Plan du cours

1) Introduction au machine learning

2) Régularisation et forêts aléatoires

3) Réseau de neurones

4) Réseau de neurones convolutifs

Pour chaque séance:

1h de cours / support transparent

2h de travaux pratiques (amener un ordinateur portable)

Classification

- Pour l’étude de variable qualitative en sortie y.

  Exemples : on cherche à classifier les images en ‘Chien’, ‘Chat’ ou ‘canard’

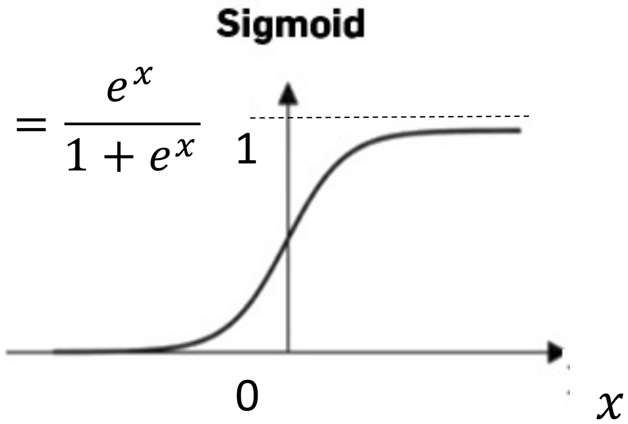
  on cherche à identifier la convection atmosphérique (avec 1 ou 0)

- L’étude de variable qualitative binaire peut se faire à l’aide d’une régression logistique.

$$f\left(\theta_0 + \theta_1 x_{1,i} + \cdots + \theta_m x_{m,i}\right) = \widehat{y_i}$$

avec $f(x) = \frac{e^x}{1+e^x}$

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



$\widehat{y_i}$ est interprété comme la probabilité p_i d’obtenir la variable qualitative correspondant à la valeur 1.

- La fonction loss est alors la negative cross-entropy :

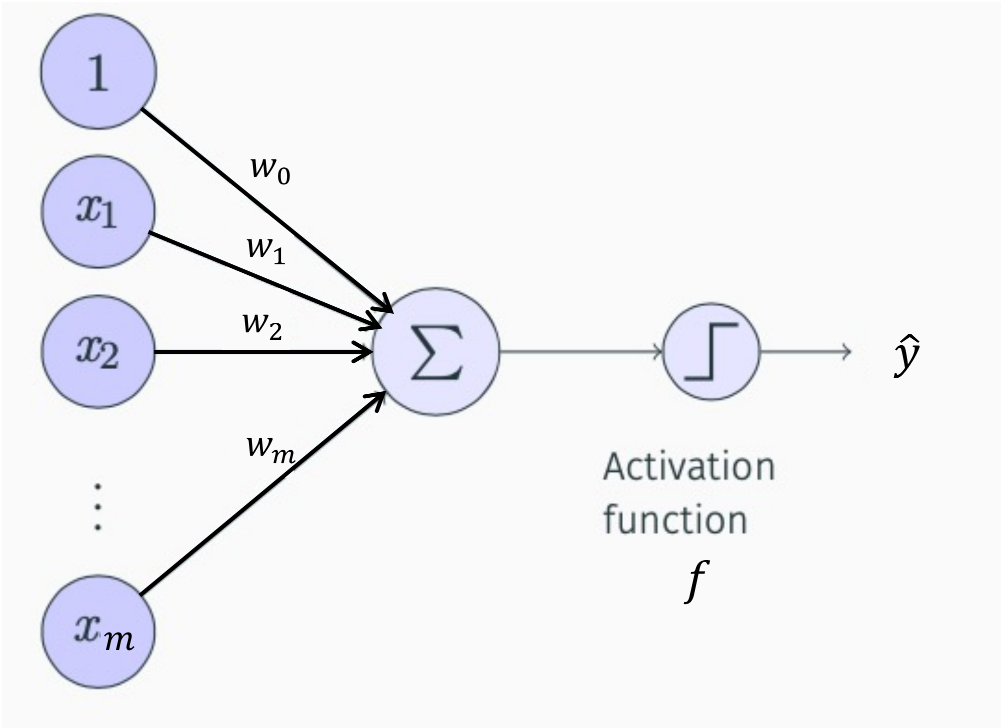
$$J(\boldsymbol{\theta}) = -(y_i \log \widehat{y_i} + (1 - y_i) \log(1 - \widehat{y_i}))$$

Le perceptron

Fct d'activation

Biais

$$\hat{y}_i = f(\boxed{w_O} + w_1 x_{1,i} + w_2 x_{2,i} + \cdots + w_m x_{m,i}) = \boxed{f}\left(w_O + \sum_{j=1}^{m} w_j x_{j,i}\right)$$



Poids / Weights

w_j

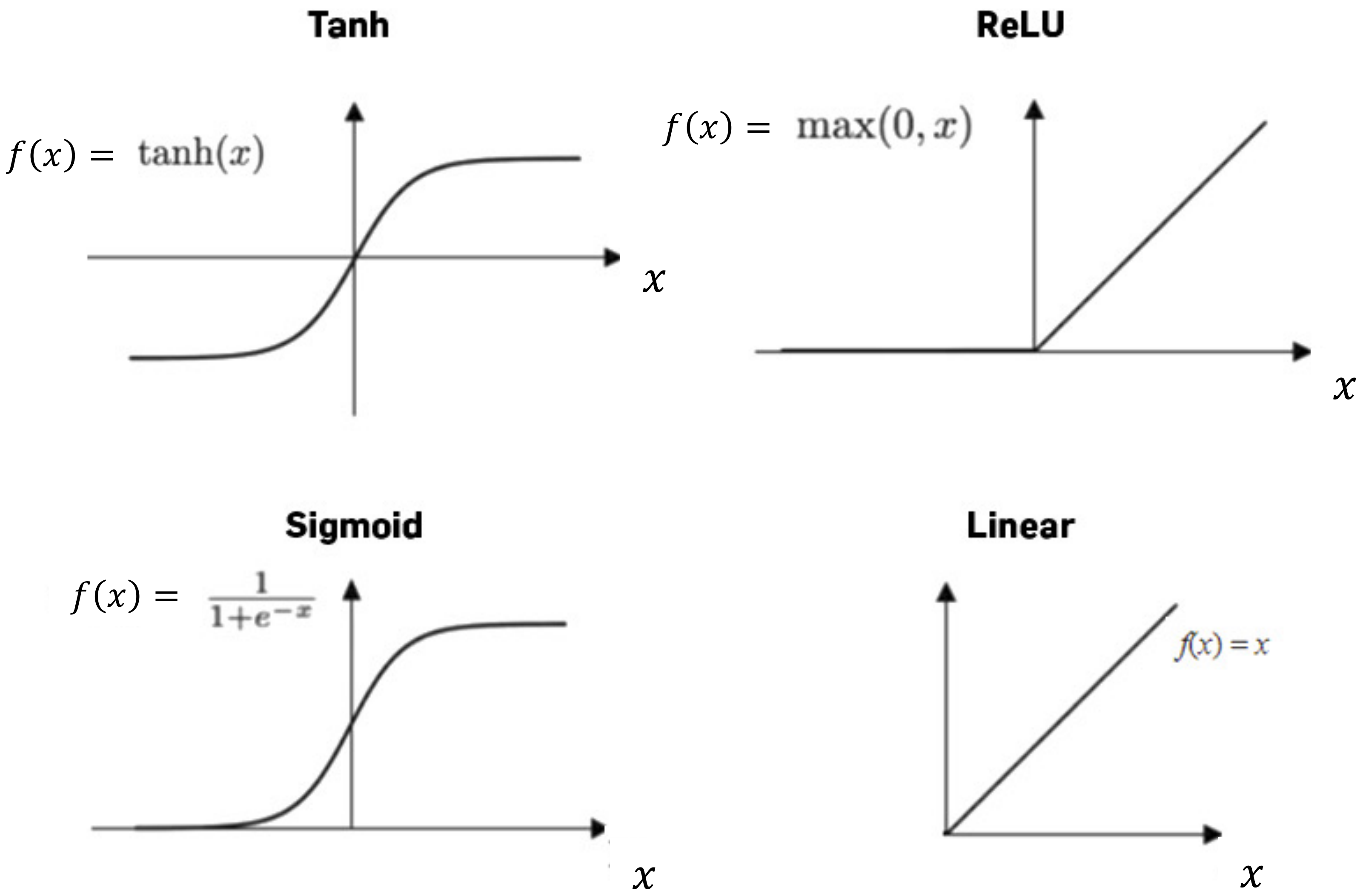
= paramètres du perceptron

$$f(x) = ?$$

Entrés / Input

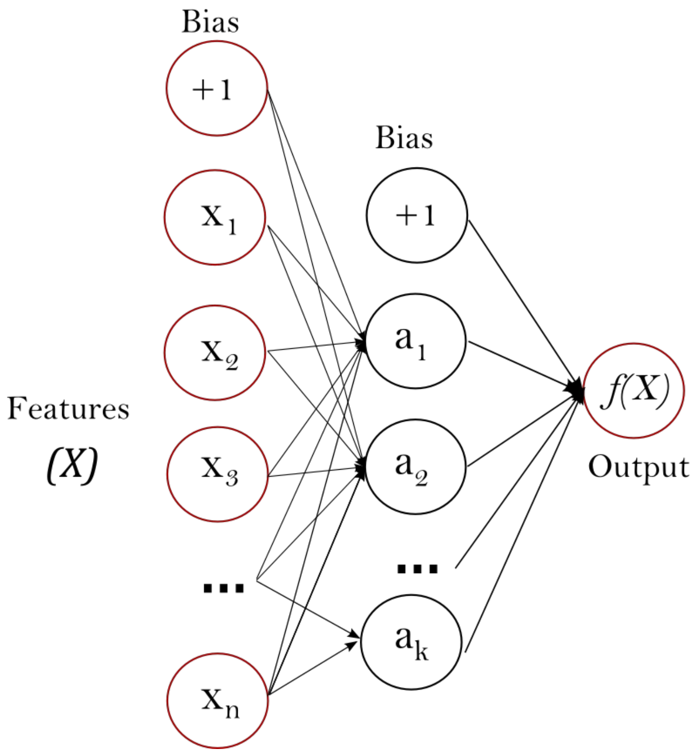
Sortie / Output

Quelques fonctions d’activation



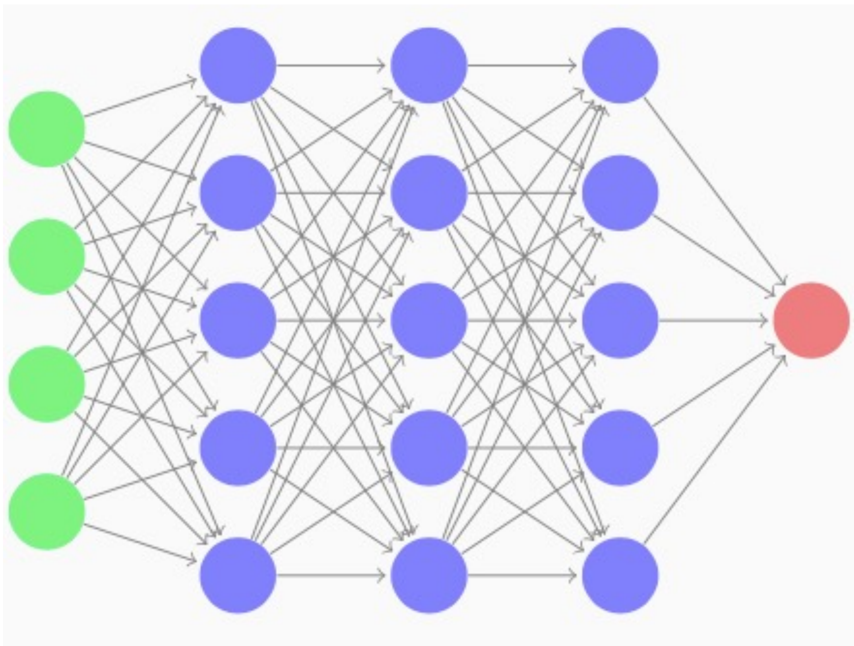
Perceptron et réseau de neurones

On peut combiner des perceptrons et former un réseau de neurones (=perceptron multi-couche).



Une couche caché avec k neurones

Rq : le biais n’est (en général) pas représenté



Trois couches cachées de 5 neurones

Classification et régression

Regression

- Dernière couche avec fonction d'activation : linéaire ou tanh
- Fonction Loss à optimiser :

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n} (\widehat{y}_i - y_i)^2$$

Classification

- Dernière couche utilisant la fonction softmax

$$p_j \text{ ou } \hat{y}_j = f_j(\boldsymbol{h}) = \frac{e^{h_j}}{\sum_k e^{h_k}}$$

- Fonction Loss à optimiser Entropie croisée :

$$J(\boldsymbol{\theta}) = -\sum_{i=1}^{n} \sum_{j=1}^{k} y_{i,j} \log(\widehat{y_{i,j}})$$

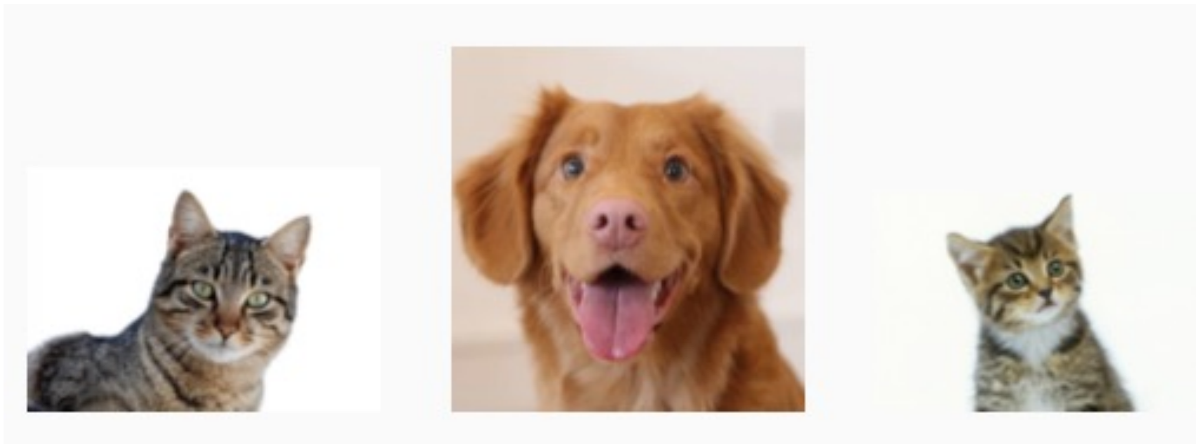
Exemple classification

On souhaite reconnaître des images de chien et de chat.

Features : X (codé par 3 matrices donnant les niveaux RBG pour chaque pixel).

$$m = \text{Nb Pixel} \times 3$$

Cible : y (codé 0 ou 1).



…

''Chat''	''Chien''	''Chat''	…
{1}	{0}	{1}	

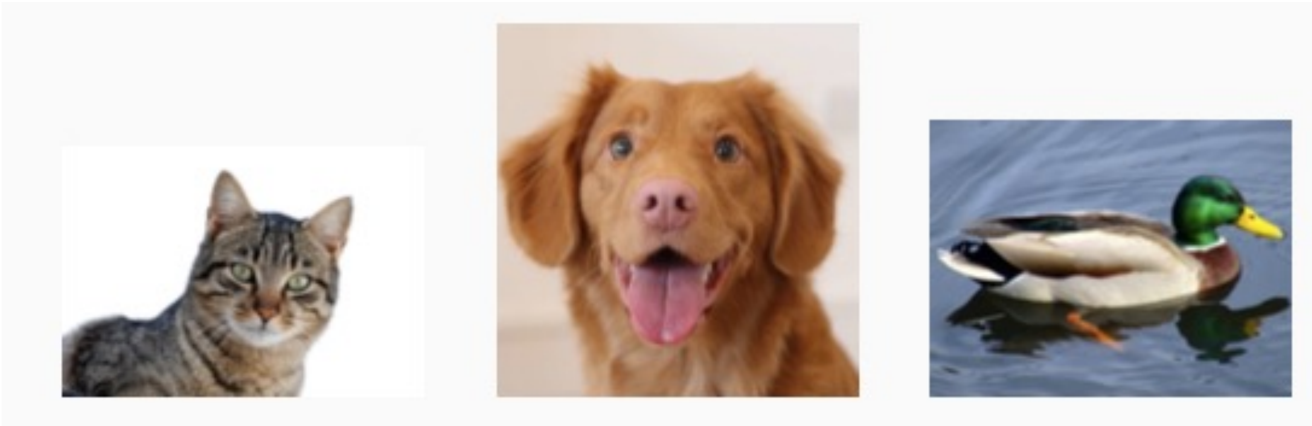
On entraîne un réseau qui estime $\hat{y}_i = f(X)$, qui s'interprète comme la probabilité que l'image soit un chat. Si $\hat{y}_i > 0.5$ alors on peut classer l'image comme celle d'un chat.

Exemple classification

On souhaite reconnaître des images de chien, de chat et de canard.

Features : X (codé par 3 matrices donnant les niveaux RBG pour chaque pixel).

$$m = \text{Nb Pixel} \times 3$$



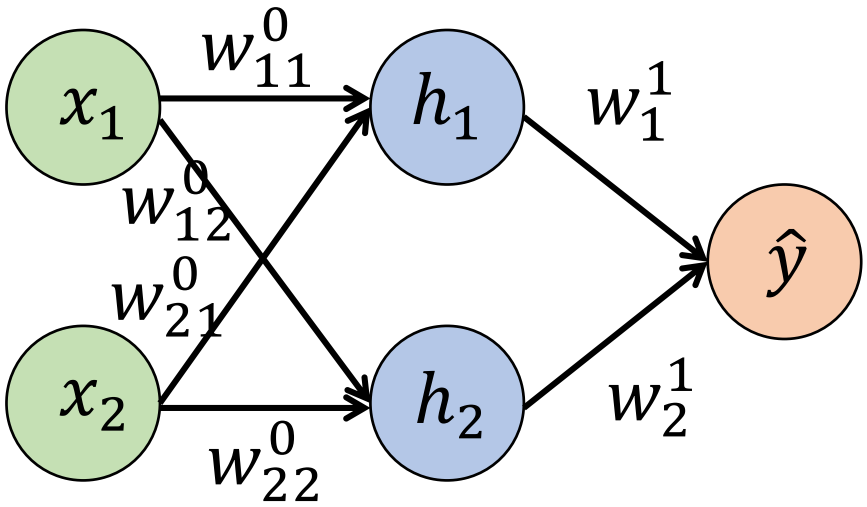
…

	''Chat''	''Chien''	''Canard''	…
Cible : y, un triplet d'entier de {0,1}	{1,0,0}	{0,1,0}	{0,0,1}	

On entraîne un réseau qui estime $\boldsymbol{\hat{y}_i} = \left(\hat{y}_{i,1}, \hat{y}_{i,2}, \hat{y}_{i,3}\right)$, un vecteur, dont la valeur est par exemple (0.1,0.7,0.3) les probabilités que l'image soit un chat, un chien ou un canard. La classe prédite est celle pour la quelle $\hat{y}_{i,j}$ est maximum.

Rétropropagation du gradient

Rétropropagation du gradient (*backward propagation*)= algorithme utilisé pour estimer le gradient de la fonction de coût pour les réseaux de neurone



1. On choisit un couple $(\boldsymbol{x}, y)$
2. On fait un calcule de $\hat{y}$ *forward* :

$h_j = f_0\left(\sum_{i=1}^{2} w_{ij}^0 x_i\right)$

$\hat{y} = f_1\left(\sum_{j=1}^{2} w_j^1 h_i\right)$

3. On calcule le gradient de la fonction loss :

$$\frac{\partial J}{\partial \hat{y}}$$

4. On rétropropage le gradient :

$$\frac{\partial J}{\partial w_j^1} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_j^1} \quad \text{et} \quad \frac{\partial J}{\partial h_j} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h_j}$$

$$\frac{\partial J}{\partial w_{ij}^0} = \frac{\partial J}{\partial h_j} \frac{\partial h_j}{\partial w_{ij}^0} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial h_j} \frac{\partial h_j}{\partial w_{ij}^0}$$

Travaux pratiques séance 3

- Explorer : https://playground.tensorflow.org/

- Deux grands types de banquise:

Banquise saisonnière (0-2m)

Banquise pluriannuelle (2 à 4m)

- Observation AMSR :

Radiomètre micro-onde aux fréquences 6.9, 10.65, 18.7, 23.8, 36.5 et 89.0GHz. Deux observations par jour à une résolution de 5 à 56 km