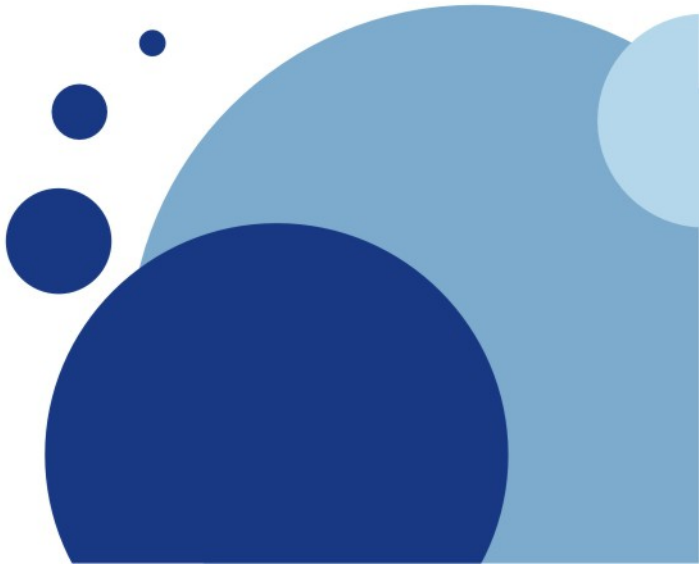
Introduction au deep learning



Programme

I. Intelligence artificielle, machine learning, positionnement, principes, notions clés, exemples.
II. Deep learning : neurone virtuel, descente de gradient, réseaux de neurones multi-couches, rétropropagation du gradient.
III. Toute petite démo.
IV. Exemples concrets d'applications dans le domaine médical.

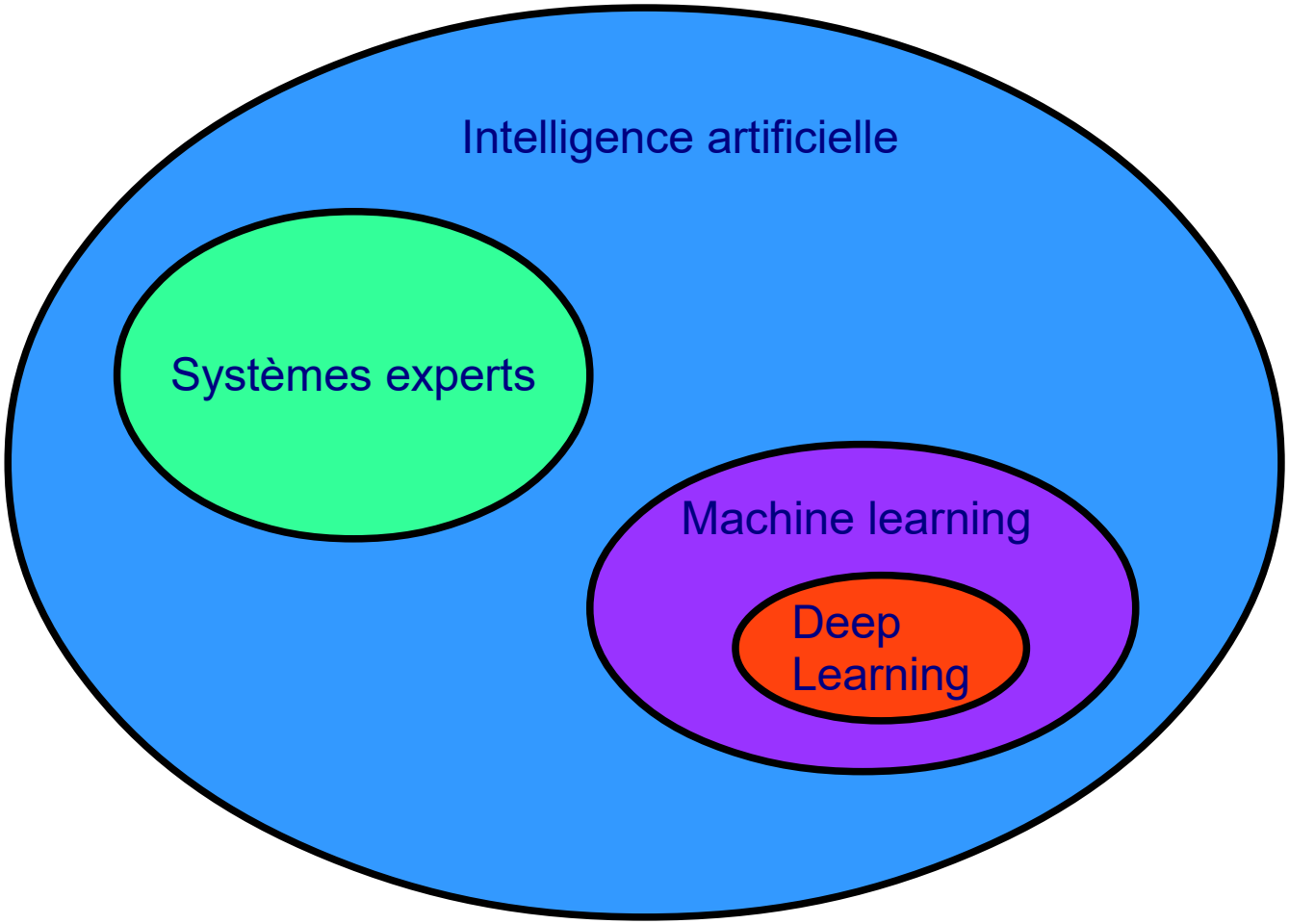
Intelligence Artificielle

De quoi s'agit-il ?

- Construction de programmes informatiques permettant de réaliser des tâches accomplies de manière, à priori, plus satisfaisantes par des êtres humains
  - Car elles demandent des processus mentaux de haut niveau tels que mémoire, connaissance, raisonnement, sentiment
    - Exemple :
      - Jouer aux échecs ou au jeu de go, établir un diagnostic médical à partir de symptômes, déterminer le “sentiment” d'un texte ou d'un discours
  - Car elles sont difficiles à décomposer en séquences de tâches élémentaires que la machine saura exécuter
    - Exemple :
      - Reconnaissance de formes dans une image, reconnaissance du langage naturel, ..

Apprentissage automatique

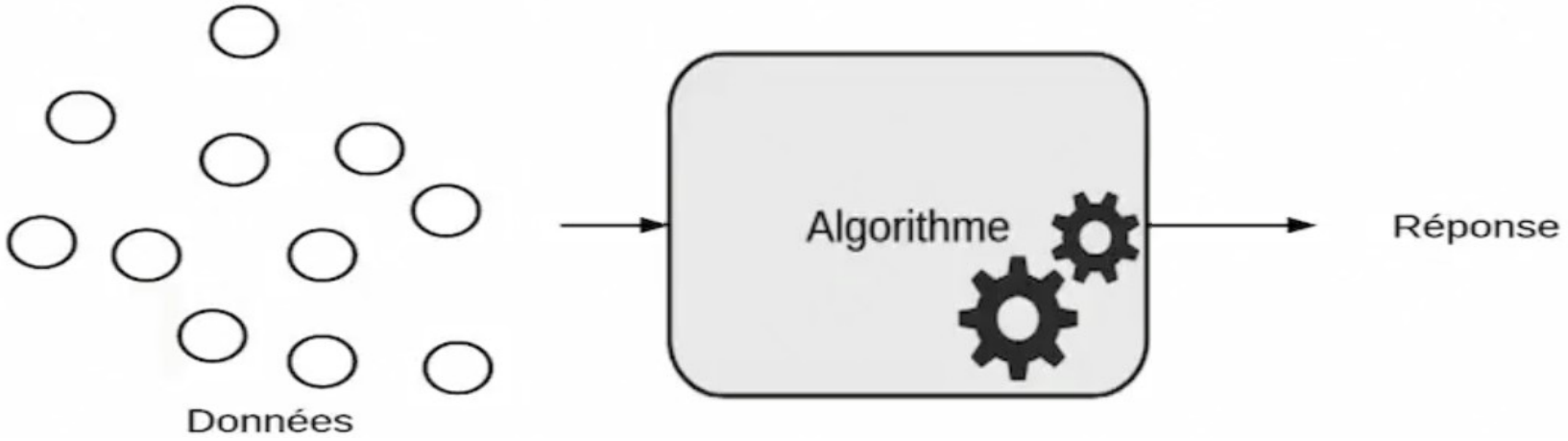
Machine Learning



Machine Learning

Qu'est ce qui le différencie ?

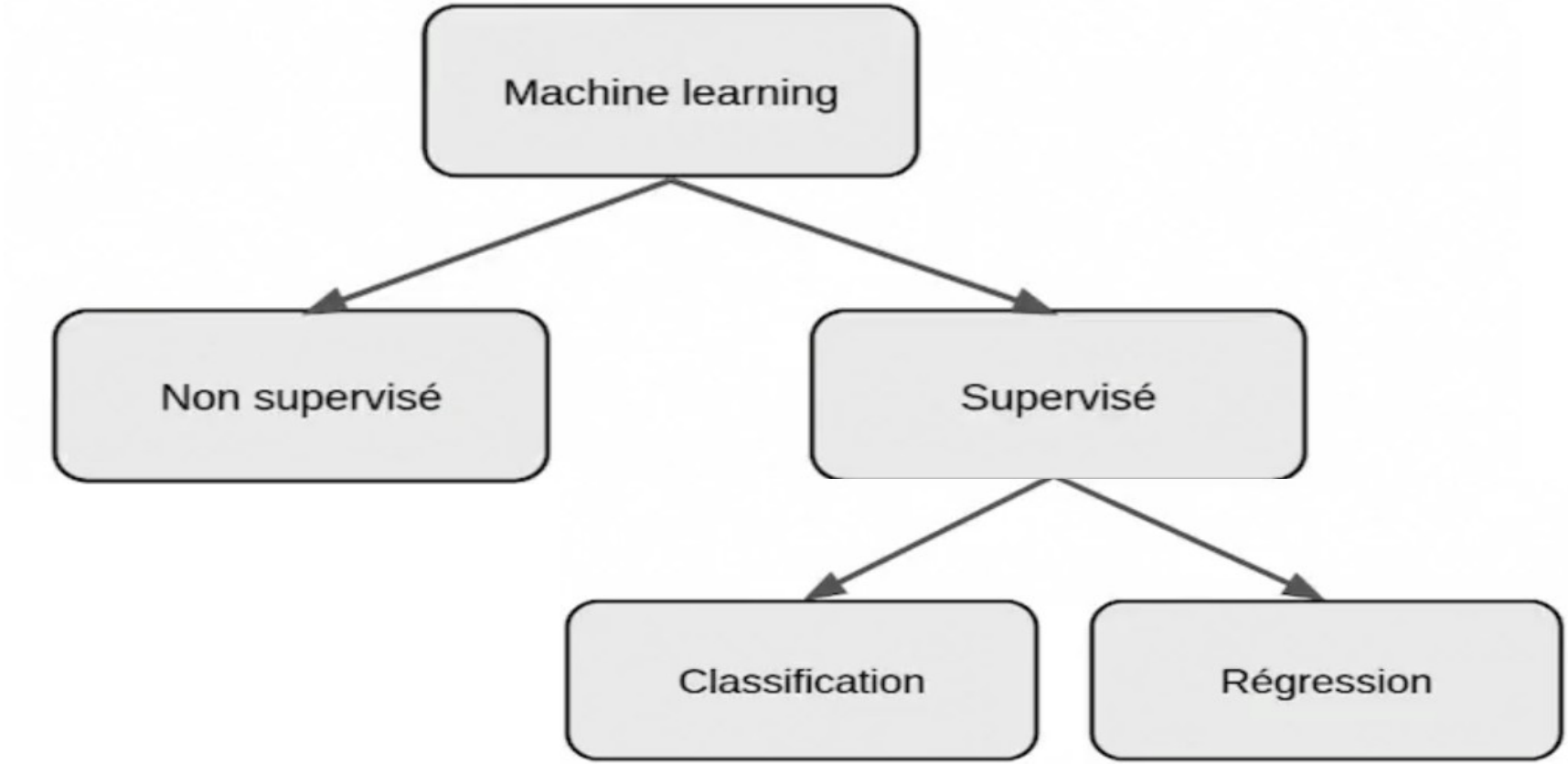
- On ne sait pas décrire comment calculer un résultat à partir des données d'entrée. On demande à la machine d'"apprendre par elle même" à le faire.



- Exemple :
  - Prédire la météo dans les 3 prochaines heures (sachant la température actuelle, l'altitude, la zone géographique, la pression atmosphérique, ...).
  - À partir de l'image d'un grain de beauté indiquer un risque de mélanome
  - Connaitre le comportement futur d'un client d'un site de vente en ligne (sachant l'âge, le genre, le nombre de visites, d'achats effectués, …)
  - …

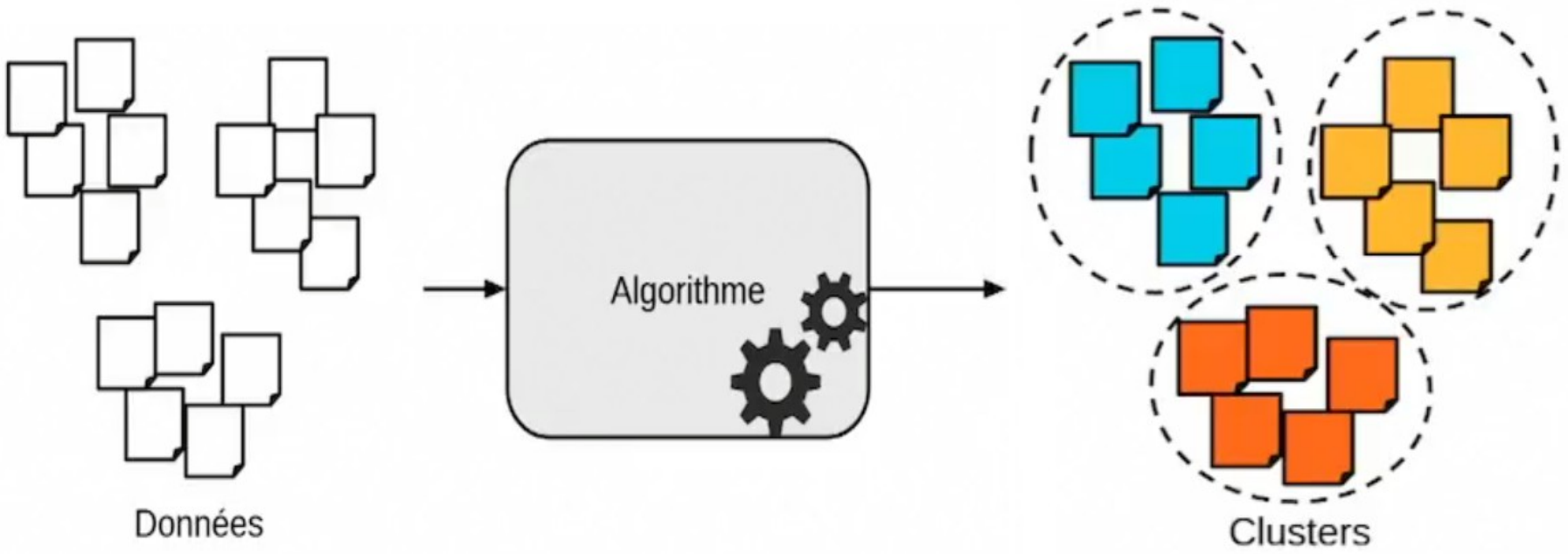
Machine Learning

Un peu de vocabulaire...



Machine Learning non supervisé

"Vois ce que tu peux trouver"



Exemple :

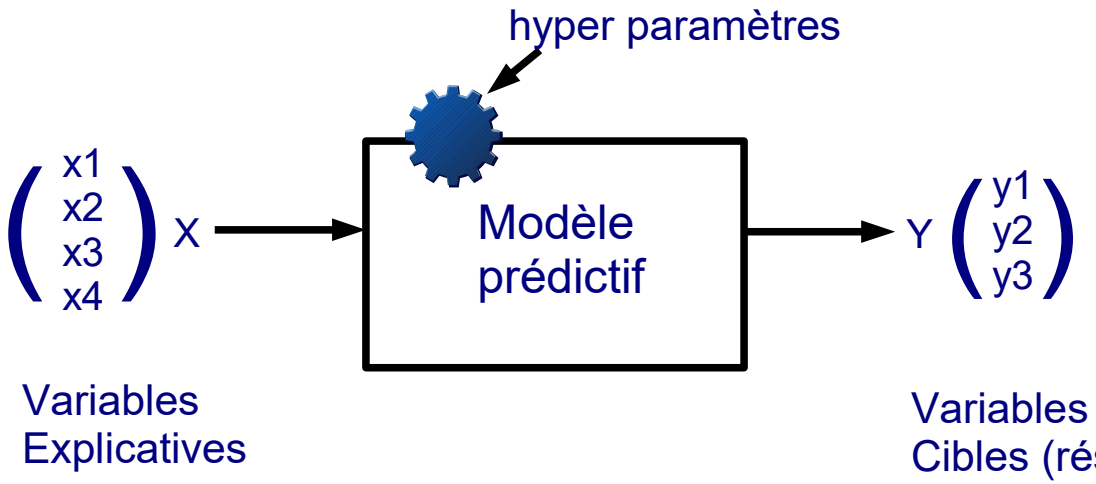
- Algorithme des k moyennes

Machine Learning supervisé

"Apprends à partir des exemples"

Apprentissage supervisé :

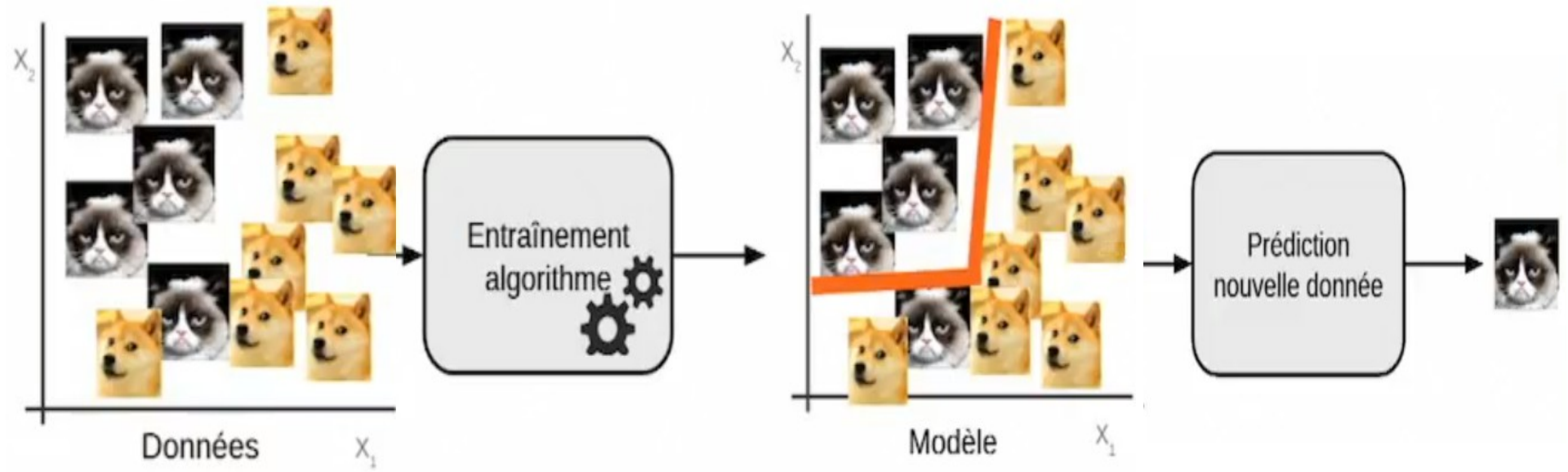
- On dispose d'un certain nombre d'observations pour lesquelles on connait déjà la réponse : les données d'apprentissage.
- Grâce aux données d'apprentissage, l'algorithme fabrique un modèle prédictif.
- On pourra ensuite appliquer le modèle sur de nouvelles observations.



Machine Learning supervisé

"Apprends à partir des exemples"

- Classification = prédire une variable catégorielle
  - OUI/NON, 1/0, chat/chien/souris...

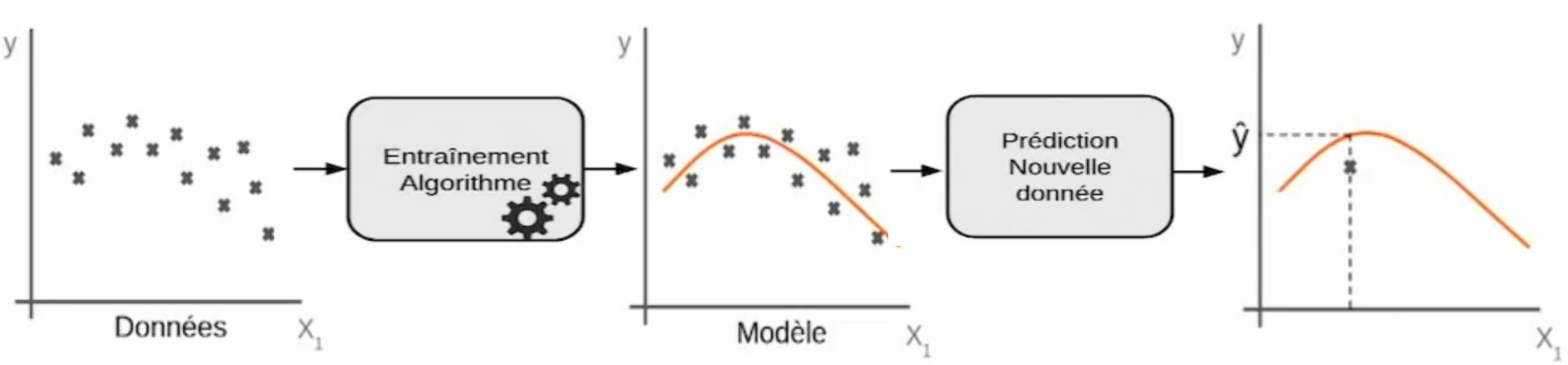


- Exemple : reconnaissance d'objets dans une image, diagnostic, ...

Machine Learning supervisé

"Apprends à partir des exemples"

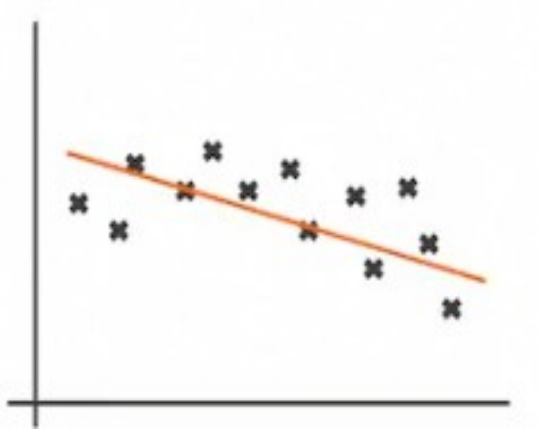
- Régression = prédire une variable quantitative
  - 38, 0.12766, -163.192….



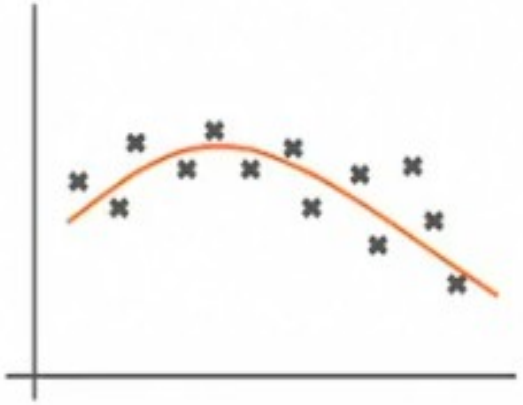
- Exemple : régression linéaire (le modèle est une droite), polynomiale (le modèle est un polynome de degré n),...

Machine Learning supervisé

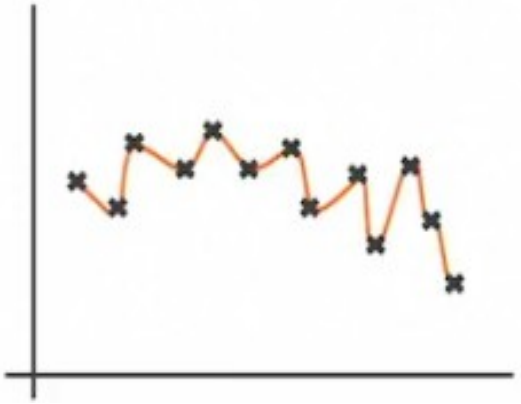
Bien calibrer le modèle



Sous apprentissage



Ce qu'il faut



Sur apprentissage

→ Potentiellement lié aux hyper paramètres du modèle (nombre de neurones dans un réseau de neurones, degré du polynome dans une régression polynomiale, taux d'apprentissage, etc.)

Machine Learning supervisé

Evaluer le modèle

Tous les modèles sont faux mais certains sont utiles
=> Il faut évaluer le modèle

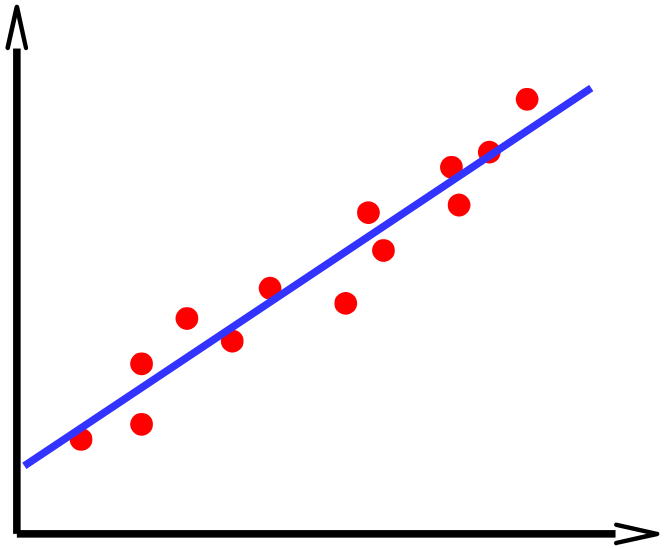
- On met “de coté” une partie des données (entre 20 et 30 %)

- On les utilisera pour évaluer le modèle pendant l'apprentissage ou lorsqu'il est terminé.
  - Pour le modèle ce sont des données “nouvelles”
    - Comparaison de ce que prédit le modèle actuel avec la valeur cible à obtenir
  - Détecter le surapprentissage

Machine Learning supervisé

Exemple : régression linéaire

- Le modèle qu'on recherche est une droite sous la forme Y=aX+b
  - Les données d'apprentissage { x_t, y_t } vont servir à trouver a et b



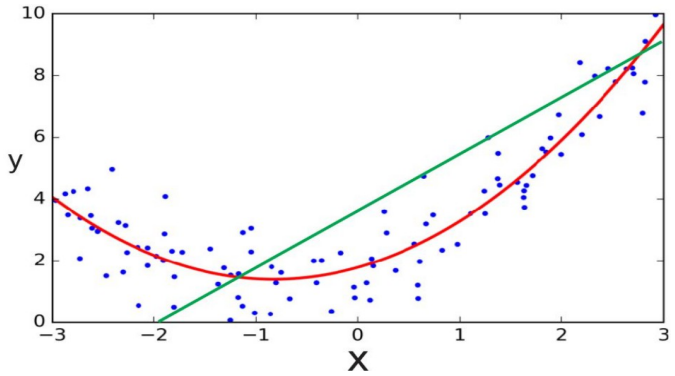
On cherche a et b qui minimisent la distance des points à la droite

Minimisation de la fonction :

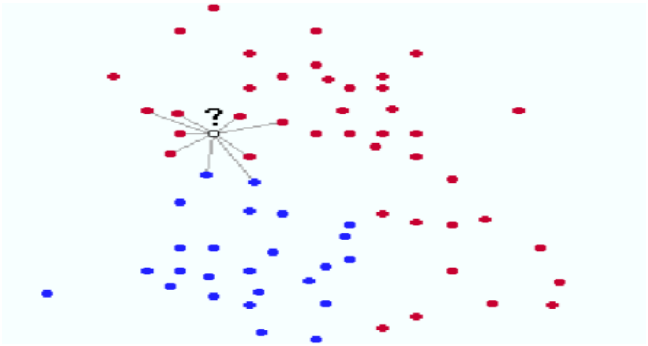
$$Loss_{a,b} = \sum_{i=1}^{p} (y_i - (ax_i + b))^2$$

Machine Learning supervisé

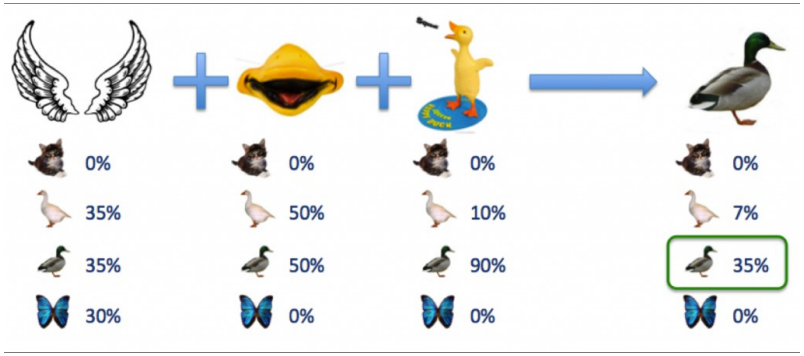
De nombreuses approches sont possibles



Régression polynomiale



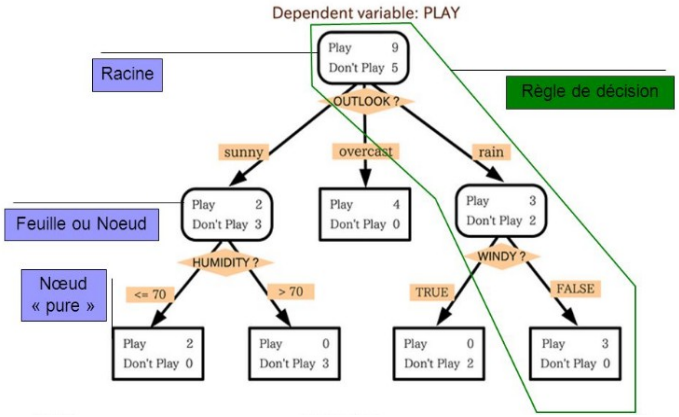
K plus proches voisins (classification)



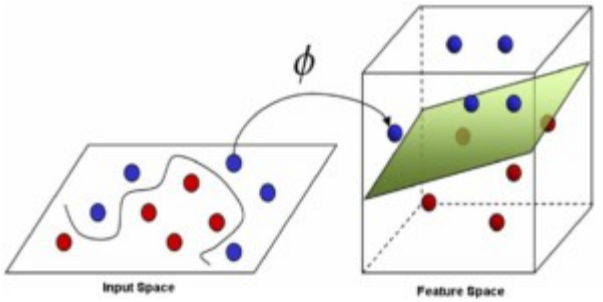
Approche probabiliste : classification bayésienne naïve

Machine Learning supervisé

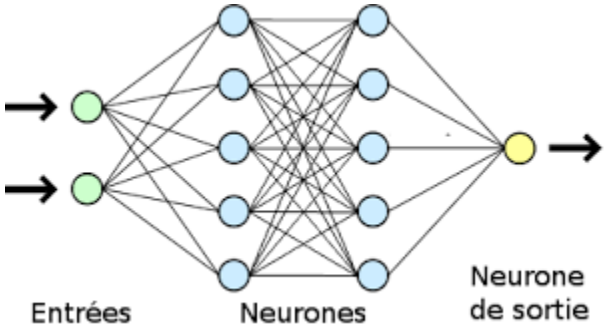
De nombreuses approches sont possibles



Arbres de décision



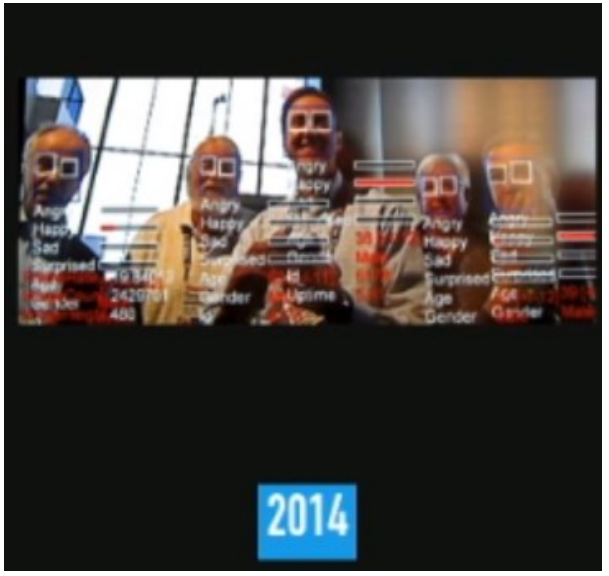
SVM

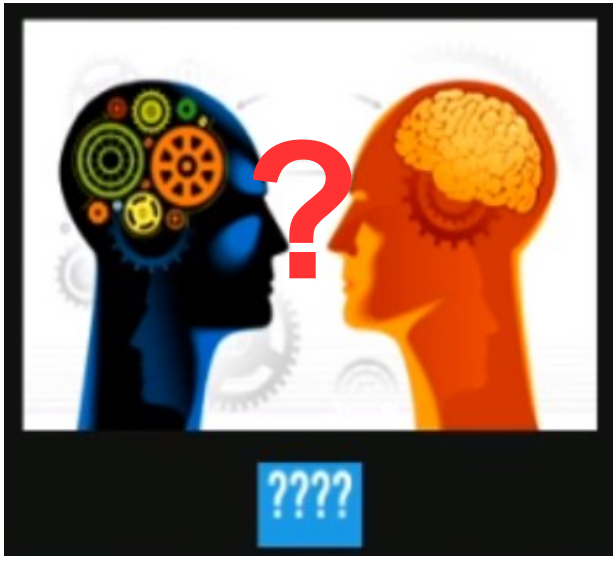


Deep learning = réseaux de neurones profonds

Deep Learning

Origine du buzz médiatique

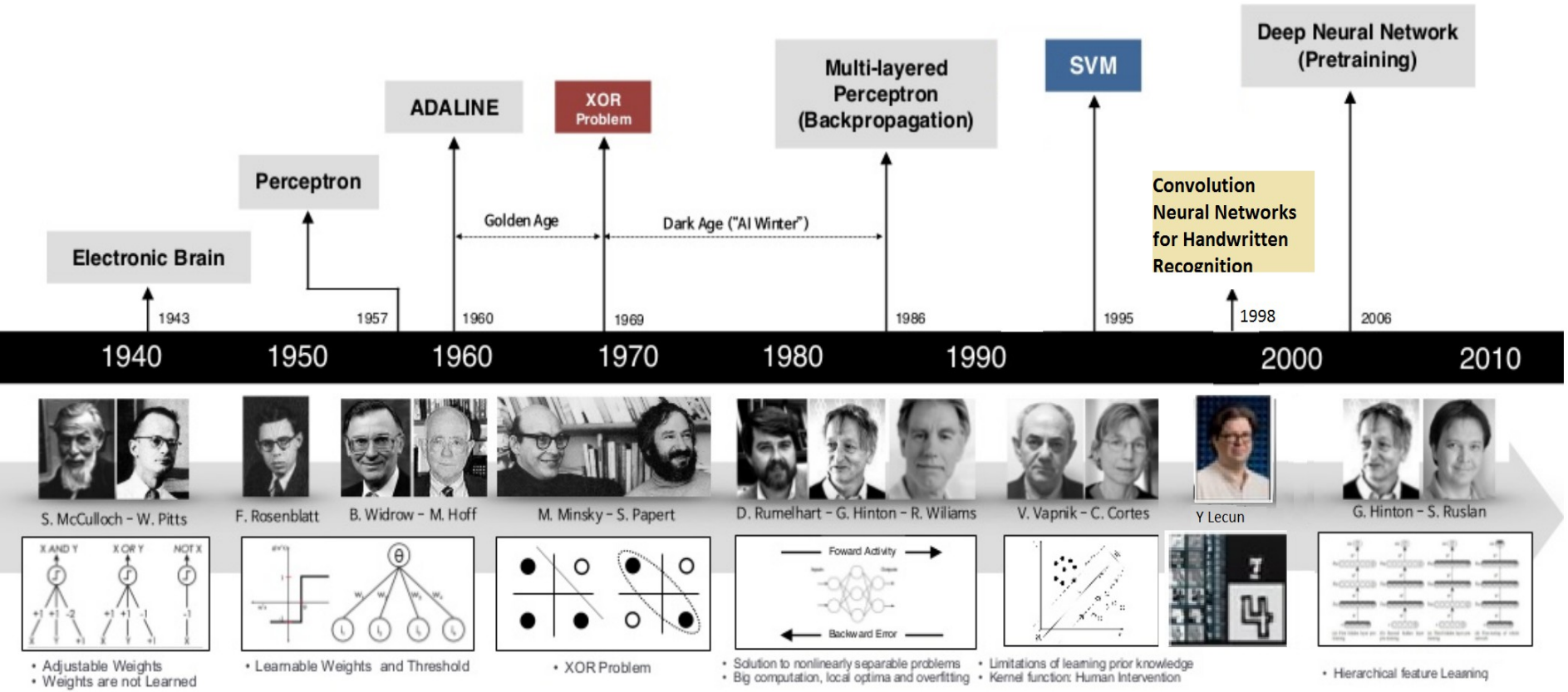




Les réseaux de neurones

De l'histoire ancienne

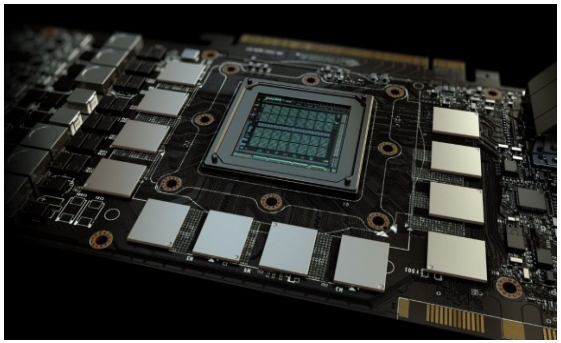
Brief History of Neural Network



Les réseaux de neurones

Pourquoi reviennent-ils "à la mode"

- Gisements de données (bigData)



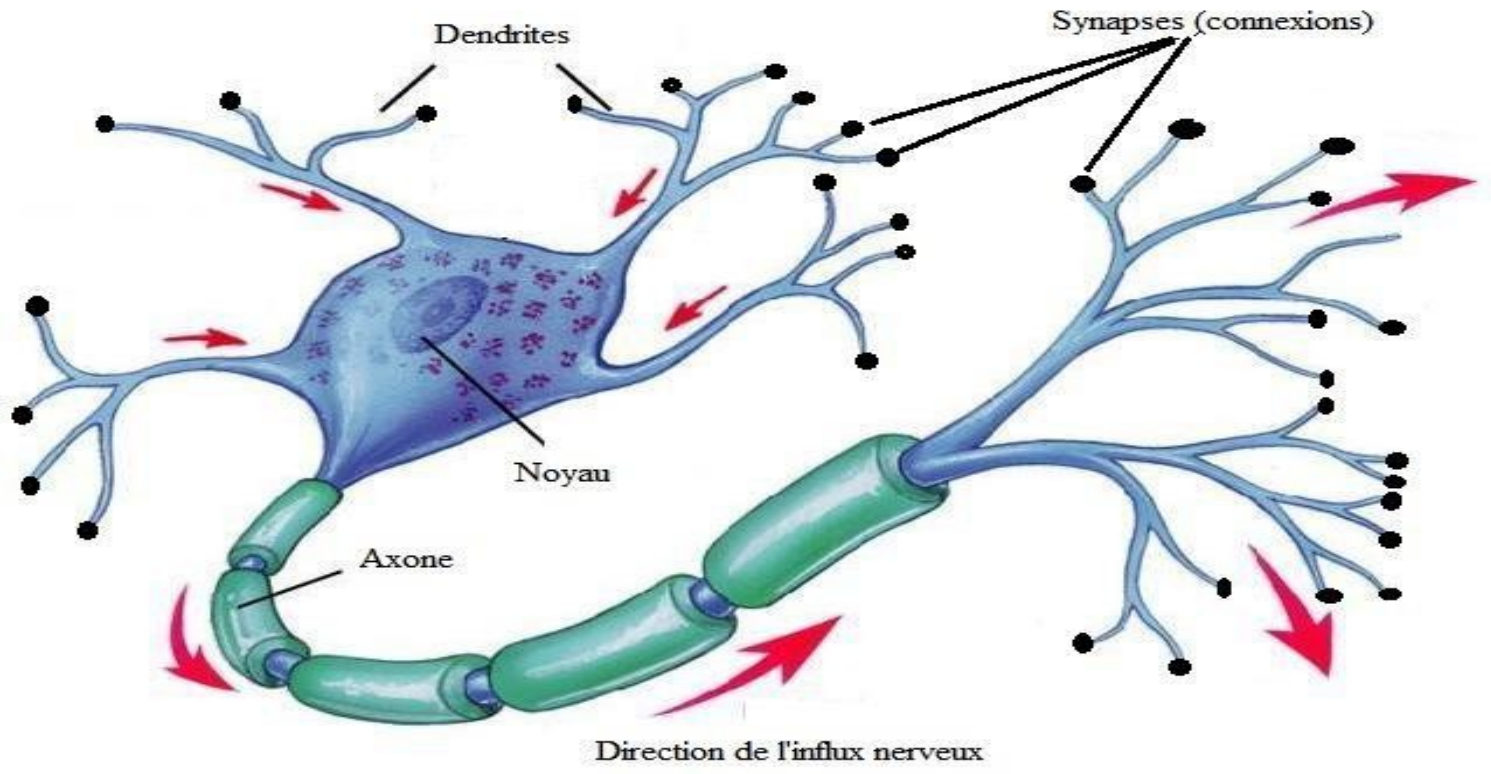
- Puissance de calcul (GPU, ...)

Microsoft Google NVIDIA Baidu 百度

- Nouveaux acteurs (investissements)

Les réseaux de neurones

Le neurone biologique

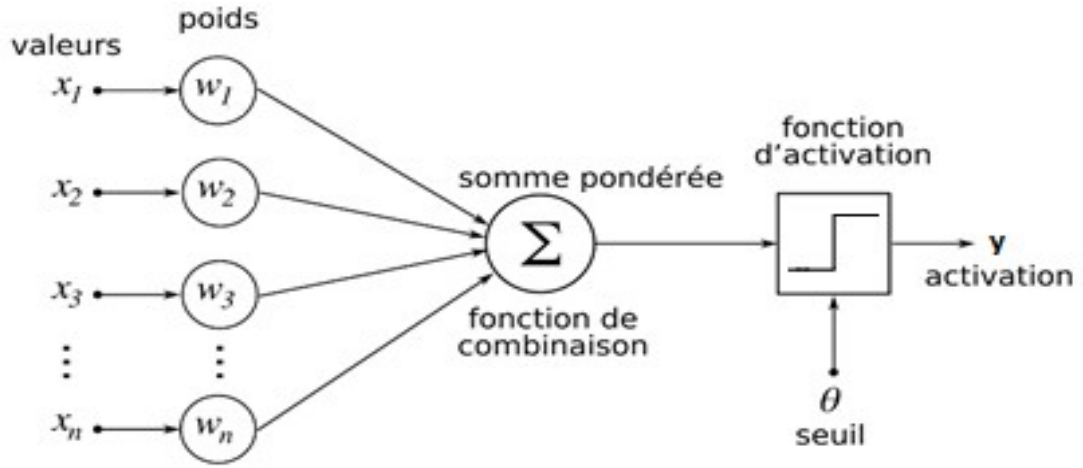


- L'influx nerveux n'est déclenché dans l'axone que si l'intensité globale de la stimulation dans les dendrites dépasse un certain seuil.

Les réseaux de neurones

Le neurone artificiel/numérique

- Le “perceptron” (Rosenblatt 1957) :



- Le comportement du neurone biologique est “mimé” par une fonction mathématique :
  - somme pondérée des entrées suivie de la fonction d'activation, ici = la fonction seuil

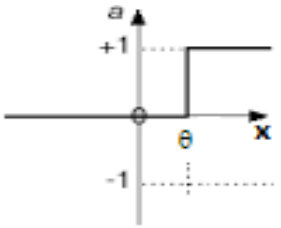
$$w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n \geq \theta \;\; ?? \;\; \Rightarrow \;\; y = 1$$

$$w_1x_1 + w_2x_2 + w_3x_3 + \ldots + w_nx_n < \theta \;\; ?? \;\; \Rightarrow \;\; y = 0$$

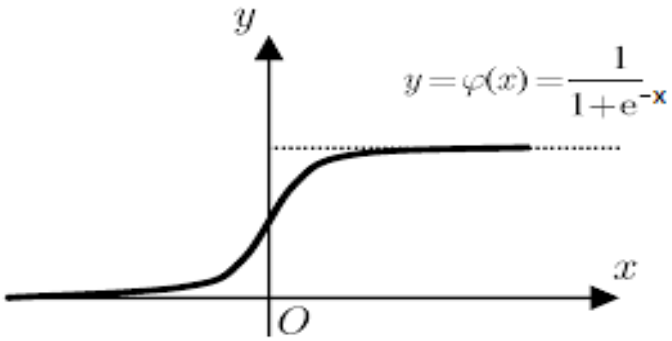
Les réseaux de neurones

Adaptation du modèle pour la classification

- La fonction seuil ne convient pas
  - Non dérivable pour x = θ
  - Trop binaire
    - On aimerait avoir “la probabilité de ...”



- On utilise souvent la fonction sigmoïde comme fonction d'activation

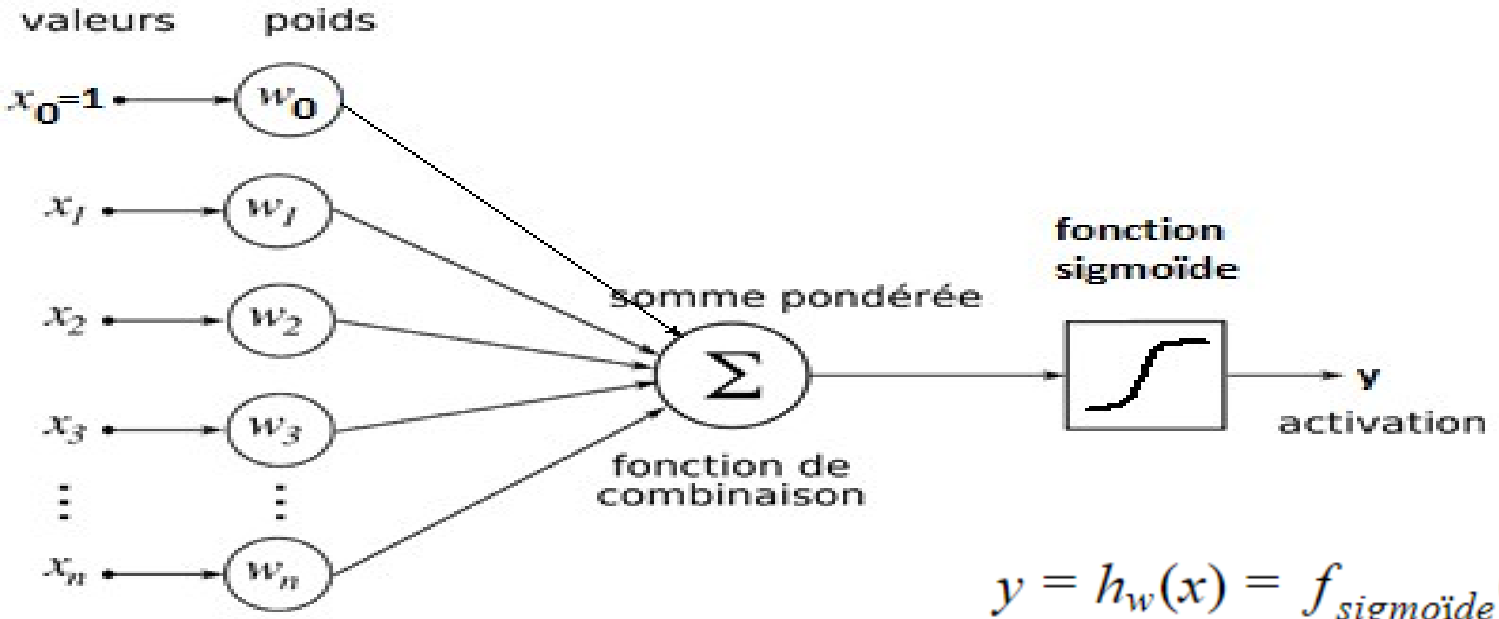


Il existe d'autres fonctions d'activation : tanh, RELU, …

- Le seuil (θ) est remplacé par un poids supplémentaire w_0 (biais) dans la somme pondérée,
  - appliqué sur une entrée fictive x_0=1

Les réseaux de neurones

Que devient notre neurone virtuel ?



$$y = h_w(x) = f_{sigmoïde}(\sum_{i=0}^{i=n} w_i x_i)$$

- un neurone fait :
  - la somme pondérée de toutes ses entrées,
  - ajoute une valeur supplémentaire qui s'appel le biais (w_0)
  - fait passer cette somme dans fonction dite d'activation qui a la particularité d'être non linéaire (sigmoïde, tanh, softmax, relu, ...).
- Les w_i sont les paramètres du neurone. L'apprentissage consistera à les ajuster le plus précisément possible.

Les réseaux de neurones

Exemple d'application ultra simpliste

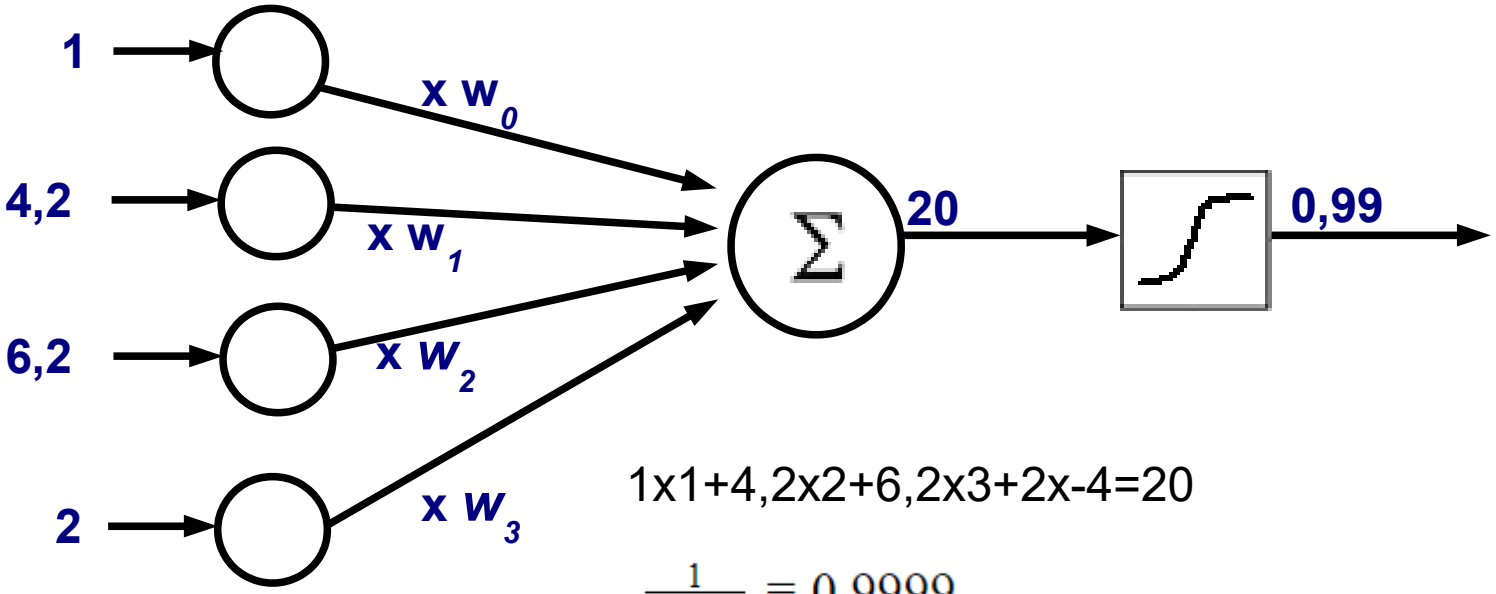
- Numération sanguine.
  - Objectif : fournir une alerte quand la numération fait apparaître une anomalie
  - *Simplification pour l'exercice : on ne tient compte que des hématies, des leucocytes et du sexe...*

Hématies (en millions/mm3)3,1	Leucocytes (en milliers/mm3)	Sexe (H=1/F=2)	Alerte ? (oui=1, non=0)
...	...	...	...
...	...	...	...
3,1	5,3	1	1
4,2	6,2	2	0
4,6	7,5	2	0
1,9	4,4	2	1
...	...	...	...

Les réseaux de neurones

Exemple d'application ultra simpliste

- Initialisons les poids aléatoirement
  - $w_0=1$, $w_1=2$, $w_2=3$, $w_3=-4$



1x1+4,2x2+6,2x3+2x-4=20

$$\frac{1}{1+e^{-20}} = 0.9999$$

- On obtient $h_w(x) = 0,9999$. La valeur souhaitée y_c est 0 !
- Il faut modifier les poids mais comment ?

Les réseaux de neurones

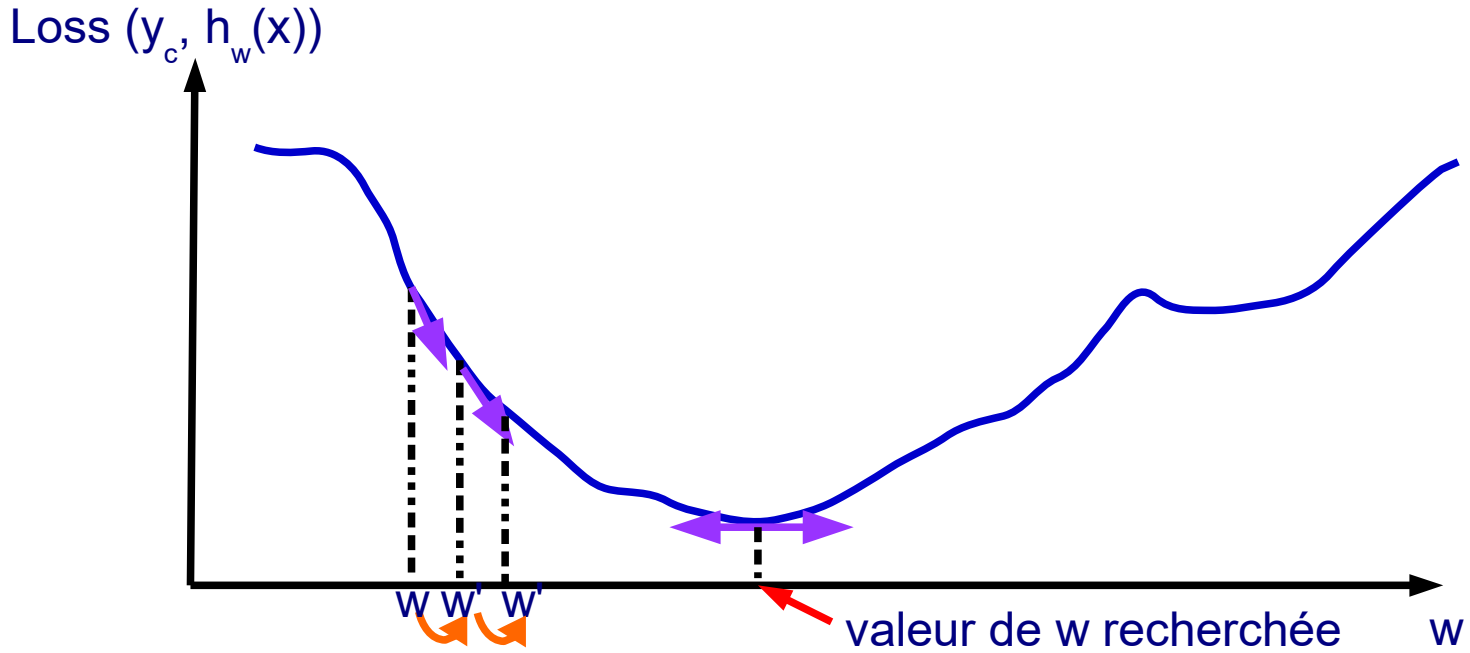
Comment le neurone apprend-t-il ?

- Comment modifier les valeurs de w_i pour améliorer la prédiction?
  - Pour chaque donnée d'entrainement on va chercher à minimiser l'erreur = la distance entre la cible y_c et la prédiction $h_w(x)$ fournie par le neurone
  - On définit une fonction Loss $(y_c, h_w(x))$ pour caractériser cette erreur (perte).
  - Exemple :
    - Régression : carré de la différence :
      - Loss $(y_c, h_w(x)) = ½ (y_c - h_w(x))^2$
    - Classification : Entropie croisée
      - Loss $(y_c, h_w(x)) = - y_c \log(h_w(x)) – (1-y_c) \log(1 – h_w(x))$

Les réseaux de neurones

Comment le neurone artificiel apprend-t-il ?

- Courbe de variation de la perte en fonction de w

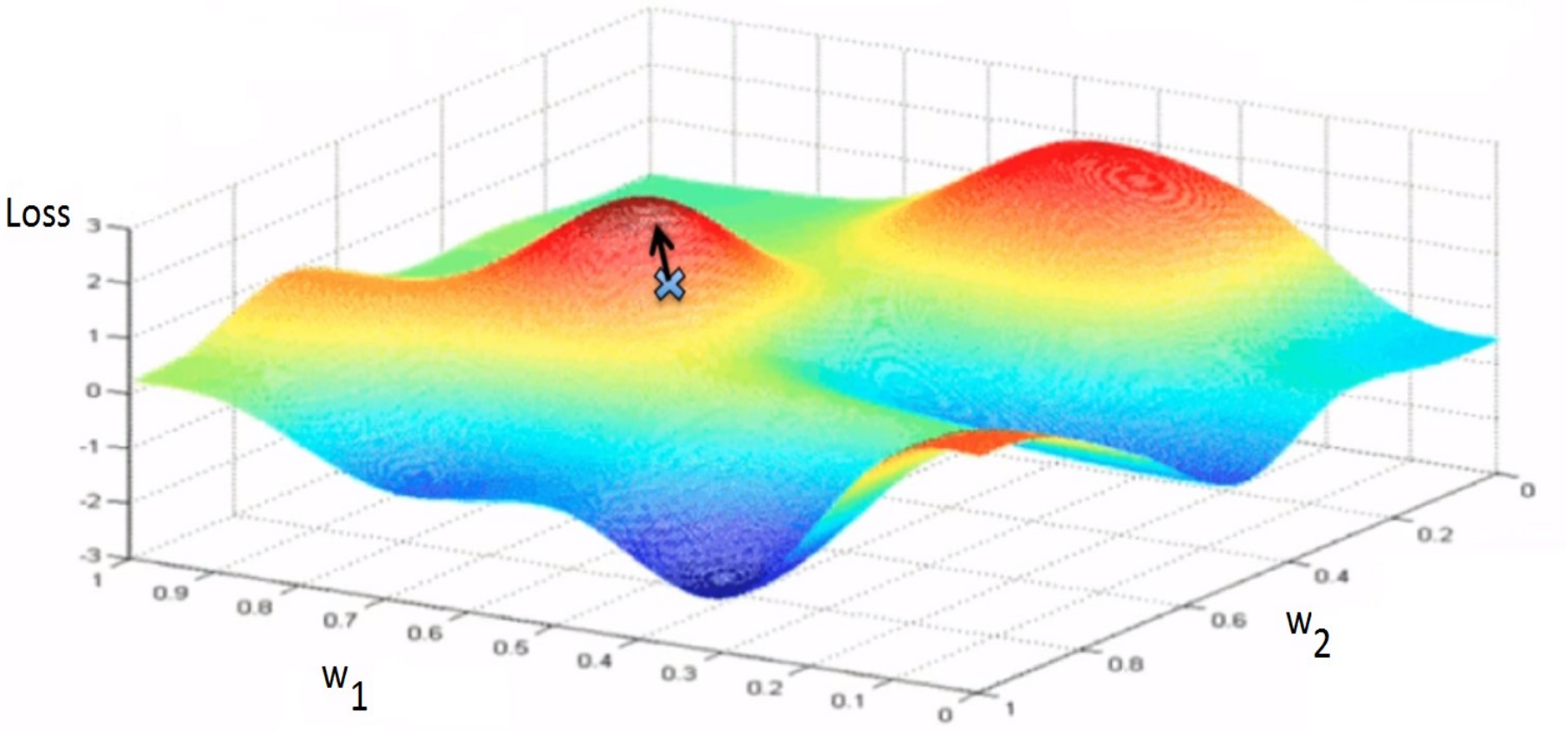


- On calcule la valeur de la dérivée dLoss/dw (gradient : $\delta Loss_w$). Elle nous indique comment faire évoluer w
  - $w' = w - \alpha * \delta Loss_w$, α est un hyper paramètre appelé taux d'apprentissage.
- Lorsque le minimum est atteint, le gradient ~= 0

Les réseaux de neurones

Descente de gradient en n dimensions

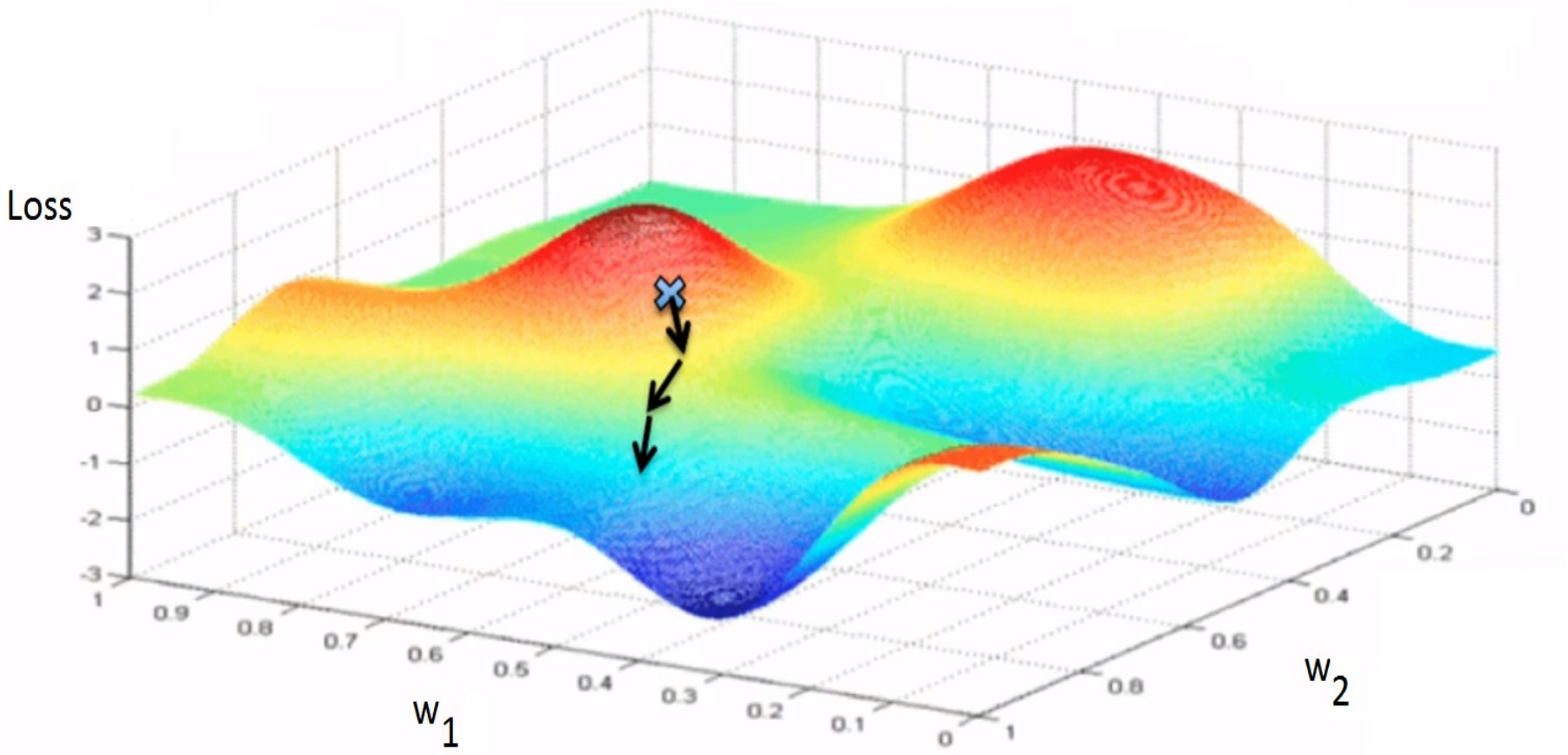
- Le gradient $\overrightarrow{\delta Loss_w}$ donne la direction (vecteur) ayant le taux d'accroissement de la fonction le plus élevé.



Les réseaux de neurones

Descente de gradient en n dimensions

- La direction opposée au gradient nous donne la direction à suivre pour minimiser la fonction



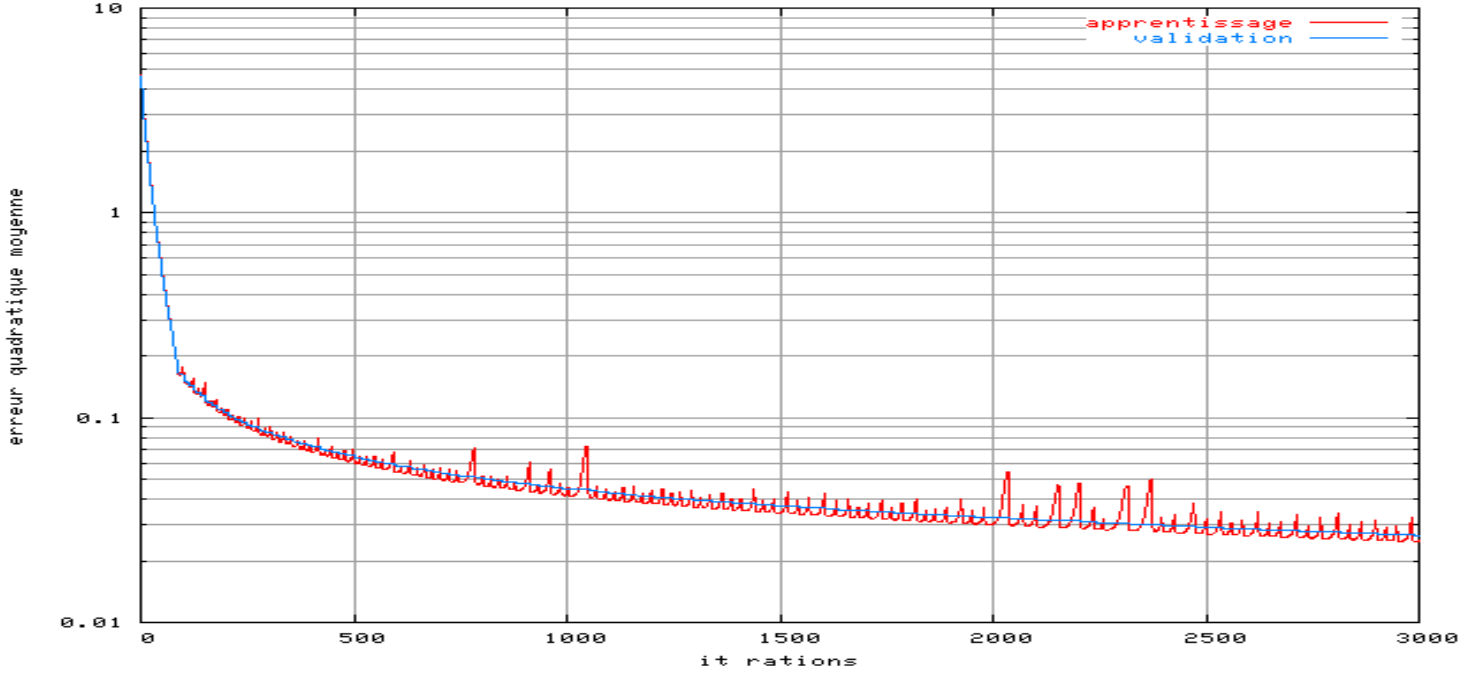
Les réseaux de neurones
Apprentissage du perceptron

- Algorithme de la descente de gradient stochastique
  - Préalable : séparer ensemble des données d'apprentissage et ensemble des données de tests qu'on réserve pour l'évaluation.
  - Initialiser les poids w_i aléatoirement
  - Tant que “apprentissage non terminé”
    - Choisir un nouvel exemple (x_t, y_t) dans les données d'apprentissage;
    - Evaluer la prédiction par le perceptron avec le poids dont on dispose;
    - Evaluer le gradient $\delta Loss_{wi} = dLoss(y_t, h_w(x_t))/dw_i$ pour tous les w_i;
      - Dans le cas de la classification : $\delta Loss_{wi} = -(y_c - h_w(x)) * x_i$ (1)
    - Modifier les poids :
      - $w_i' = w_i - \alpha * \delta Loss_{wi}$,
        - » α = taux d'apprentissage (au départ hyper paramètre) qu'on peut faire décroître au cours du temps
    - Evaluer le nouveau modèle (perceptron avec les nouveaux poids w_i') sur l'ensemble de tests.

(1) Preuve : http://mediamining.univ-lyon2.fr/people/guille/m4101/regression_logistique.html

Les réseaux de neurones
Courbe d'apprentissage *du perceptron*

- On itère un grand nombre de fois en ajustant les poids
- Quand tout se passe bien (convergence), l'erreur baisse avec le nombre d'itérations
  - Diminution de la perte moyenne sur les données d'apprentissage
  - Diminution des erreurs sur les données de tests (validation)



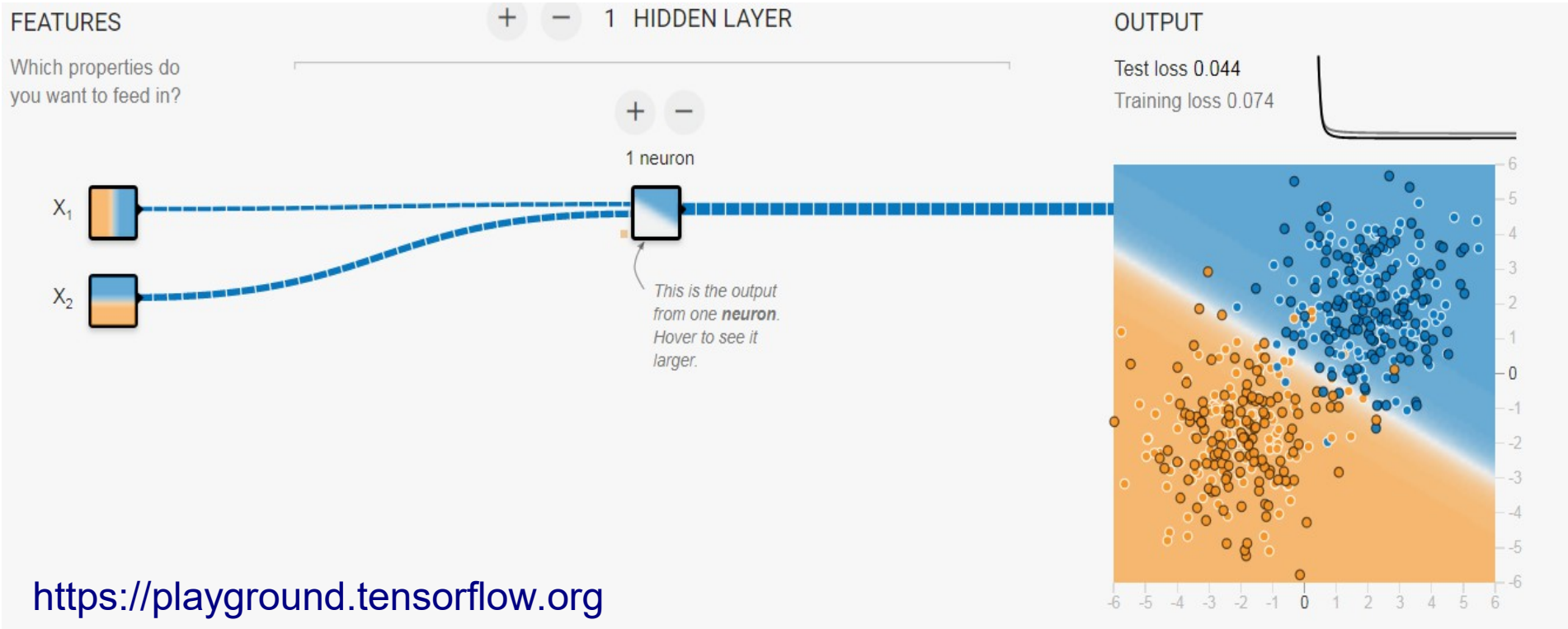
Les réseaux de neurones

Apprentissage du perceptron

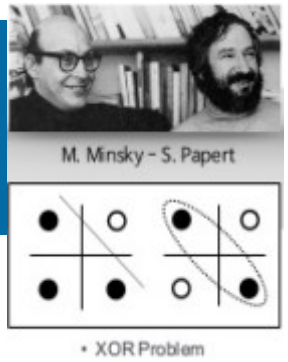
- Quand décide-t-on d'arrêter l'apprentissage ?
  - (en fonction des résultats de l'évaluation du perceptron sur les données de tests)
  - L'apprentissage est terminé quand :
    - L'évaluation du modèle sur les données de tests donne un résultat satisfaisant (moins de n% d'erreurs), n décidé à l'avance
    - L'évaluation ne progresse plus
      - L'algorithme ne converge pas
      - L'algorithme converge mais vers un taux d'erreurs qui reste trop élevé....

Les réseaux de neurones
Limites d'un seul neurone

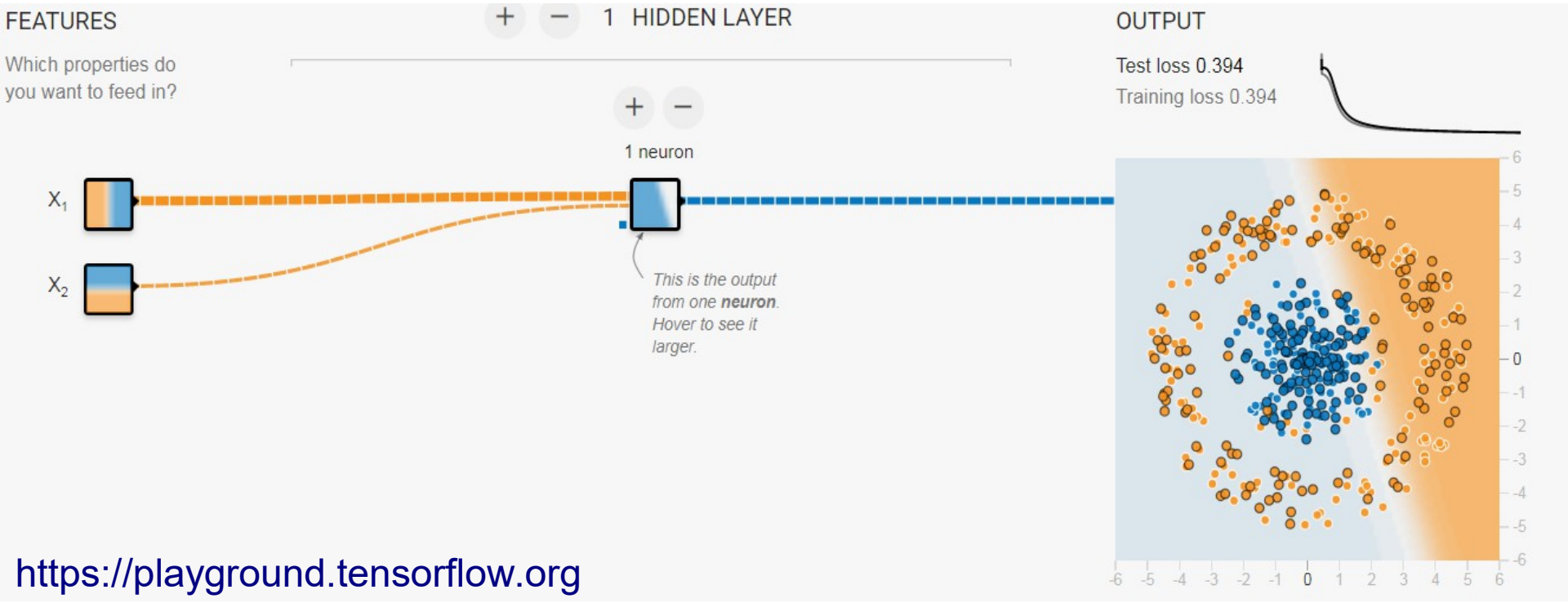
- Le perceptron seul est un classificateur linéaire
  - *Cherche à établir un séparateur linéaire (droite, plan, hyperplan) entre les deux classes (par exemple 1 et 0)*



Les réseaux de neurones
Limites d'un seul neurone



- Le perceptron tout seul est un classificateur linéaire
  - *Dans la plupart des cas les données ne sont pas linéairement séparables => çà ne marchera pas*
  - *Exemple : problème d'analyse des numérations sanguines*

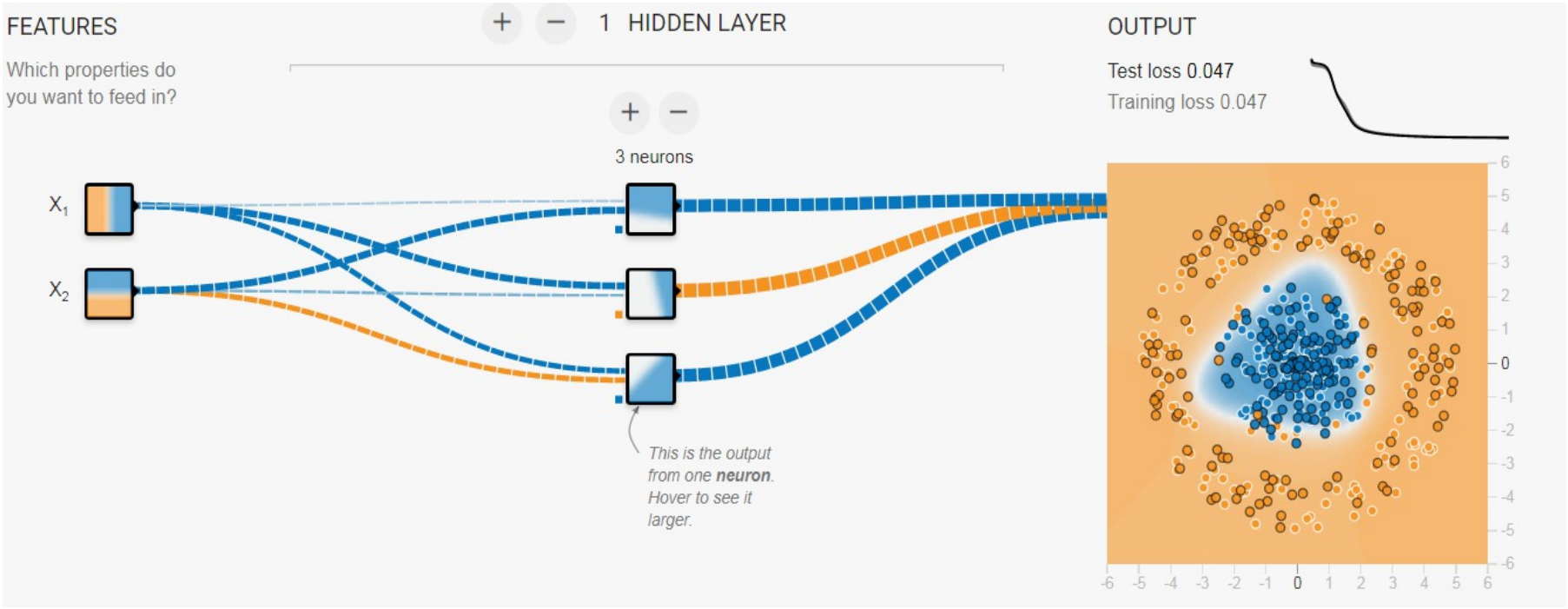


https://playground.tensorflow.org

Les réseaux de neurones

Les réseaux de neurones multi couches (MLP)

- En utilisant plusieurs neurones interconnectés, on va pouvoir résoudre des problèmes plus complexes

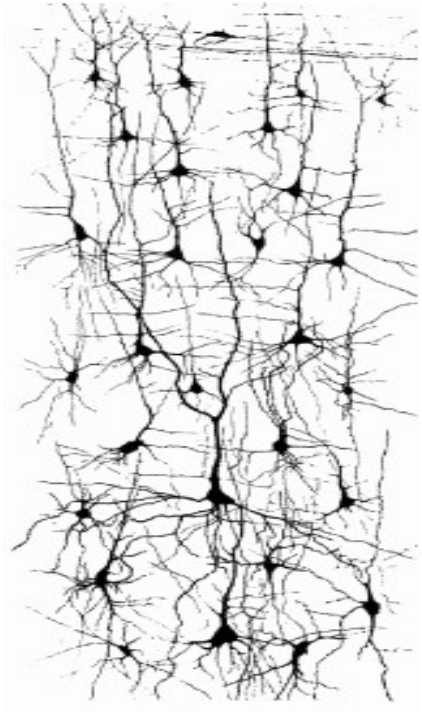


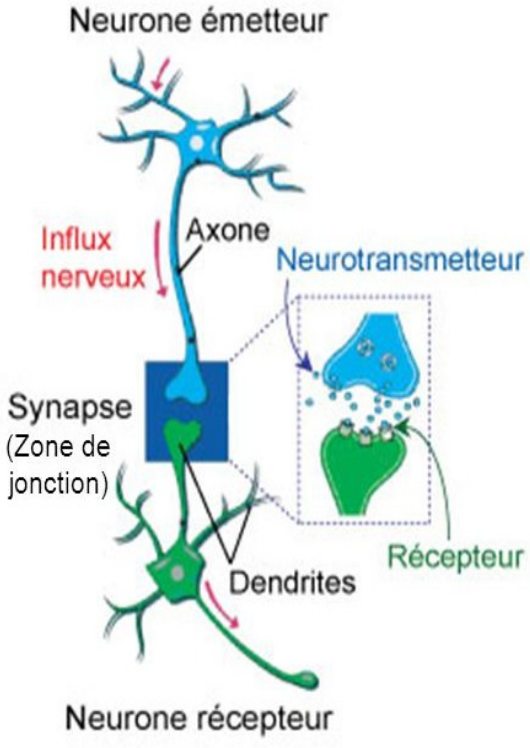
- Ci dessus la couche dite “cachée” permet de réaliser des sous ensemble qui vont se combiner

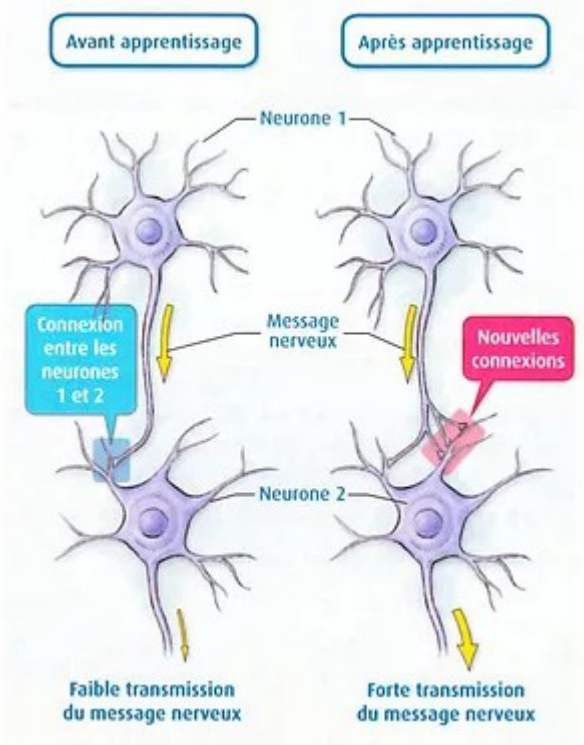
Les réseaux de neurones

Les réseaux de neurones multi couches (MLP)

- Analogie avec la biologie :
  - Dans le cerveaux les neurones sont interconnectés au sein de circuits neuronaux.
  - Les connections s'effectuent au niveau des synapses / dendrites



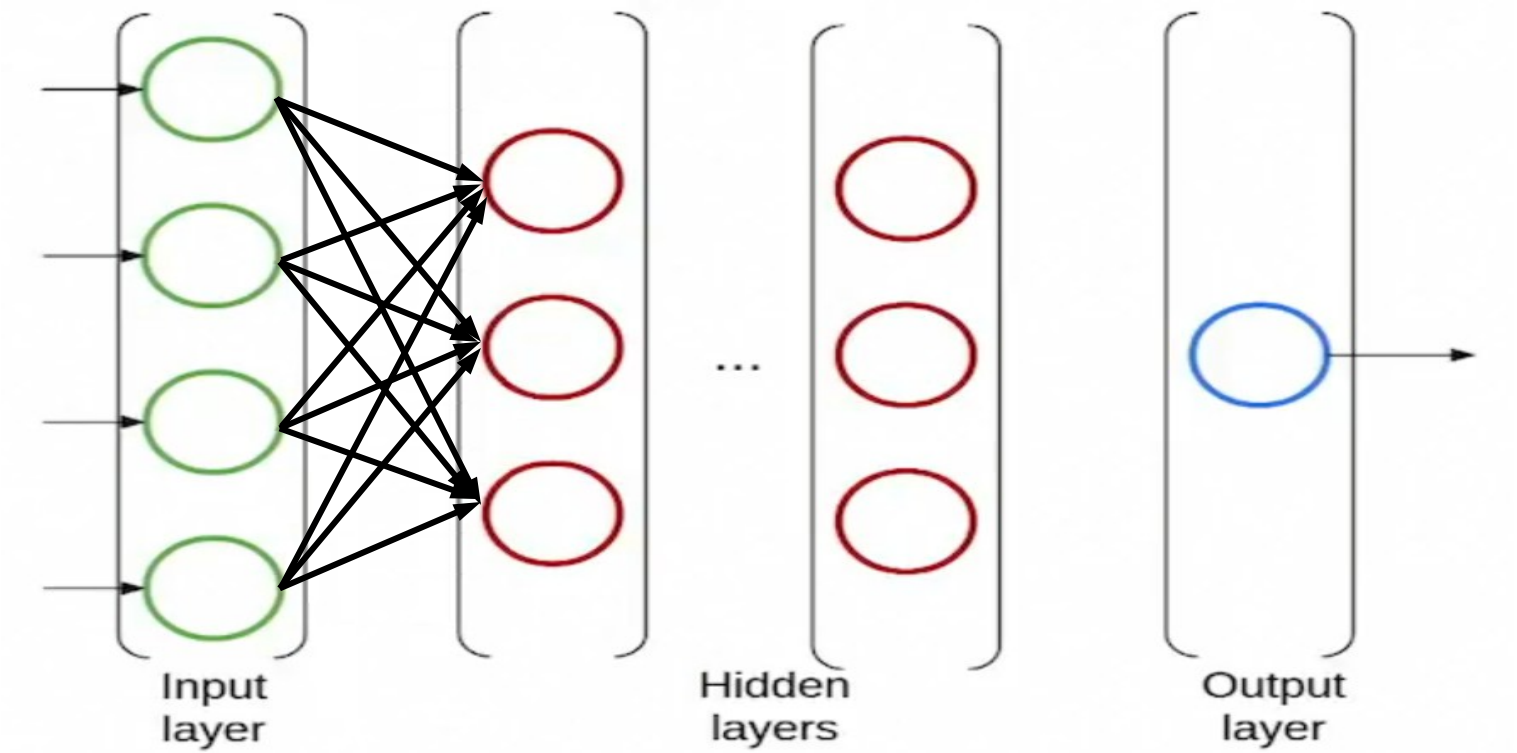




Les réseaux de neurones

Les réseaux de neurones multi couches (MLP)

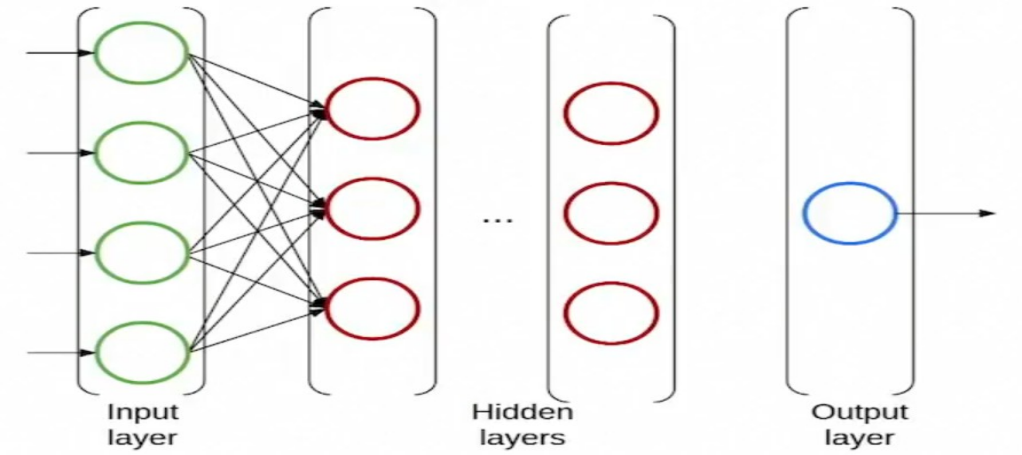
- Réseau de neurones artificiels



- “Fully connected”
  - Chaque neurone de chaque couche (hors entrée) réalise la somme pondérée des sorties de tous les neurones de la couche précédente + la fonction d'activation

Les réseaux de neurones

Les réseaux de neurones multi couches (MLP)



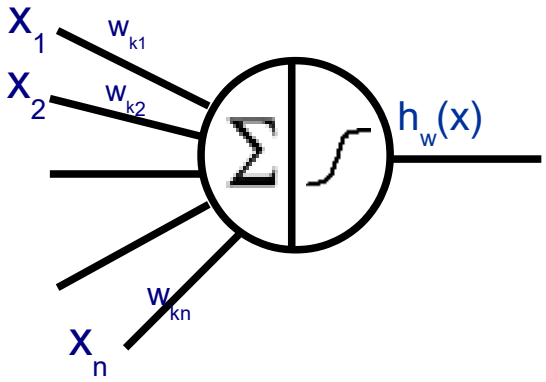
- Feed-forward = propagation avant
  - On calcule les “sorties des couches” les unes après les autres.
- Il peut y avoir beaucoup de neurones dans chaque couche
- Il peut y avoir beaucoup de couches cachées
- Beaucoup de calculs (produits de matrices, ...), et beaucoup de poids à apprendre. Il faudra donc
  - un très “vaste” ensemble de données d'apprentissage (big data)
  - de la puissance CPU/GPU
  - Un temps d'apprentissage long

Les réseaux de neurones

Comment le réseau apprend-t-il ?

- L'apprentissage consistera à affiner progressivement tous les poids du réseau en s'appuyant sur les résultats obtenus avec les données d'apprentissage.
- L'erreur (ou distance) entre le réultat obtenu et la cible à obtenir est quantifiée par une fonction de perte.
- Comme avec un seul neurone, on cherche à minimiser la fonction de perte en faisant varier les poids dans la bonne direction (descente de gradient).
- Le calcul du gradient est “facile” sur la couche de sortie (idem perceptron)

{x_i} : sorties de la couche k-1

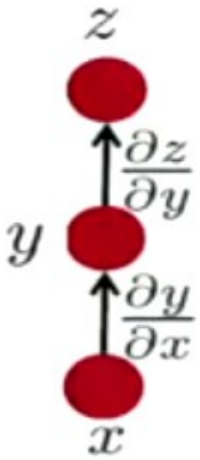


classification : $\delta Loss_{wi} = -(y_c - h_w(x)) * x_i$

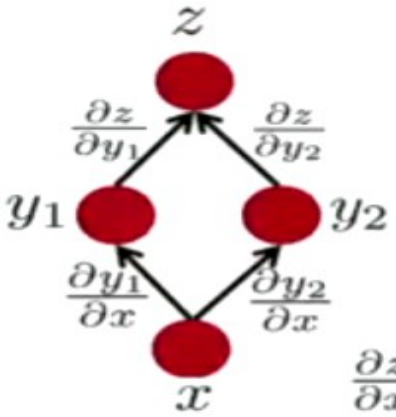
Les réseaux de neurones

Calcul du gradient pour les autres couches?

- Intuitivement : les poids de la couche i-1 contribuent à l'erreur constatée à la couche i :
  - En fonction des données d'entrée de la couche i-1
  - En fonction de l'ensemble des poids appliqués entre la couche i-1 et la couche i.
- Plus précisément : on va propager le calcul du gradient de l'avant vers l'arrière (couche i → couche i-1)
  - en s'appuyant sur les règles de dérivation en chaine d'un graphe de calcul.



$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}$$



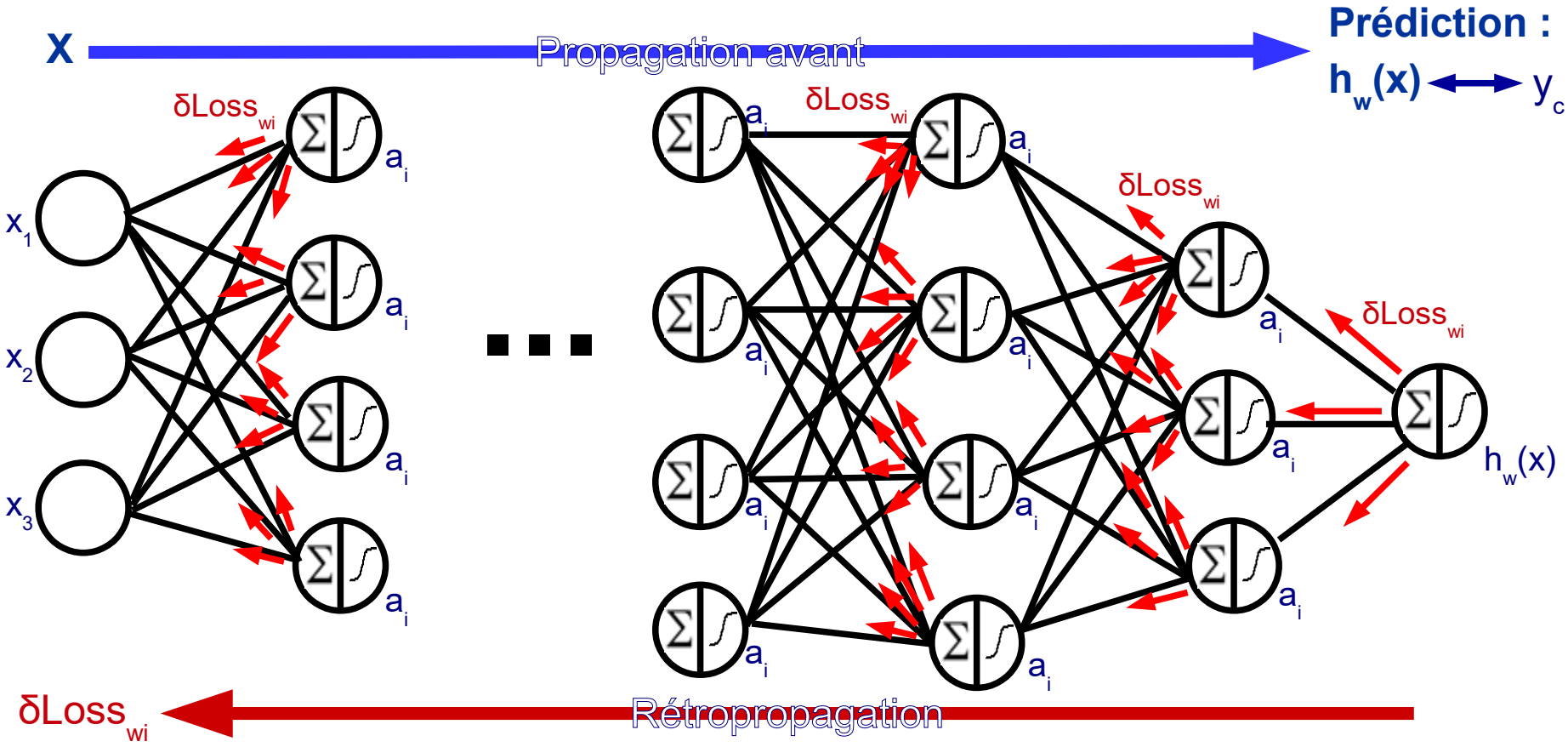
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y_1}\frac{\partial y_1}{\partial x} + \frac{\partial z}{\partial y_2}\frac{\partial y_2}{\partial x}$$

Yoshua Bengio : https://www.youtube.com/watch?v=_mtPw0Uq2b0

Les réseaux de neurones

La rétropropagation du gradient

- Concept clé du deep learning
- Rétropropagation = propagation du gradient de la dernière couche à la seconde couche



Les réseaux de neurones

Algorithme d'apprentissage

- Préalable : séparer ensemble des données d'apprentissage et ensemble des données de tests qu'on réserve pour l'évaluation.
- Initialiser les poids w_i aléatoirement
- Tant que “apprentissage non terminé”
  - Choisir un nouvel exemple (x_t, y_t) dans les données d'apprentissage;
  - Propagation avant : calculer la sortie du réseau avec les poids w_i dont on dispose;
  - Rétropropagation : Evaluer le gradient $\delta Loss_{wi} = dLoss(y_t, h_w(x_t))/dw_i$ pour tous les w_i du réseau en commencant par la dernière couche puis puis en propageant vers les couches antérieures;
  - Modifier les poids du réseau :
    - $w_i' = w_i - \alpha * \delta Loss_{wi}$,
      - α = taux d'apprentissage (hyper paramètre) qu'on fait décroître au cours du temps
  - Evaluer le réseau sur l'ensemble de tests.

Généralement on travaille par lot

Gradient moyen du lot

Les réseaux de neurones

Algorithme d'apprentissage

- On itère un très grand nombre de fois sur des millions de données
  - “de la répétition vient la perfection”...
  - L'apprentissage est généralement une opération longue, voire très longue
    - Ayant de gros besoins en puissance de calcul
    - ..
- Un fois que le réseau à appris, on peut l'utiliser sur une machine beaucoup moins puissante (PC, téléphone, informatique embarquée…) pour faire des prédictions.
- On peut récupérer un réseau “pré entrainé” et affiner son apprentissage..

Les réseaux de neurones

Petite démonstration

- Démonstration....
  - Langage Python
  - Tensorflow
    - Librairie open source d'apprentissage automatique développée par Google
    - Masque la complexité (descente de gradient, retropropagation, …)
  - Exemple diabetes (données labo américain)

	PatientID	Pregnancies	PlasmaGlucose	DiastolicBloodPressure	TricepsThickness	SerumInsulin	BMI	DiabetesPedigree	Age	Diabetic
0	1354778	0	171	80	34	23	43.509726	1.213191	21	0
1	1147438	8	92	93	47	36	21.240576	0.158365	23	0
2	1640031	7	115	47	52	35	41.511523	0.079019	23	0
3	1883350	9	103	78	25	304	29.582192	1.282870	43	1
4	1424119	1	85	59	27	35	42.604536	0.549542	22	0

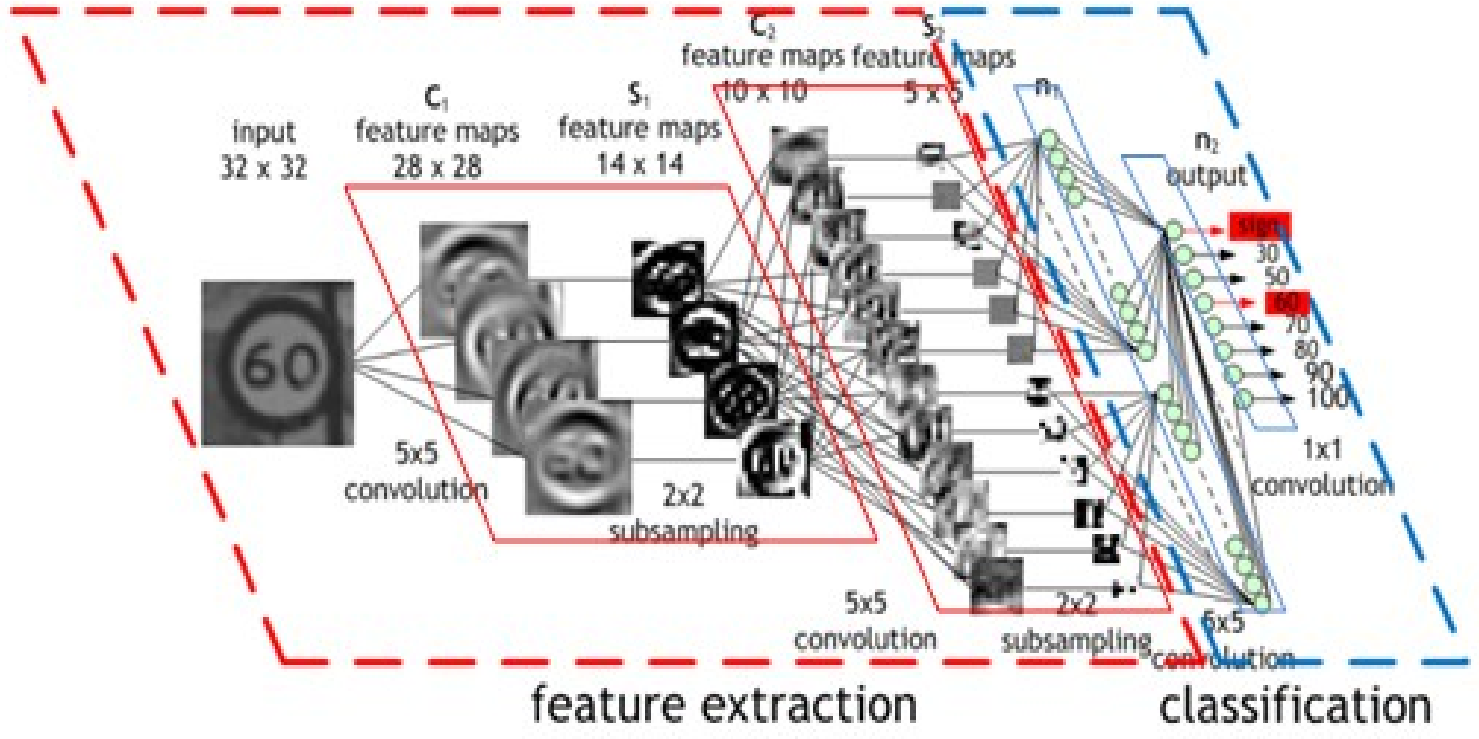
- Dataset : 15000 lignes
- 20% des données mise de coté pour l'évaluation
- 2 couches cachées de 20 neurones

Code : https://github.com/bouchetjl/deepLearning/tree/master/DiabetesLab1

Deep Learning

Quelques archectures de réseaux connues

- Les réseaux à convolution (CNN)

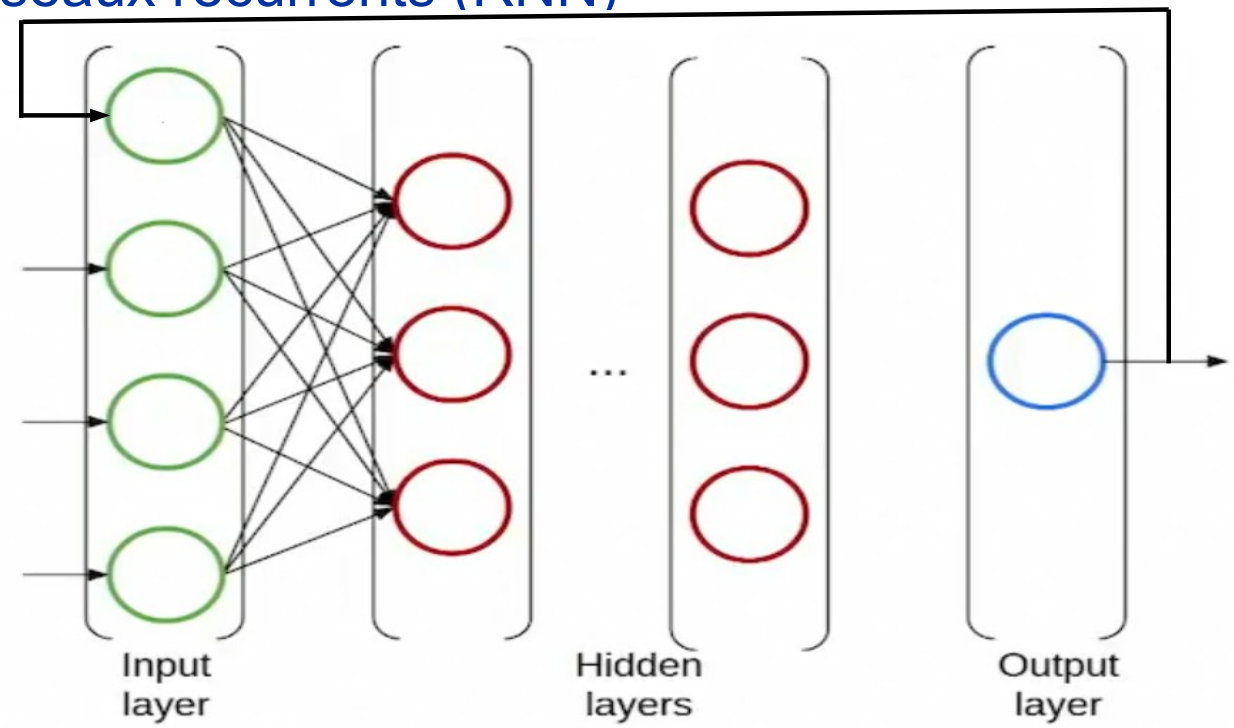


- Très utilisés dans le domaine de la reconnaissance d'images, vision par ordinateur.
- Très bons résultats

Deep Learning

Quelques archectures de réseaux connues

- Les réseaux récurrents (RNN)



- On ajoute une notion de mémoire court terme (réseaux LSTM)
- Très utilisés dans le domaine de la reconnaissance du langage naturel : dialogue homme machine, traduction automatique, ….
- Très bons résultats

Les réseaux de neurones

Applications industrielles (santé)

- Spécificités du domaine de la santé, accés aux données
  - Importance de la confidentialité
  - Données bruitées, très souvent incomplètes
- Applications privilégiées
  - Imagerie médicale : segmentation, comptage de cellules, d'organes,
  - Assistance en radiologie
  - Assistance aux médecins pour le diagnostic
  - Prévision de l'apparition d'une maladie ou d'un événement indésirable
- Nombreuses startups positionnées sur ce secteur.

Les réseaux de neurones

Applications industrielles (santé)

- Quelques exemples
  - Philips IntelliSpace Portal
    - Segmentation d'organes, détection de la tuberculose (radio thorax), ...
  - Dreamquark
    - application mobile ou web,
    - au moyen d'un dispositif de caméra rétinienne adaptable sur un smartphone
    - diagnostic précoce très performant de la rétinopathie diabétique et du glaucome, maladies en très fort développement en raison de l'explosion du diabète
  - Cardiologs
    - Reconnaissance des patterns dans les ECG
    - Pour détection rapide et précise de troubles cardiovasculaires
  - ...

Les réseaux de neurones

Applications industrielles (santé)

- Quelques exemples (suite)
  - Google AI
    - analyse des biopsies de tissus suspects, détection de cancer
    - Pour cette tâche, google annonce une exactitude à 89% contre 73% pour un pathologiste sans contrainte de temp
  - Implicity
    - Cardiologie connectée, filtrage “intelligent” des alertes en connaissance du dossier patient
  - Lumatia
    - Prévision des coûts et de dépenses de santé à partir de l'analyse de dossiers patient au format FHIR.

    ...