Enseigner l'Intelligence Artificielle

Les réseaux de neurones

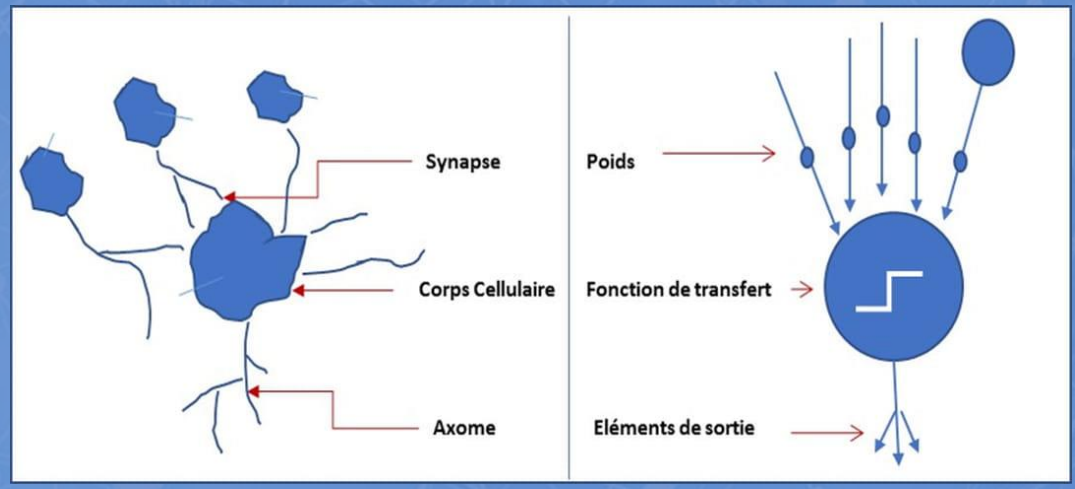
Laurent Gimazane

Le réseau de neurones artificiels

Les réseaux neuronaux artificiels sont une modélisation au plus près du réel fonctionnement des réseaux de neurones de notre cerveau. Le plus célèbre d'entre eux est le perceptron multicouche (Le perceptron est un algorithme qui reçoit des données avec un certain poids, qui les calcule et qui produit un résultat transmis à d'autres perceptrons par des liens interconnectés), un système artificiel capable d'apprendre par l'expérience et qui a été modélisé par Franck Rosenblatt en 1957. Neurones biologiques d'où émergent les associations implicites du cerveau.

Le neurone artificiel est donc appelé aussi neurone formel et qui reprend le fonctionnement du neurone biologique de manière simplifiée. En somme, c'est un dispositif à plusieurs entrées et une sortie qui modélise certaines propriétés du neurone biologique, mais ce n'est qu'avec l'augmentation de la puissance de calcul courant des années 2000 que le perceptron a pu être utilisé et a été démocratisé.

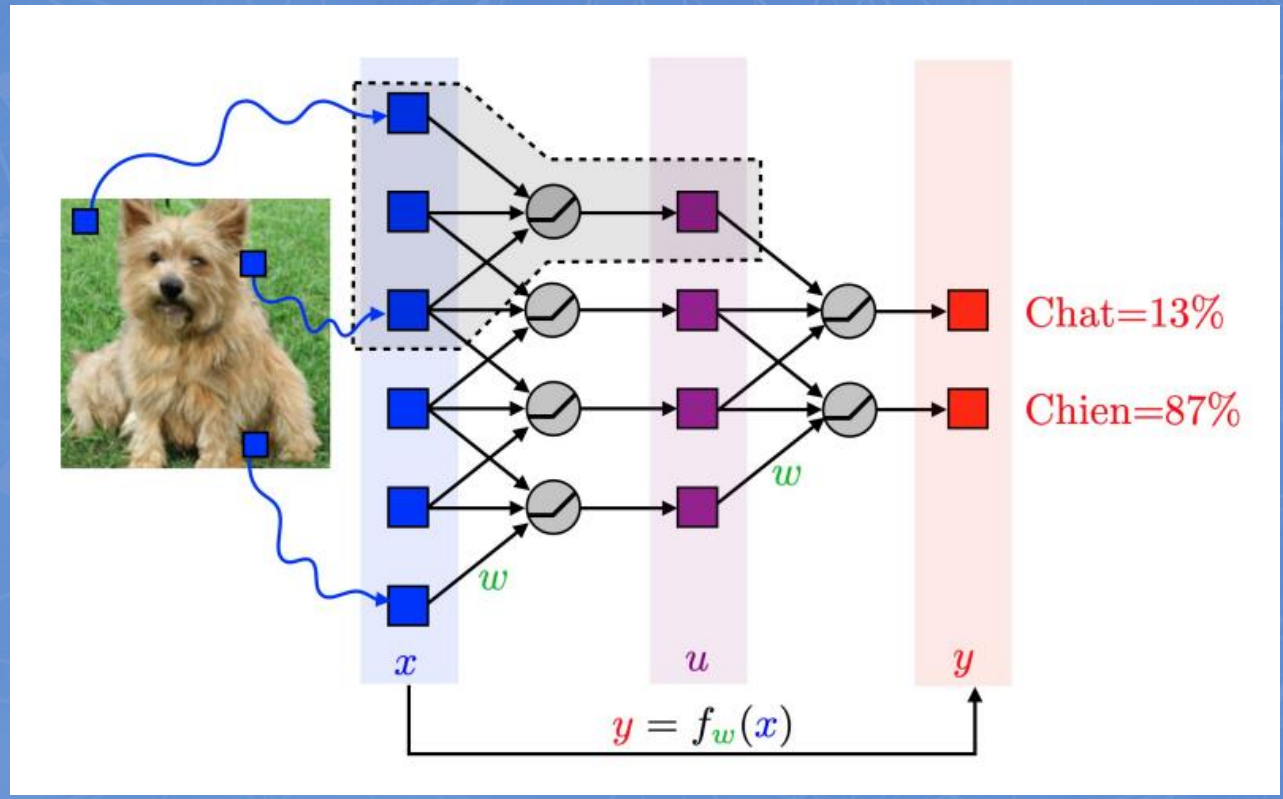
On organise les réseaux neuronaux avec une couche de neurones d'entrées et de sorties bien définie. On définit aussi des liens directs entre les nœuds pour savoir où se dirige l'information c'est-à-dire comment elle se propage.



Similitude structurelle entre un réseau de neurones artificiels et biologiques

Le réseau de neurones artificiels

Alors que certains problèmes peuvent être décrits comme ayant des ensembles discrets de règles pour les caractéristiques de données (c'est-à-dire n'ayant pas de points de données ou de règles en commun avec des ensembles voisins), la définition de règles pour la reconnaissance d'images est assez différente. Par exemple, définir ou créer des règles pour distinguer un chat d'un chien dans une image est extrêmement compliqué, car ils partagent de nombreuses caractéristiques communes.

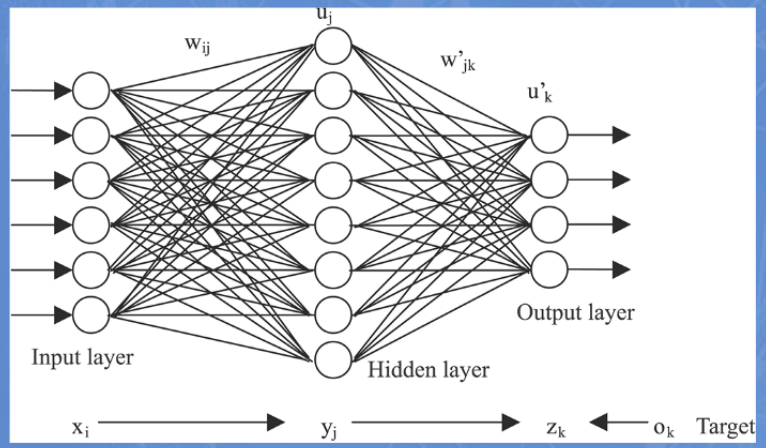


Les mathématiques des réseaux de neurones – Lien page

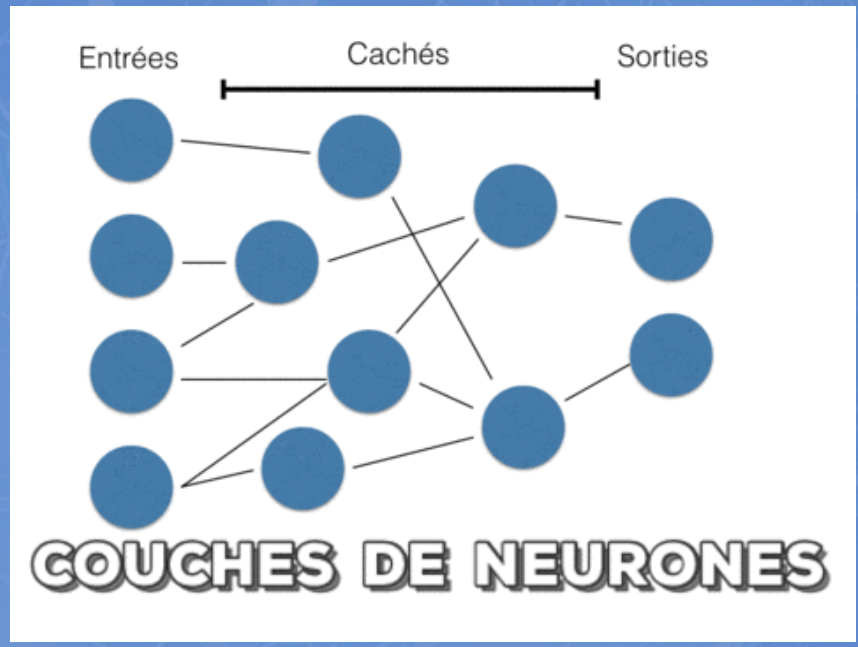
Le réseau de neurones artificiels

C'est pourquoi nous utilisons d'énormes quantités d'informations pour générer ces règles ; c'est grâce à cette formation approfondie et à l'énorme quantité de données que l'ANN peut devenir un outil viable pour fournir une solution possible. Ces réseaux sont constitués de nœuds : des neurones artificiels, organisés en couches, avec des connexions pondérées entre les neurones des couches qui les précèdent et les suivent immédiatement. Les neurones des couches d'entrée reçoivent les données brutes et les couches de sortie produisent le résultat final. Entre les deux, il y a généralement une série de couches cachées qui traitent les données. Ils sont activés à différents degrés en fonction des caractéristiques de données que la couche précédente a observées et propagées. Par conséquent, la couche de sortie peut fournir une sortie estimée (avec des degrés de confiance variables) indiquant que les données d'entrée, basées sur leurs caractéristiques observées, ont une qualité spécifique. Par exemple, il peut identifier un caractère spécifique, une image comme contenant un objet spécifique ou un e-mail dont le texte est évalué comme spam.

Autre précision et différence notable, la notion de temps, importante en biologie, n'est pas prise en compte pour la majorité des neurones formels.



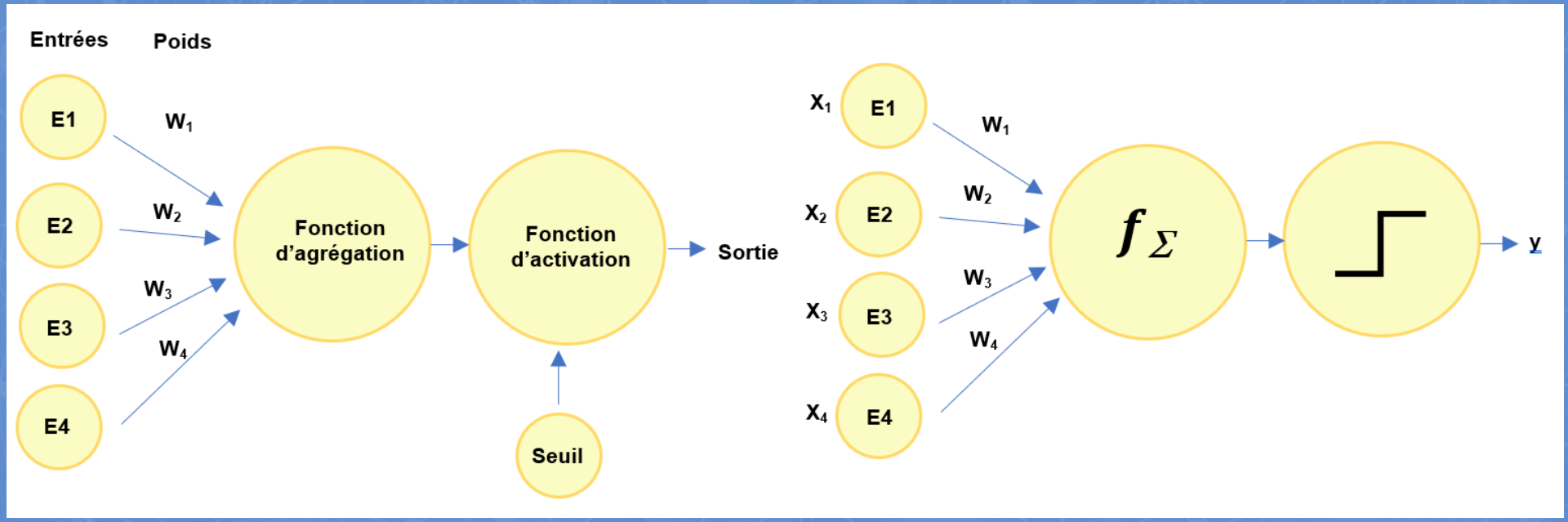
Représentation et principe d'un réseau de neurones artificiels



Principe du réseau de neurones artificiels

Un neurone est une unité de calcul. Elle peut comporter 1 ou plusieurs entrées et 1 sortie calculée grâce à différentes caractéristiques :

- Des entrées (X ou E) et une sortie qui peut varier de 0 à 1, (ou -1 à 1 avec la fonction sigmoïde), la sortie applique le même fonctionnement.
- Chacune des entrées à un poids (W – Weight en anglais) accordé à chacune des entrées et permettant de modifier l'importance de certaines par rapport aux autres.
- Une fonction d'agrégation, qui permet de calculer une unique valeur à partir des entrées et des poids correspondants (une somme par exemple).
- Un seuil (ou biais), permettant d'indiquer quand le neurone doit agir (dans le cas d'une sortie de 0 à 1, le seuil peut être fixé à 0,5).
- Une fonction d'activation, qui associe à chaque valeur agrégée une unique valeur de sortie dépendant du seuil.
- Le résultat du calcul du neurone n'est autre que la somme des produits de toutes les entrées et des poids, le tout passé par un "filtre" que l'on appelle fonction d'activation (sortie).



Principe du réseau de neurones artificiels

Illustration :

Votre cerveau peut traiter la douce odeur de pain qui s'échappe d'une boulangerie en plusieurs étapes : « Je sens une odeur de pain chaud » (c'est l'entrée de données)… « J'adore le pain chaud ! » (pensée) … 'Je vais m'acheter une baguette' (prise de décision) … 'Je dois faire attention à mon alimentation et limiter les féculents'(souvenir) … 'En ne mangeant qu'un seul morceau je limite les risques tout en me satisfaisant' (raisonnement) "Je le fais !" (action).

C'est donc une méthode de Machine Learning que vous pouvez utiliser pour des problématiques de prédiction et de classement en particulier pour des phénomènes complexes à modéliser et/ou non linéaires.

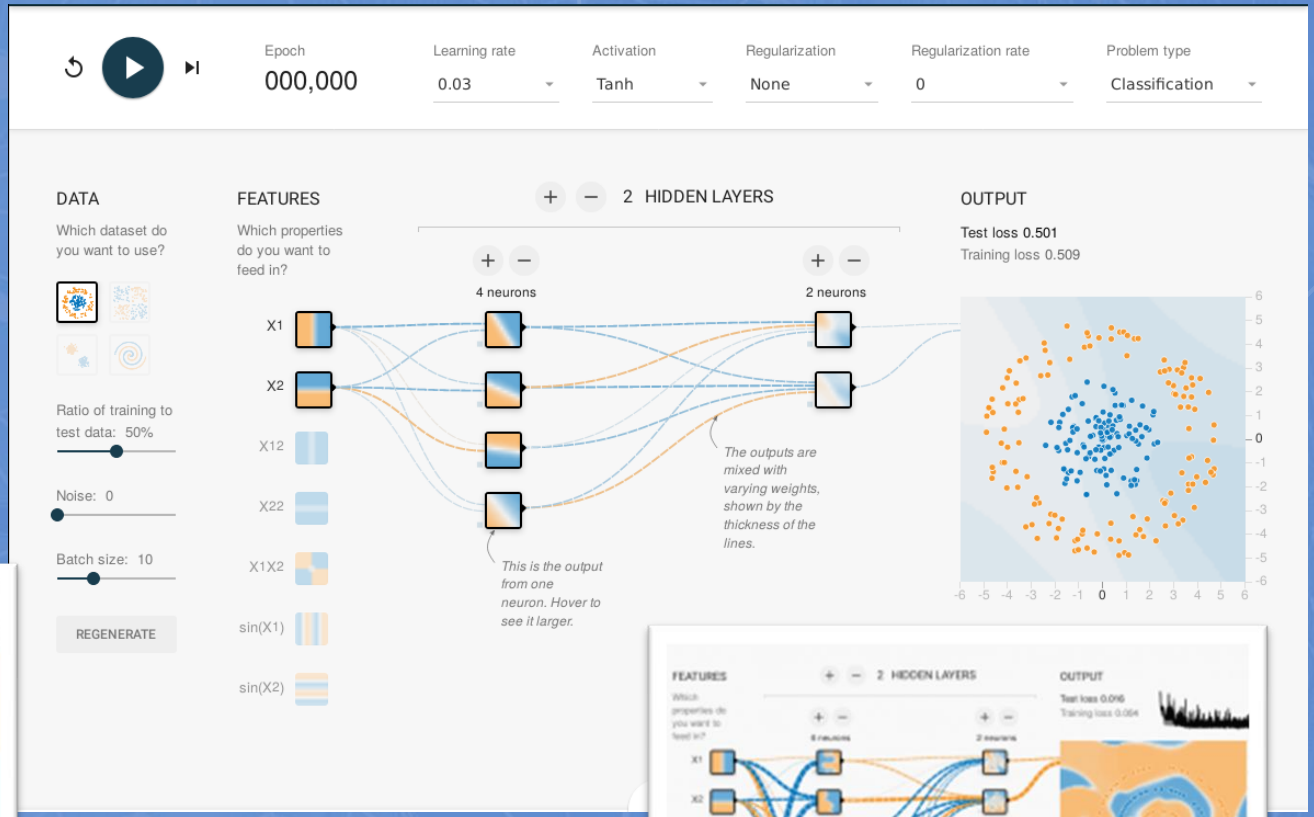
A noter qu'il existe différents types de réseaux de neurones :

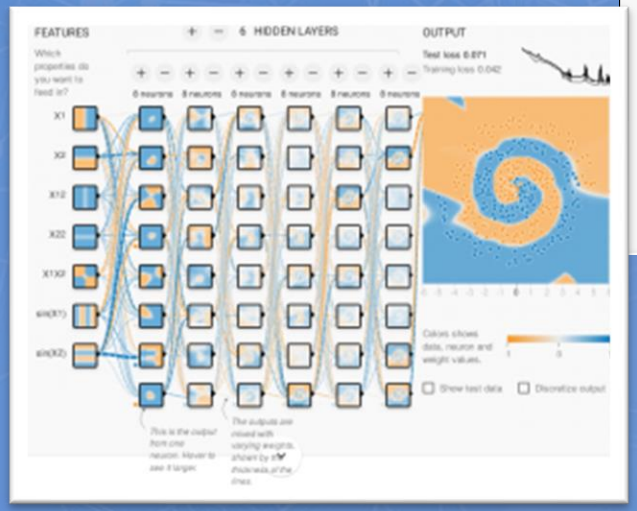
- réseaux de neurones convolutifs,
- réseaux de neurones récurrents,
- réseaux de neurones artificiels,
- réseaux de neurones discriminatifs...

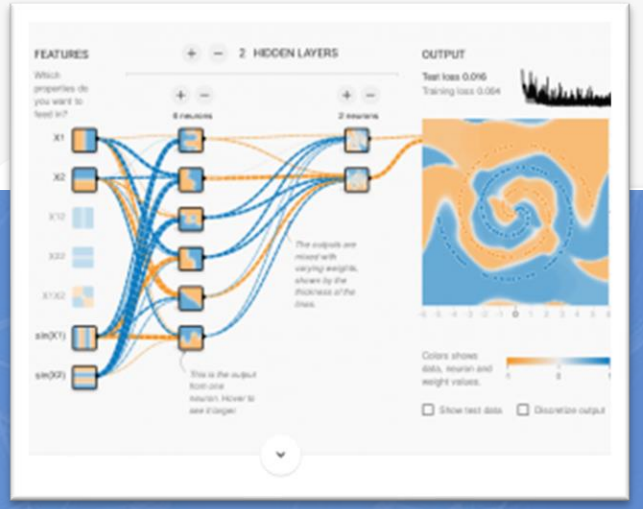
Illustration d’un réseau de neurones artificiels

Une illustration du réseau de neurones avec le site **TensorFlow**

[Lien](#) vers le site







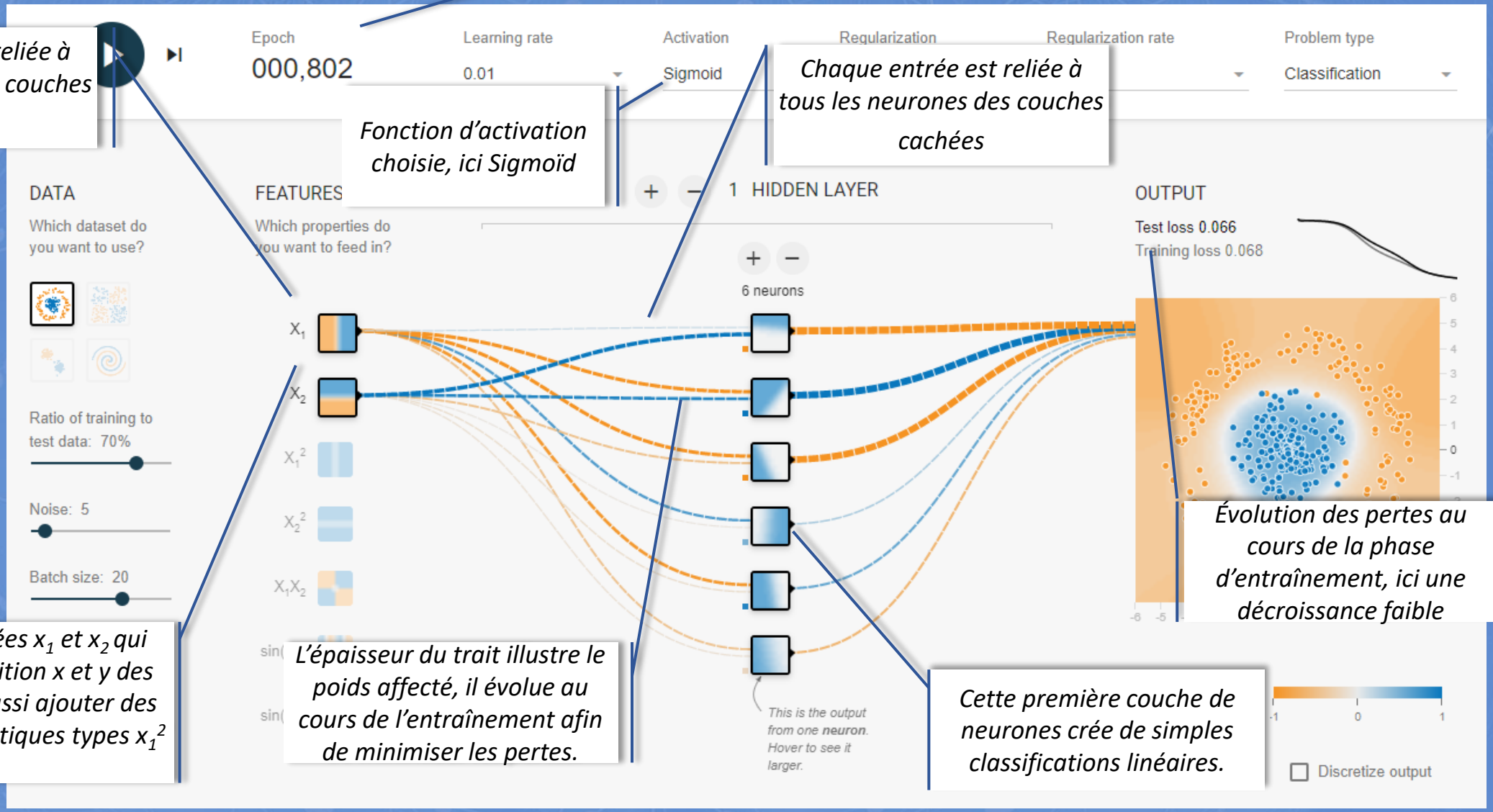
Quelques explications

Une Epoch est un apprentissage sur le jeu de données complet. Il en faut plusieurs (ici 802) pour arriver à apprendre correctement, Une itération est le passage sur une donnée, il y a donc plusieurs itérations par Epoch.

Chaque entrée est reliée à tous les neurones des couches cachées

Fonction d'activation choisie, ici Sigmoïd

Chaque entrée est reliée à tous les neurones des couches cachées



Évolution des pertes au cours de la phase d'entraînement, ici une décroissance faible

Ici on utilise les 2 entrées x_1 et x_2 qui correspondent à la position x et y des points, mais on peut aussi ajouter des combinaisons mathématiques types x_1^2 ou $x_1.x_2$

L'épaisseur du trait illustre le poids affecté, il évolue au cours de l'entraînement afin de minimiser les pertes.

Cette première couche de neurones crée de simples classifications linéaires.

Pour aller plus loin - Présentation

Avant toutes choses, précisons que si un problème est statistique, alors un réseau neuronal pourra le résoudre. Pour des problèmes plus complexes, on combinera cette technique avec d'autres algorithmes (algorithmes génétiques par exemple).

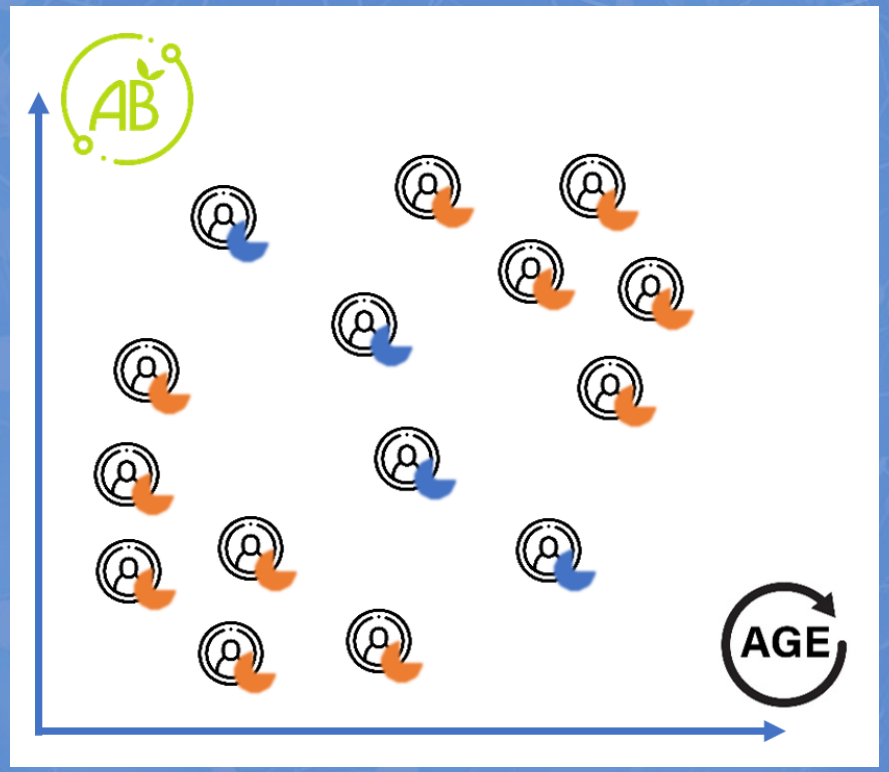
Pour utiliser un réseau neuronal, on doit commencer par une phase d'apprentissage. Prenons un exemple où toutes les données ne reflètent pas la réalité, mais admettons qu'on ait fait un sondage et qu'on souhaite un modèle de réseau de neurones pour prédire un résultat.

Pour qu'un réseau soit efficace, il faut l'entraîner et cela, avec un ensemble de données classifiées et normalisées.

Imaginons qu'on veuille déduire la préférence politique d'un futur électeur ! On aurait déjà plein d'informations, mais pour celles dont il manque un paramètre, que fait-on ? On utilise un réseau de neurones !

Prenons un exemple, les habitudes de consommation des acheteurs de produits bio selon la classe d'âge, on cherche à prédire si un client appartient au groupe bleu (bleu=1, orange=0) en fonction de son ancienneté et du pourcentage de produits bio achetés.

On observe qu'il s'agit d'un problème non linéairement séparable (on ne peut pas séparer les 2 groupes avec une droite). C'est là que rentre en jeu les algorithmes de réseaux de neurones, ils vont transformer les données en entrée afin que le problème devienne linéairement séparable. Et ce nouveau problème pourra alors facilement être résolu.



Pour aller plus loin – Phase de définition des données

On commence par une couche d'entrée dans laquelle chaque neurone correspond à une variable explicative (quantitative ou qualitative). On affecte des "poids" pour que certaines connexions soient plus forte que d'autres puis une ou plusieurs couches de neurones cachés et enfin une couche de neurone de sortie qui correspond à la variable prédite. Le nombre de neurones sur une couche peut varier, c'est le Data Scientist qui détermine la structure du réseau.

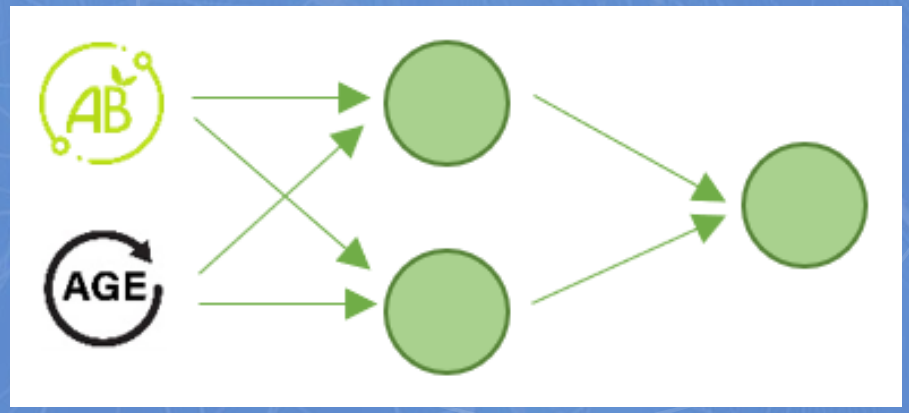
A noter que plus on ajoute de couches, plus les calculs sont longs, plus la prédiction lors de la phase d'apprentissage sera performant et plus le risque de sur-apprentissage (un algorithme qui se généralise mal à de nouvelles données) est élevé.

Pour utiliser un réseau neuronal, on doit commencer par un phase d'apprentissage car parfois toutes les données ne reflètent pas réalité.

Il existe 3 types d'apprentissages :

- L'apprentissage supervisé
- L'apprentissage non supervisé
- L'apprentissage par renforcement

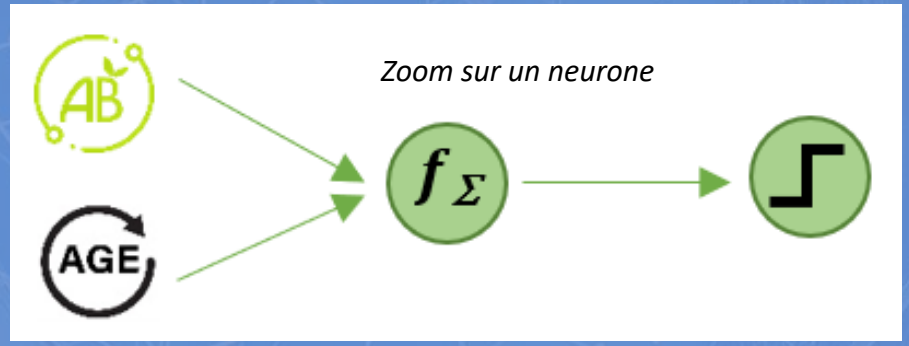
Ici, nous étudierons seulement l'apprentissage supervisé.



Couche d'entrée
Variables du modèle

Couche cachée

Couche de sortie



Fonction d'agrégation

Fonction d'activation

Pour faire simple, si on peut réduire notre problème à un tableau Excel alors celui-ci peut être résolu par un ANN.

Etape1 – Normalisation des données

La normalisation est une opération du pré-traitement de données qui a pour but de mettre sur une même échelle toutes les variables quantitatives. Supposons qu'on ait à comparer les variables taille et âge. Ces deux variables n'étant pas sur une même échelle de grandeur, la comparaison n'aura pas véritablement de sens. Une mise à échelle est nécessaire pour mettre nos variables sur la même échelle, c'est ce qui facilitera la comparaison et lui donnera du sens.

Il existe plusieurs types de normalisation (min-max ; standardisation; robuste…)

$$x_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \in [0\ ,\ 1]$$

Formule de normalisation min-max

Avant de commencer, il y a un élément à considérer : il a plusieurs valeurs différentes pour le pourcentage de produits bio acheté et pour l'âge. Si on avait un critère pour le sexe des consommateurs ce serait 2 niveaux avec des valeurs non quantitatives. Parce qu'un réseau neuronal ne comprend que les chiffres, nous devrions traduire des niveaux de taille tels "Petit/Moyen/Grand" en deux chiffres qui représenteraient ces 3 états :

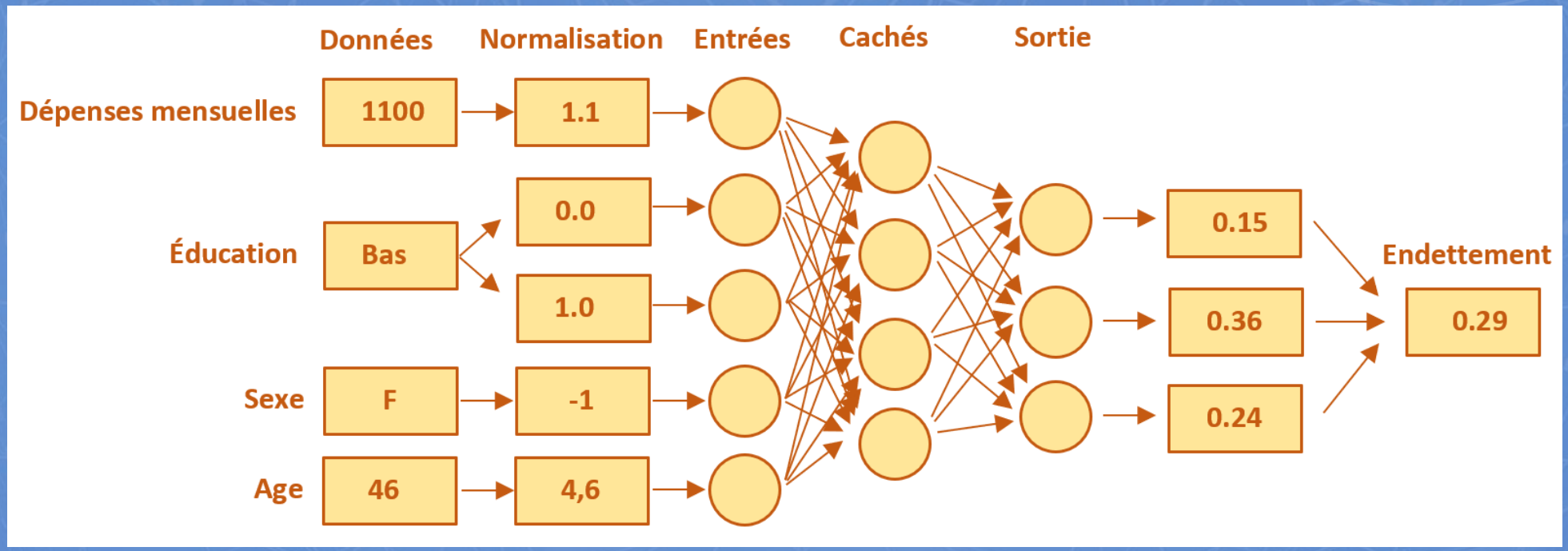
Petit : 0.0 | 1.0
Moyen : 1.0 | 0.0
Grand : 0.0 | 0.0

Etape1 – Normalisation des données - Exemple

Voici un exemple de données à classer normalisées pour prédire le taux d'endettement d'un foyer selon 4 critères d'entrée.
On peut remarquer qu'en "Sortie" : il y a trois valeurs. Il s'agit là de probabilité de probabilités !

Techniquement, ces chiffres ne sont pas réellement des probabilités, mais on peut les interpréter comme tel. On peut donc lire :

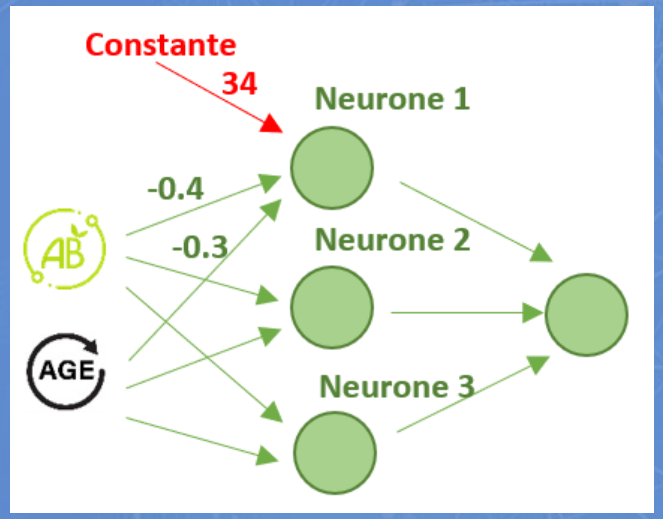
- La probabilité d'être fortement endetté (proche de 1)
- La probabilité d'être faiblement endetté (proche de 0)
- La probabilité d'avoir une situation incertaine (proche de 0,5)



Etape2 – Construction de la première couche cachée

A l'initialisation de l'ANN, l'ensemble de nos neurones ont une valeur aléatoire. Pour notre exemple, nous allons tester une seule couche cachée avec 3 neurones.

Pour chaque neurone de la couche cachée, l'algorithme va calculer une combinaison linéaire des neurones précédents et va appliquer une fonction sigmoïde au résultat (1/(1+exp(-x))).



Comment se calcule un neurone ?

Il existe deux types de neurones, concentrons-nous sur un seul type : le neurone **produit scalaire**. L'autre type est le neurone **de distance**.
Un neurone possède une valeur et les connexions entre chaque neurone ont une valeur nommée "poids".
Chaque Perceptron a un "faux neurone", c'est-à-dire une constante qui est importante pour l'apprentissage, ici 40.
Cela donne : 35 -(0.4 x 40) + (-0.3 x 62) = 0,4 pour le neurone caché1.

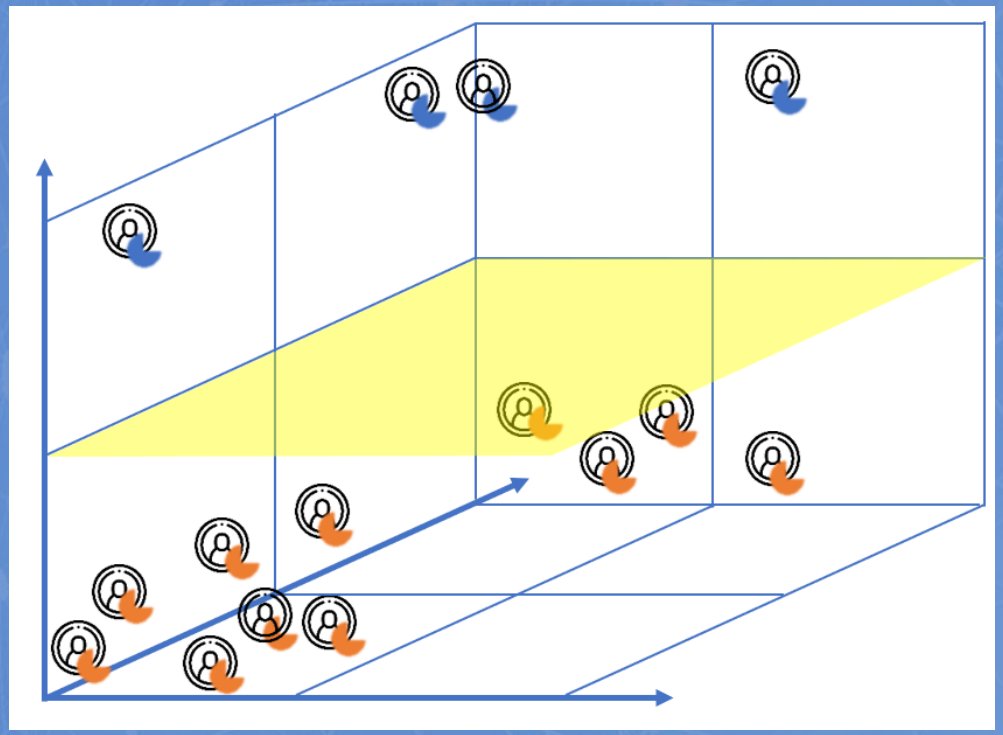
On applique la fonction d'activation; la fonction sigmoïde, ci-contre. Cette fonction a pour but d'établir un seuil d'une valeur où le neurone est stimulé ou pas, s'il est au-dessus d'un certain seuil (ici 0,5) il pourra propager son potentiel.

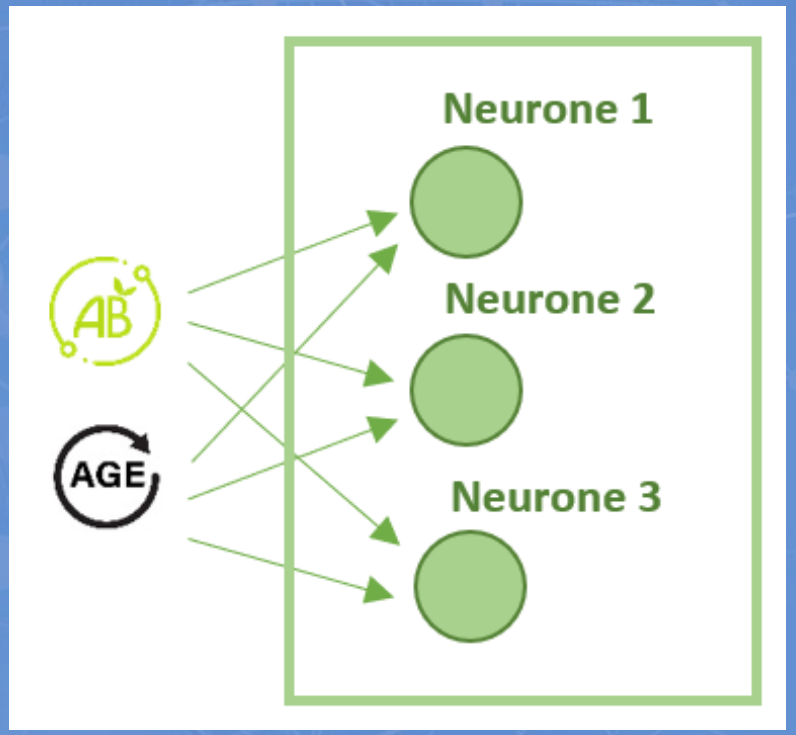
$$\frac{1}{1 + e^{(-(34 - 0.4*40 - 0.3*62))}} = 0.4$$

Avec le seul neurone 1, la sortie vaut 0,4 < seuil 0,5 donc groupe orange

Etape3 - Résolution d’un problème linéairement séparable

On calcule de la même manière la valeur de chaque individu pour les 3 neurones de la couche cachée. On va utiliser les valeurs des neurones cachés comme **nouvelles coordonnées.** Comme nous avons utilisé 3 neurones, la représentation se fait en 3 dimensions (chaque dimension correspond à un des neurones de la couche cachée).
Le problème est devenu linéairement séparable puisqu’on peut définir un plan qui sépare les 2 groupes.

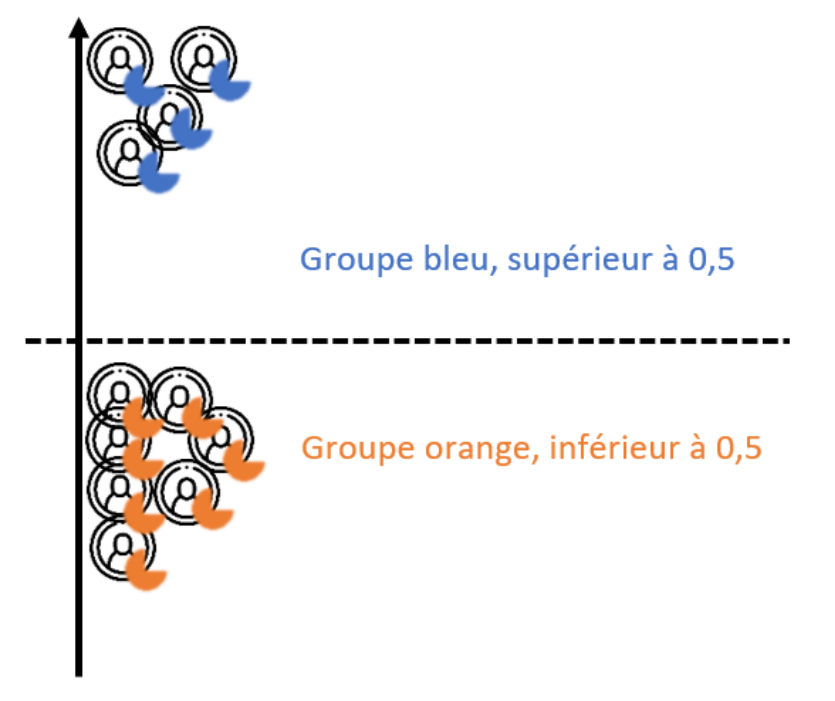


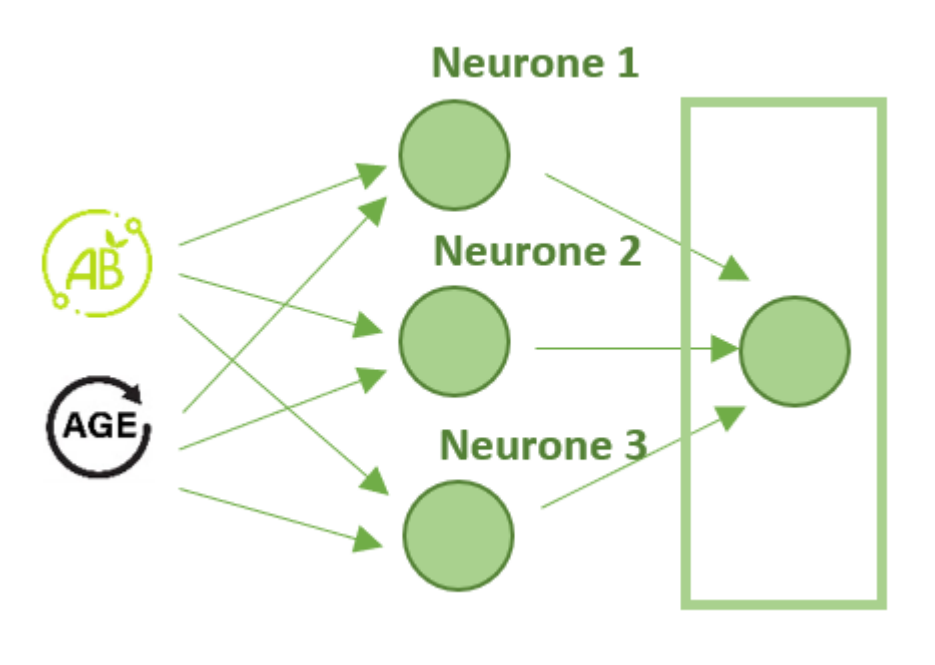


Etape3 - Résolution d’un problème linéairement séparable

Le problème est donc devenu plus facile à résoudre. On va pouvoir ajouter le neurone de sortie qui fera quant à lui une combinaison linéaire des 3 Neurones cachés et appliquera la fonction sigmoïde au résultat. En sortie on obtient une seule valeur pour chaque client, comprise entre 0 et 1.

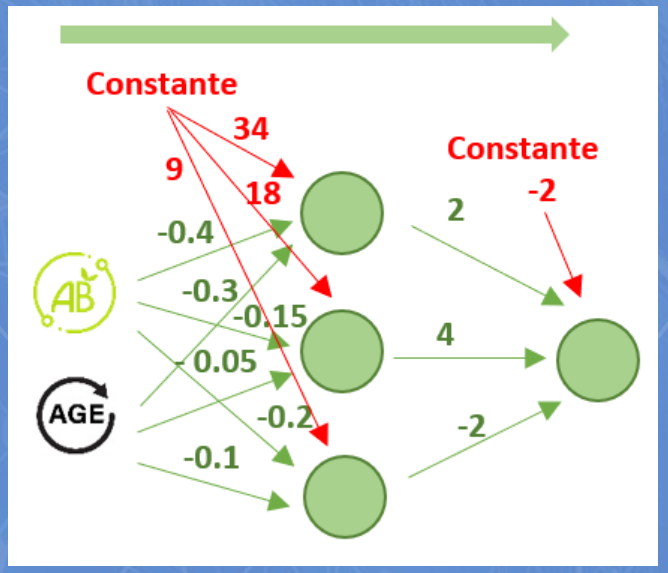
On cherche à connaitre la probabilité d’un client d’appartenir au groupe bleu. Plus la valeur en sortie est proche de 1 plus cette probabilité est élevée. Il ne nous reste donc plus qu’à classer les clients : Ceux qui ont une valeur supérieure à 0,5 seront tagués bleus et les autres seront oranges.

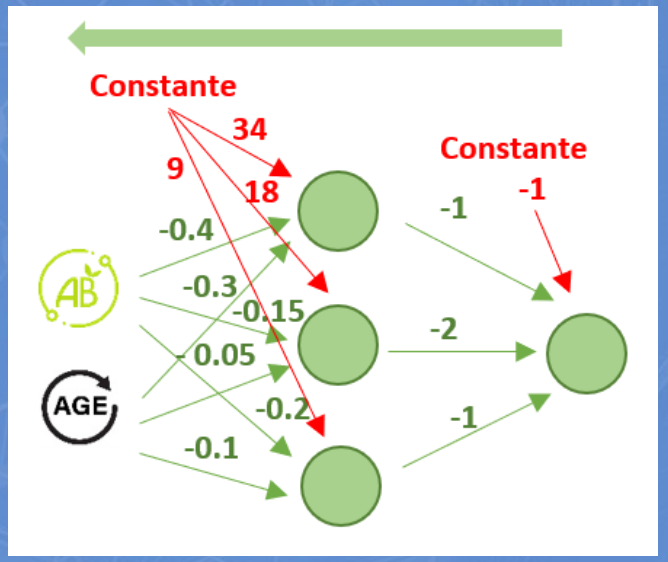




Pour aller plus loin – Rétropropagation

Pour optimiser l'apprentissage d'un ANN, il faut un mécanisme capable de corriger une erreur de prédiction, mais comment un ANN peut évoluer s'il se trompe sur le résultat final ?
Pour cela, on utilise un algorithme qui s'appelle la rétropropagation du gradient (*BackPropagation*). Cette propagation démarre de la dernière couche (Sortie) vers la première (Entrée). L'objectif est de modifier les poids de l'ensemble des connexions entre les neurones.





Initialisation
Pour commencer on choisit aléatoirement tous les coefficients.

Propagation avant
Pour chaque individu on calcule la valeur en sortie du réseau avec les poids actuels.
Dans notre exemple on obtient **0,268** pour un individu dont la valeur à prédire est 0 (le client appartient au groupe orange). S'il ne l'est pas, alors on va effectuer un calcul pour modifier le poids des connexions et la valeur des neurones précédents. Ainsi, le neurone qui est le plus responsable de l'erreur sera susceptible d'être modifié davantage.

Rétropropagation du gradient
L'étape de rétroporopagation va permettre de corriger les poids du neurone de sortie aux neurones précédents en calculant le gradient de l'erreur pour chaque neurone.
Au fur et à mesure de l'apprentissage, le nombre d'erreurs diminue : l'apprentissage est donc fini. On peut enfin utiliser notre ANN et l'utiliser pour calculer des probabilités sur un problème donné.
Dans cette deuxième approximation nous obtenons **0,05,** plus proche de 0.
De ce fait un algorithme peut devenir particulièrement long si vous multipliez le nombre de neurones.

Pour aller plus loin – Rétropropagation

Avantages :

- Traitement de problème variés et complexes, y compris non linéaires
- Peu sensible au bruit ou au manque de fiabilité des données
- Bonne performance

Inconvénients :

- Effet boîte noire, algorithmes plus difficiles à expliquer
- Risque de sur-apprentissage
- Choix des paramètres initiaux