^{(L)}} = 2(a^{(L)} - y)\sigma\prime \left(z^{(L)} \right)$$

$$\frac{\partial C}{\partial z^{(L)}} = 2(a^{(L)} - y)\sigma \left(z^{(L)} \right) \left(1 - \sigma \left(z^{(L)} \right) \right)$$

$$\frac{\partial C}{\partial z^{(L)}} = 2(a^{(L)} - y)a^{(L)} \left(1 - a^{(L)} \right)$$

Nous disposons ici de toutes les ressources nécessaires pour construire un réseau neuronal artificiel.