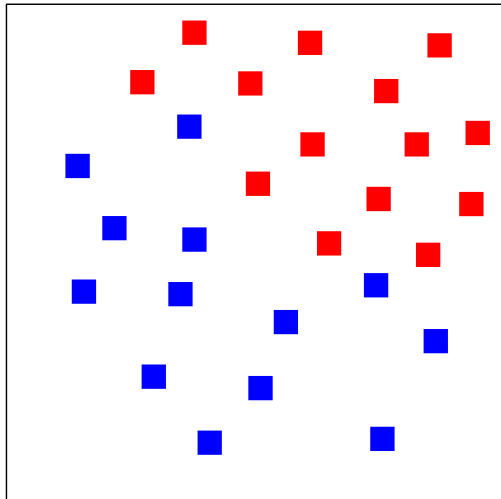


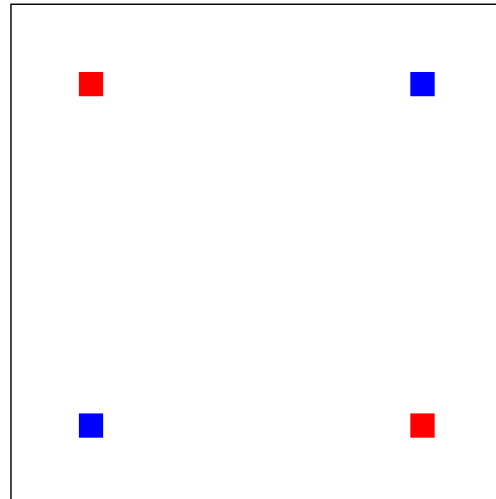
Données non linéairement séparables

Données non linéairement séparables

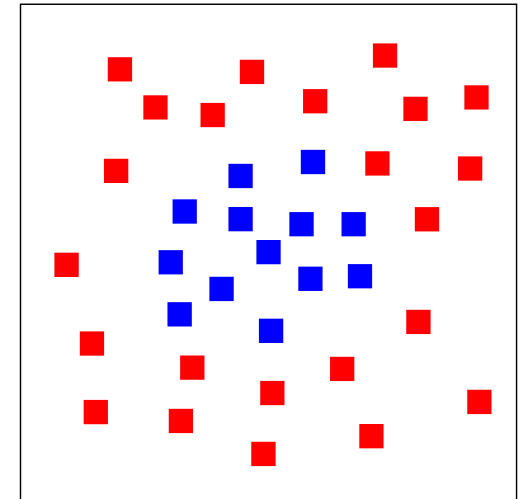
- Que faire dans ces situations ?



Soft (bruit)



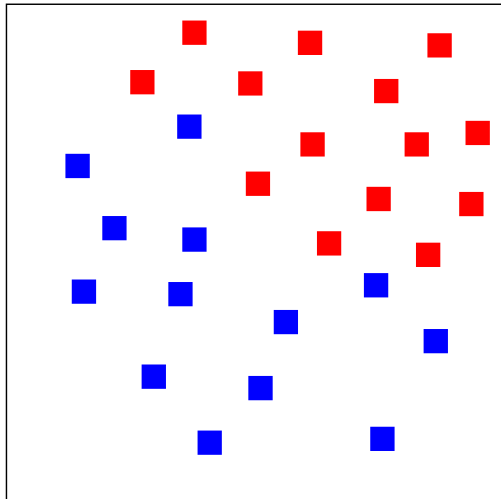
Hard (Intrinsèquement non linéaire)



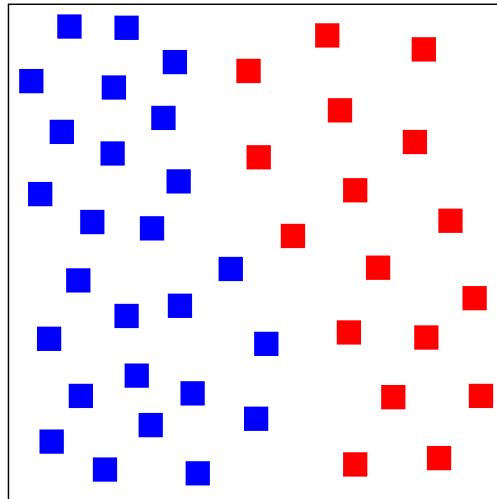
Intrinsèquement non linéaire réel

Données non linéairement séparables

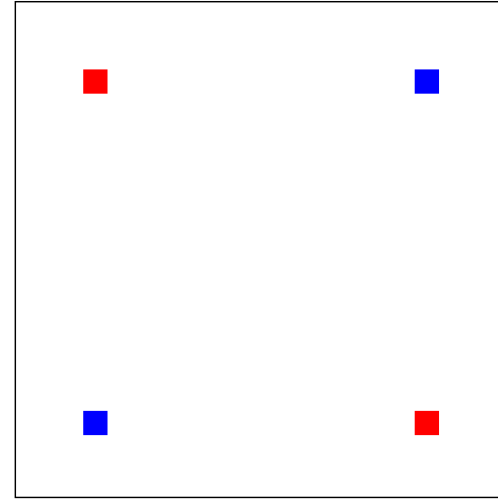
Que faire dans ces situations ?



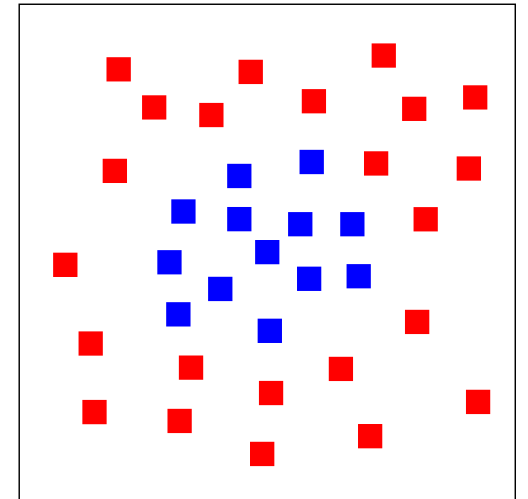
Soft (bruit)



Presque Linéaire (bruit ?)



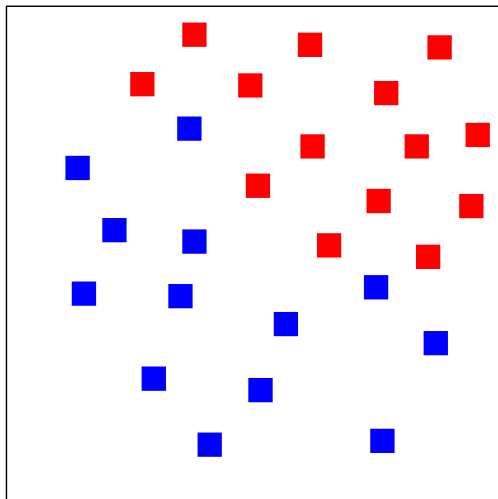
Hard (Intrinsèquement
non linéaire)



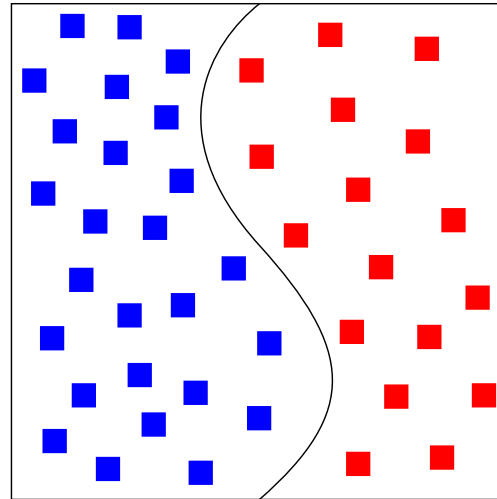
Intrinsèquement non linéaire réel

Données non linéairement séparables

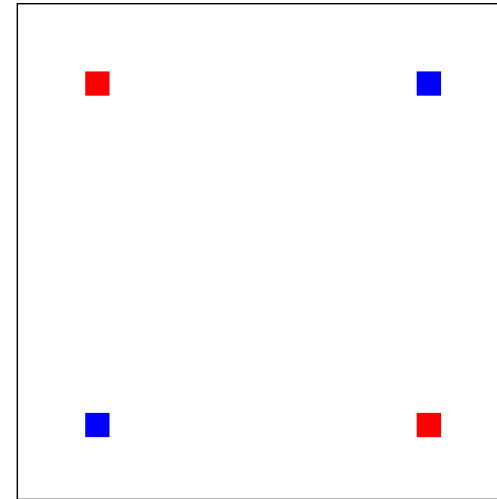
Que faire dans ces situations ?



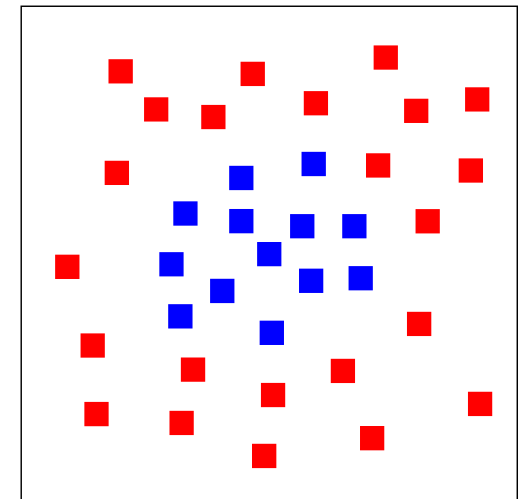
Soft (bruit)



Presque Linéaire (bruit ?)



Hard (Intrinsèquement
non linéaire)



Intrinsèquement non linéaire réel

Données non linéairement séparables

De quelle linéarité parle-t-on dans le cas du perceptron ?

Données non linéairement séparables

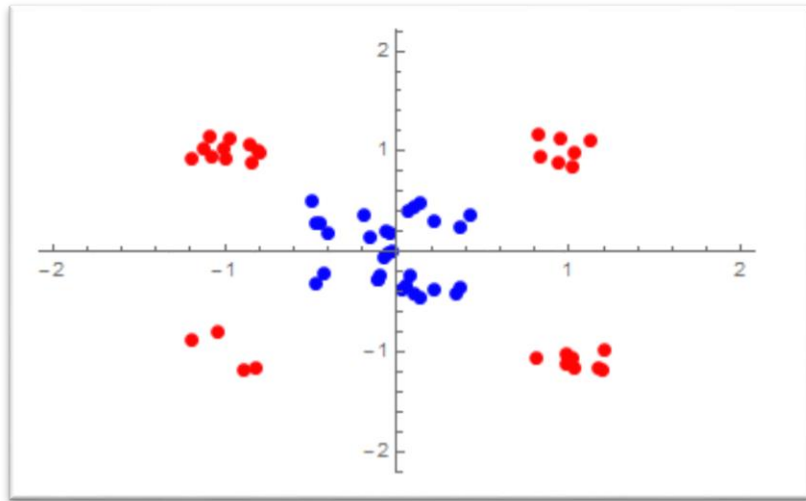
De quelle linéarité parle-t-on dans le cas du perceptron ?

Linéaire en fonction de W , pas de X !

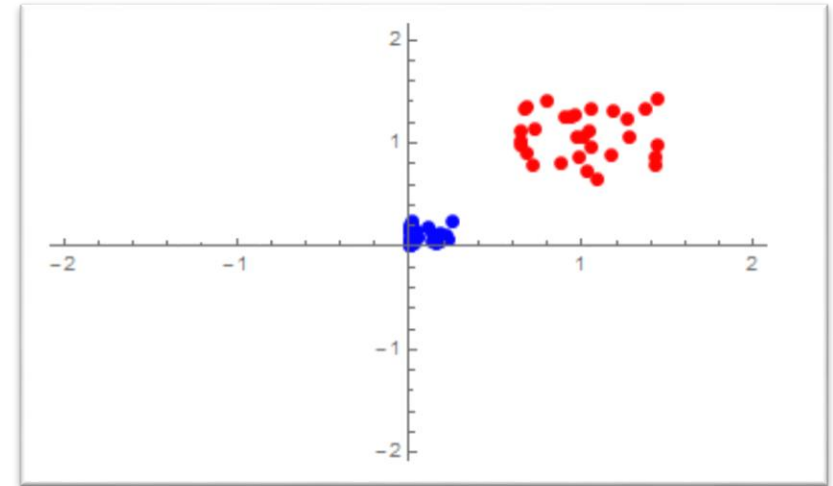
Transformation non linéaire des données d'entrée

Données non linéairement séparables

Transformation non linéaire sur les entrées...



$$X = (x, y, 1)$$

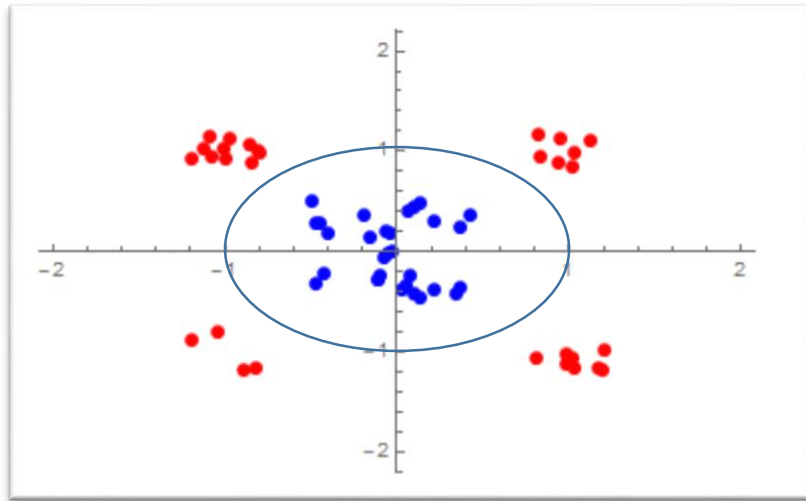


$$X = (x^2, y^2, 1)$$

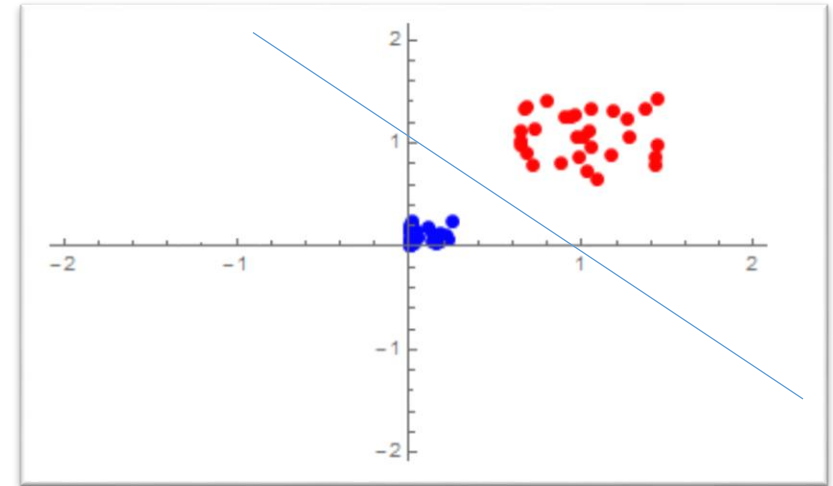
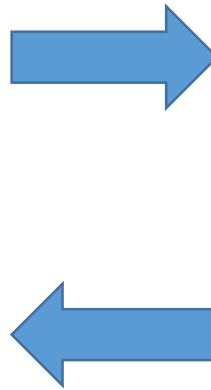
Données non linéairement séparables

Transformation non linéaire sur les entrées...

... et classement dans ce nouvel espace par un perceptron !



$$X = (x, y, 1)$$

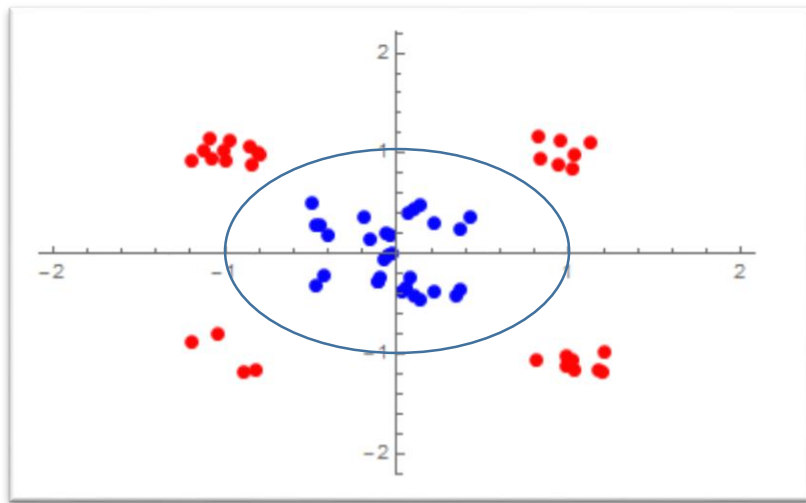


$$X = (x^2, y^2, 1)$$

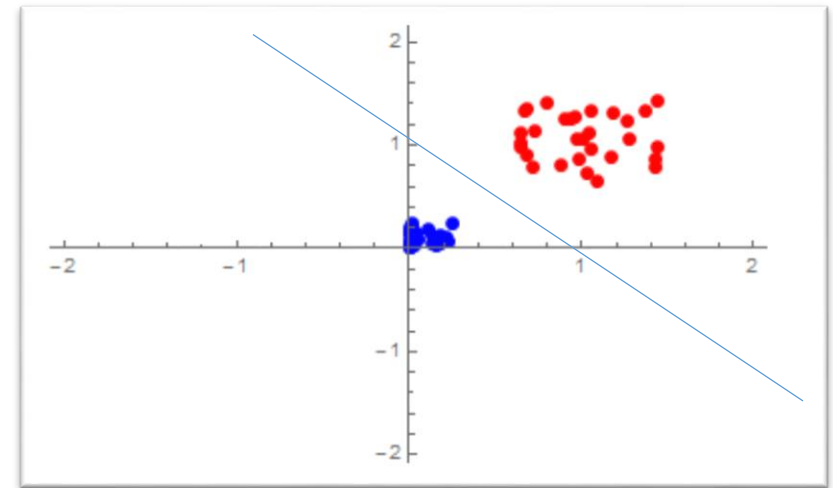
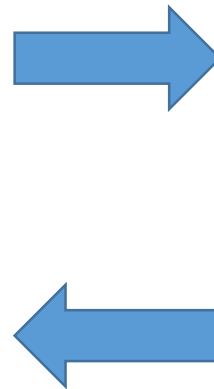
Données non linéairement séparables

Le perceptron semble avoir toujours le même nombre d'entrées

=> Capacité de généralisation inchangée ? 😊



$$X = (x, y, 1)$$



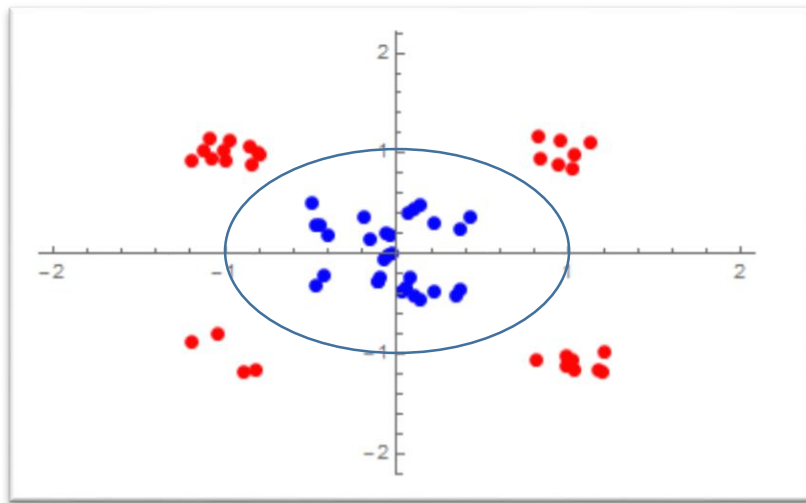
$$X = (x^2, y^2, 1)$$

Données non linéairement séparables

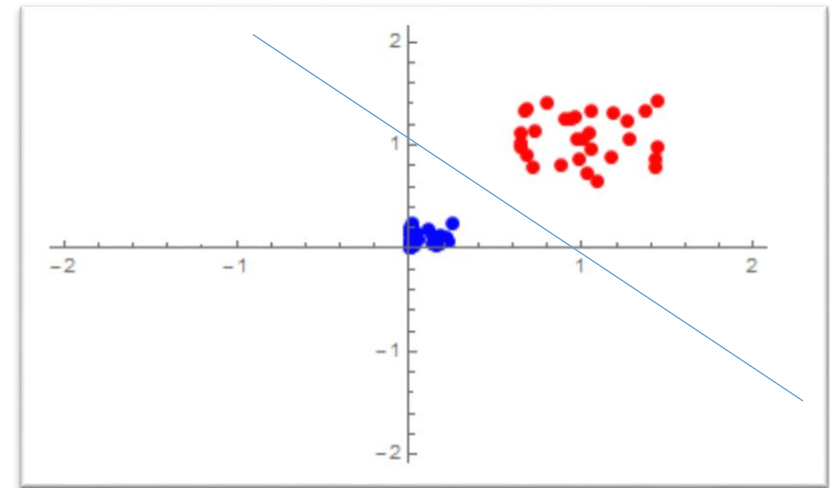
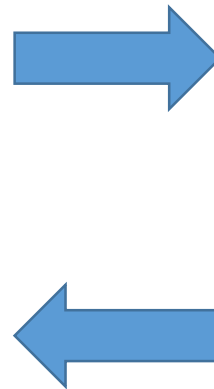


Le perceptron semble avoir toujours le même nombre d'entrées

=> Capacité de généralisation inchangée ? 😊



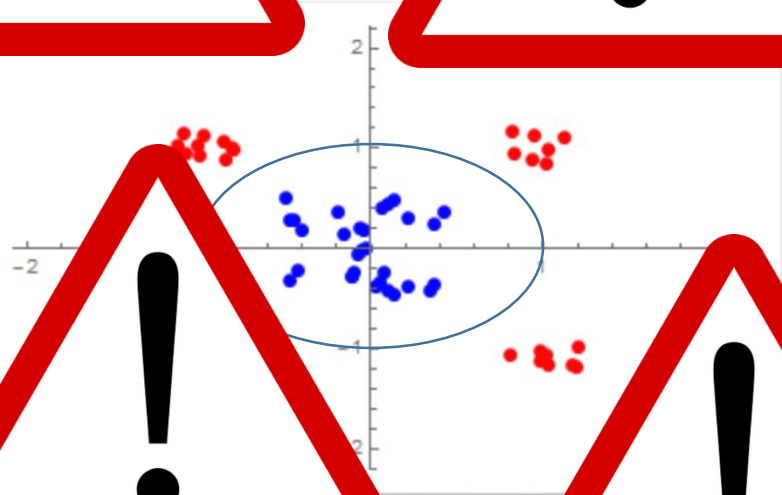
$$X = (x, y, 1)$$



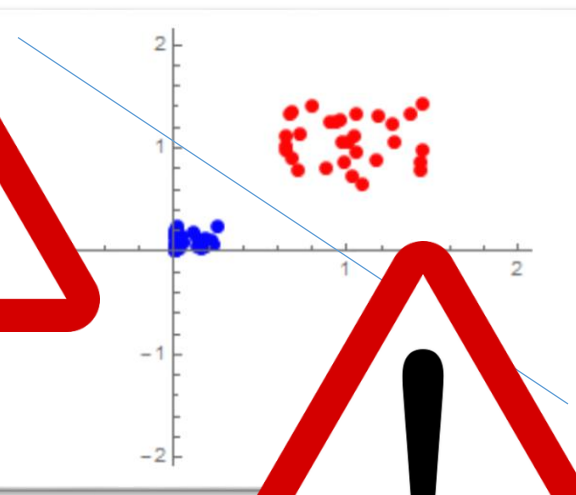
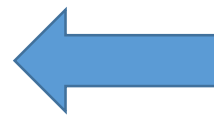
$$X = (x^2, y^2, 1)$$

Données non linéairement séparables

Un perceptron se voit-il toujours le même nombre d'itérations de mise à jour ? 😊



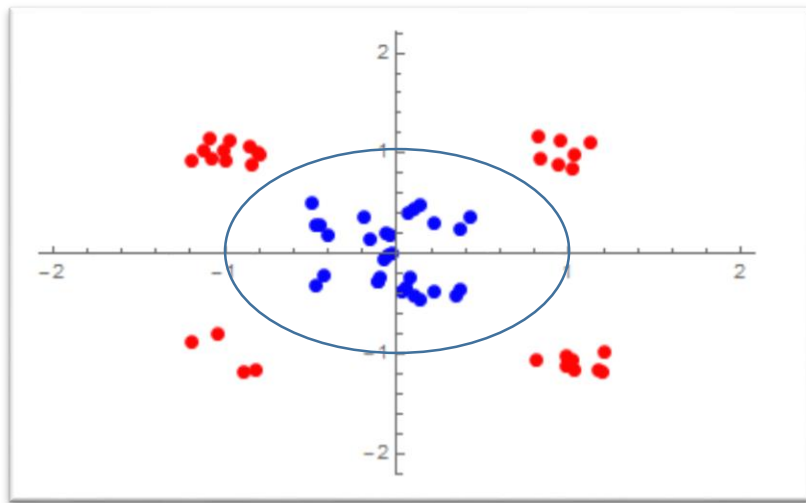
$$X = (x, y, 1)$$



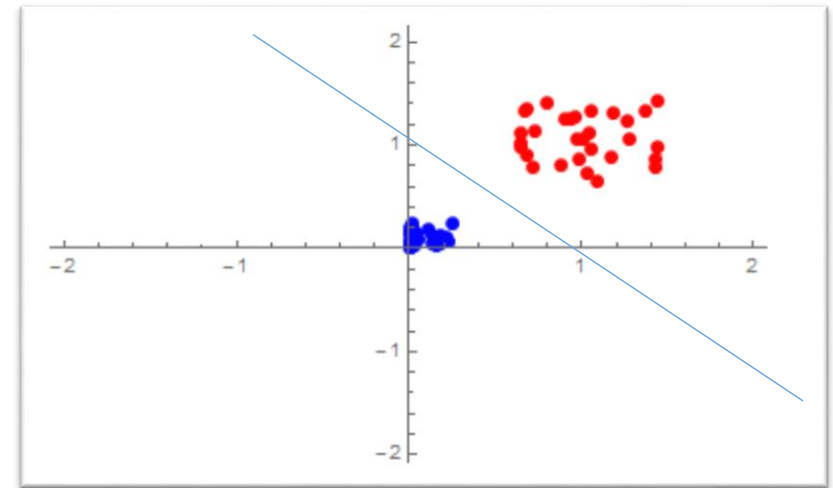
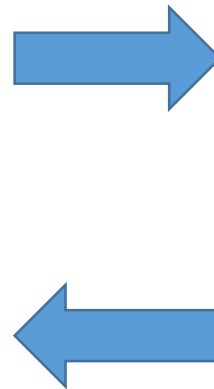
$$X = (x^2, y^2, 1)$$

Données non linéairement séparables

Nous avons choisi cette transformation en particulier ...



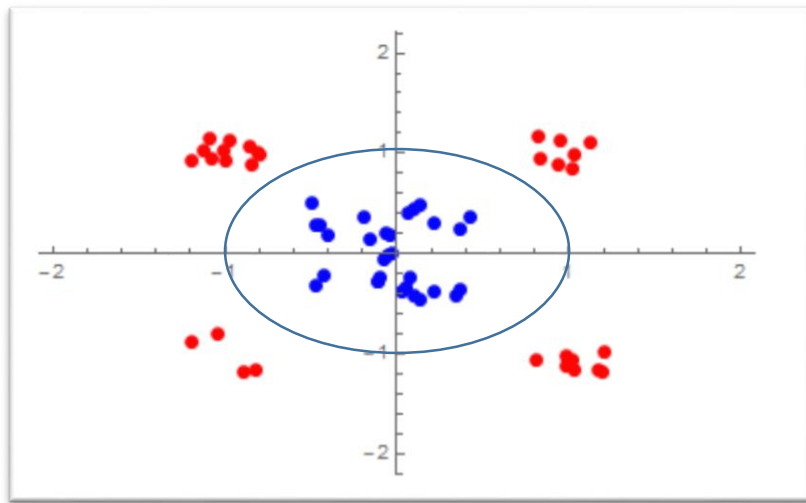
$$X = (x, y, 1)$$



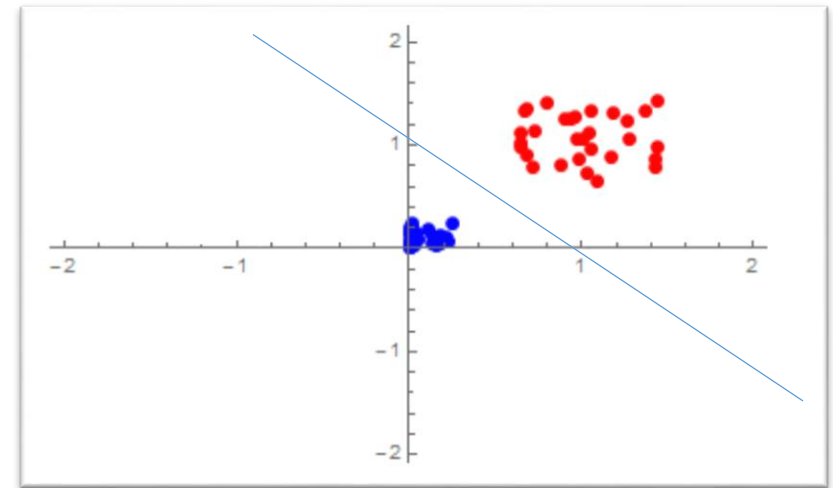
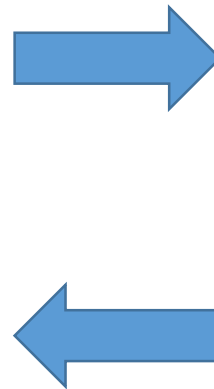
$$X = (x^2, y^2, 1)$$

Données non linéairement séparables

Nous avons choisi cette transformation en particulier ...
... car nous avons observé les données !!!



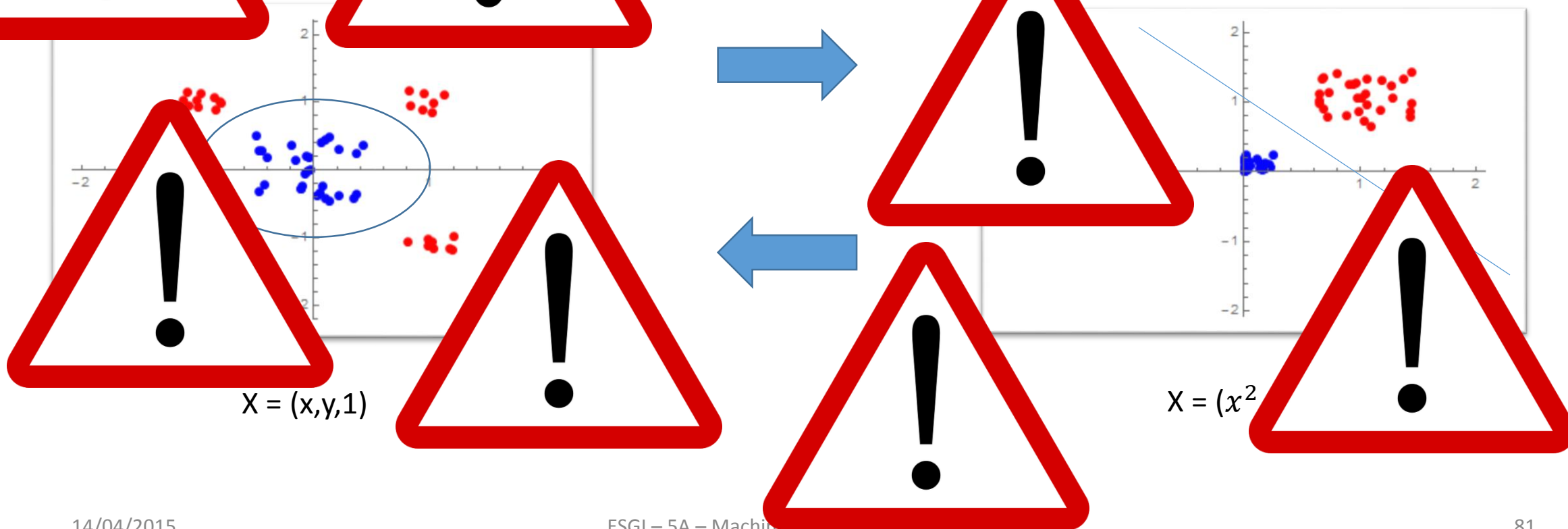
$$X = (x, y, 1)$$



$$X = (x^2, y^2, 1)$$

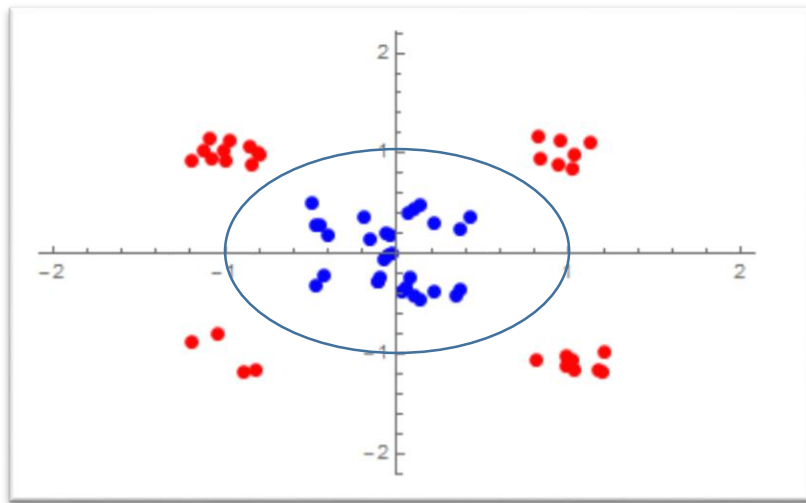
Données non linéairement séparables

avons choisi une transformation en particulier ...
car nous n'avons pas préservé les données !

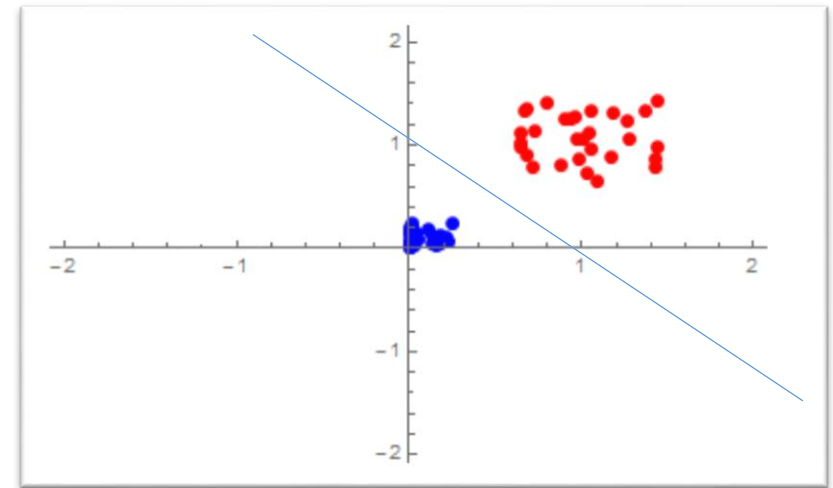
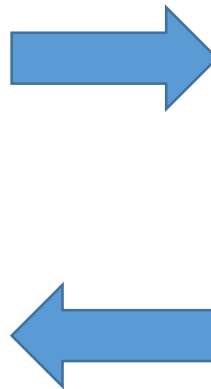


Données non linéairement séparables

Entrées réelles :



$$X = (1, x, y)$$

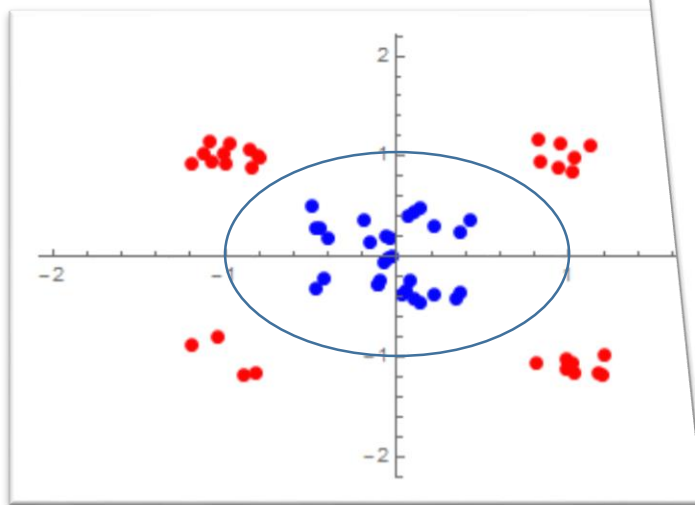


$$X = (1, x^2, y^2) \longleftrightarrow X = (1, x, y, xy, x^2, y^2)$$

Si on fixe $w_1 = w_2 = w_3 = 0$

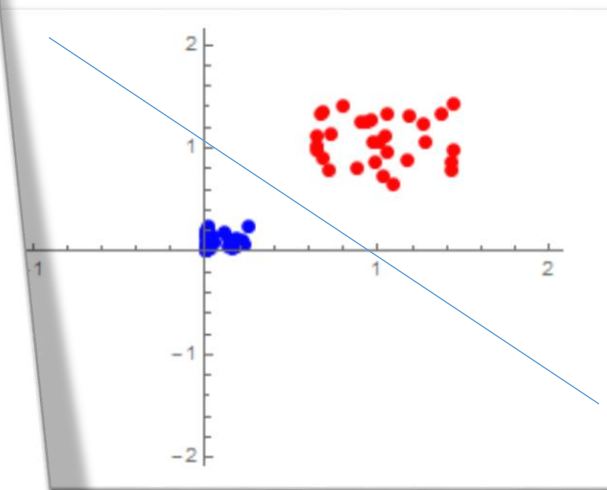
Données non linéairement séparables

Entrées réelles :



$$X = (1, x, y)$$

Real Algorithms
Don't require
Human intervention
to Learn !



$$X = (1, x^2, y^2) \longleftrightarrow X = (1, x, y, xy, x^2, y^2)$$

Si on fixe $w_1 = w_2 = w_3 = 0$

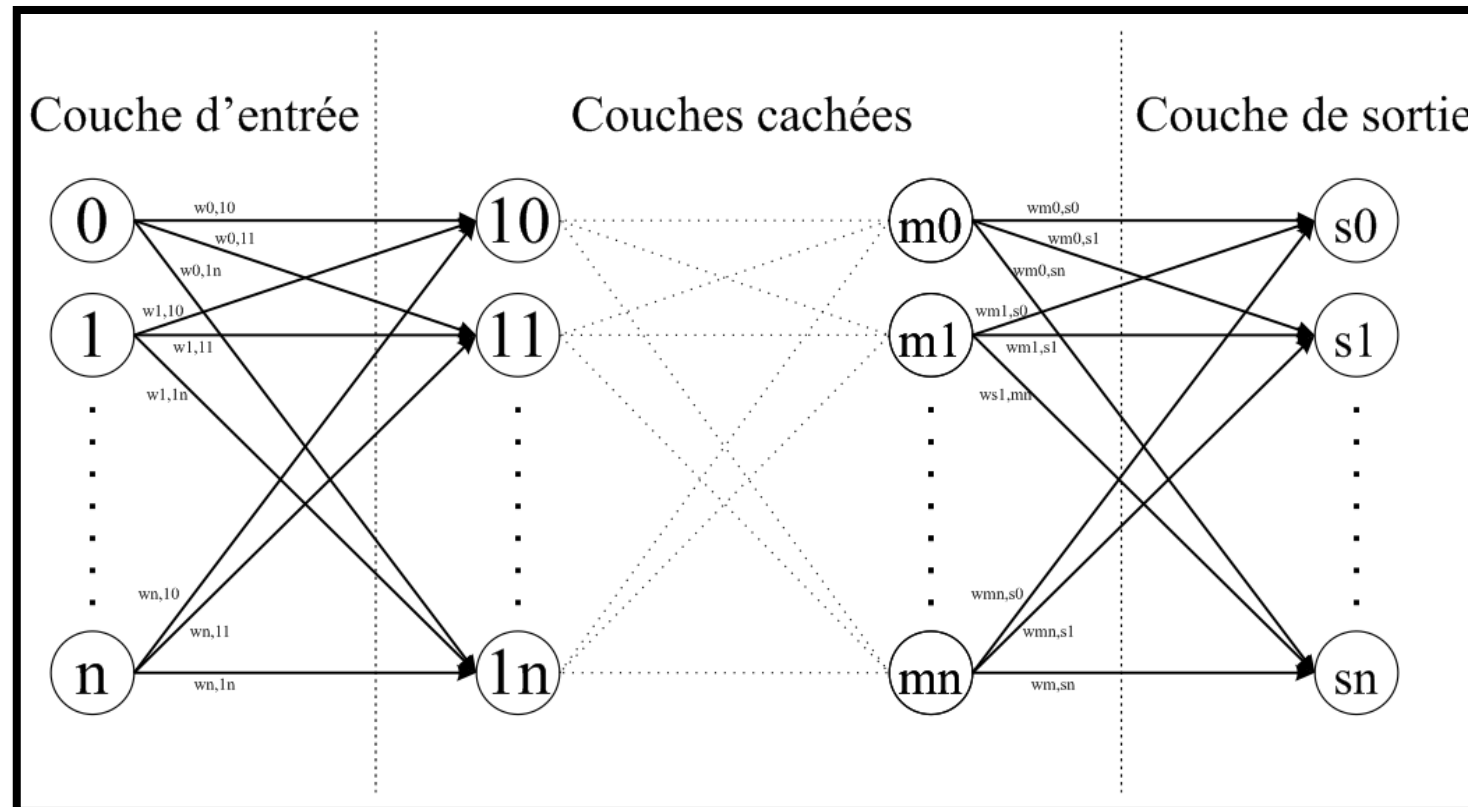
Données non linéairement séparables

Comment correctement estimer le prix de transformations non linéaires des entrées vis-à-vis de la généralisation?

Perceptron Multi Couches

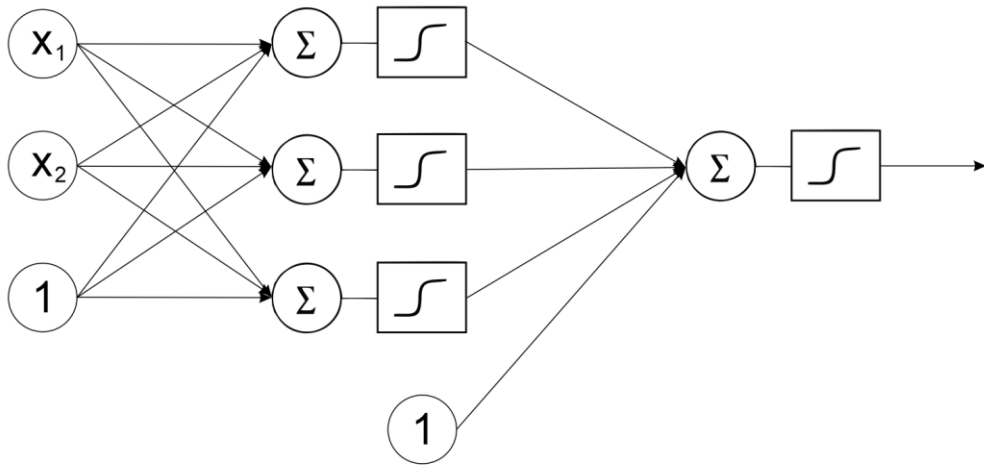
Principes

Intuition : mettre des perceptrons en série et en parallèle ...

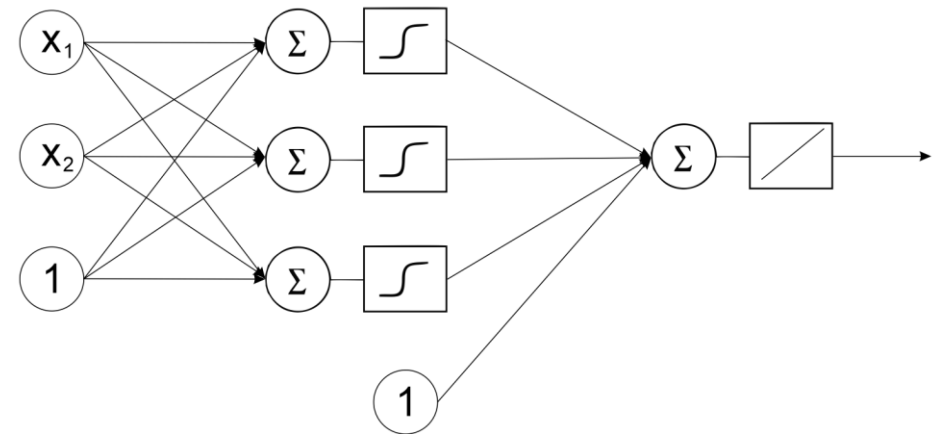


Principes

Intuition : mettre des perceptrons en série et en parallèle ...



Perceptron multi couches pour la classification

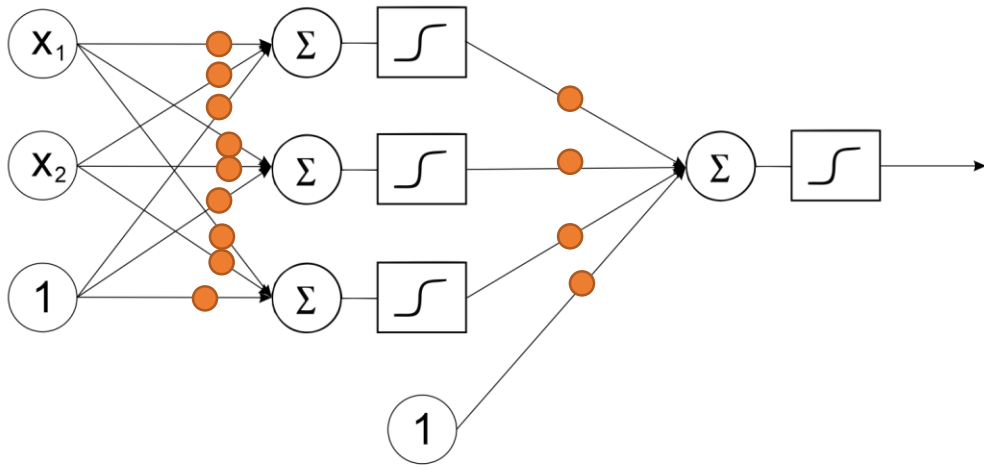


Perceptron multi couches pour la régression

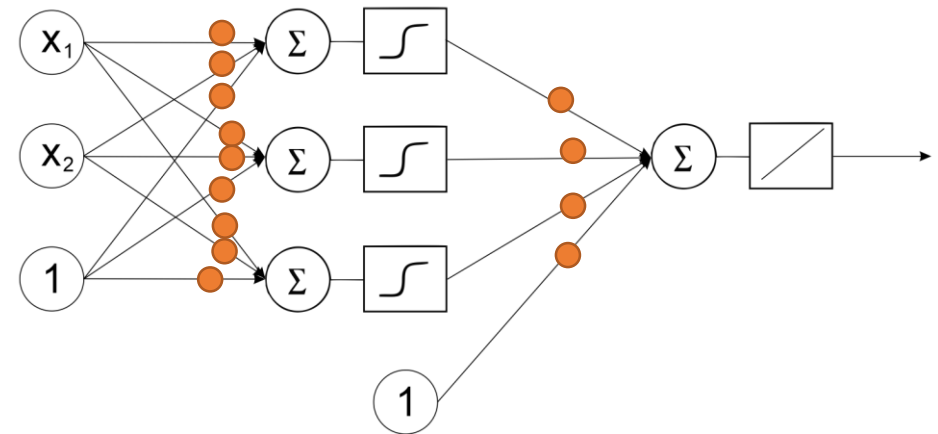
Principes

Intuition : mettre des perceptrons en série et en parallèle ...

Paramètres : ensemble des poids



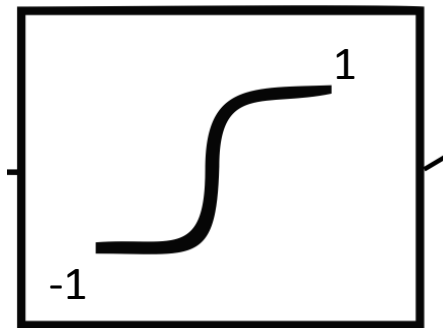
Perceptron multi couches pour la classification



Perceptron multi couches pour la régression

Principes

Nous n'utilisons pas le signe de la somme des entrées pour chaque perceptron, mais une fonction dite « d'activation » sigmoïde.



$Tanh(x)$

Principes

Soit w_{ij}^l le poids de la couche l liant le neurone i de la couche $l - 1$ au neurone j de la couche l .

Soit s_j^l le signal (somme pondérée des entrées) du neurone j de la couche l .

Soit θ la fonction sigmoïde appliquée au signal de chaque neurone intermédiaire (on utilisera $Tanh$).

Soit x_j^l la valeur de sortie effective d'un neurone.

Soit d^l le nombre de neurones appartenant à la couche l (sans compter le neurone de biais)

Principes

Soit x_j^l la valeur de sortie effective du neurone j de la couche l .

Règle récursive de calcul des X:

$$x_j^l = \theta(s_j^l) = \text{Tanh}\left(\sum_{i=0}^{d^{l-1}} w_{ij}^l x_i^{l-1}\right)$$

Principes

Comment trouver les w_{ij}^l minimisant l'erreur de classification sur la base d'exemples?

Rétropropagation du gradient

Pour la classification, répéter :

- Prendre un exemple étiqueté au hasard : $\begin{bmatrix} x_1^0 \\ \vdots \\ x_{d^0}^0 \end{bmatrix} \rightarrow \begin{bmatrix} y_0 \\ \vdots \\ y_{d^L} \end{bmatrix}$

- Pour tous les neurones j de la dernière couche L calculer :

$$\delta_j^L = (1 - (x_j^L)^2) \times (x_j^L - y_j)$$

- En déduire pour tous les autres neurones de l'avant dernière couche à la première :

$$\delta_i^{l-1} = (1 - (x_i^{l-1})^2) \times \sum_{j=1}^{d^l} (w_{ij}^l \times \delta_j^l)$$

- Puis mettre à jour tous les w_{ij}^l :

$$w_{ij}^l \leftarrow w_{ij}^l - \alpha x_i^{l-1} \delta_j^l$$

Rétropropagation du gradient

Pour la régression, répéter :

- Prendre un exemple étiqueté au hasard : $\begin{bmatrix} x_1^0 \\ \vdots \\ x_{d^0}^0 \end{bmatrix} \rightarrow \begin{bmatrix} y_0 \\ \vdots \\ y_{d^L} \end{bmatrix}$

- Pour tous les neurones j de la dernière couche L calculer :

$$\delta_j^L = (x_j^L - y_j)$$

- En déduire pour tous les autres neurones de l'avant dernière couche à la première :

$$\delta_i^{l-1} = (1 - (x_i^{l-1})^2) \times \sum_{j=1}^{d^l} (w_{ij}^l \times \delta_j^l)$$

- Puis mettre à jour tous les w_{ij}^l :

$$w_{ij}^l \leftarrow w_{ij}^l - \alpha x_i^{l-1} \delta_j^l$$