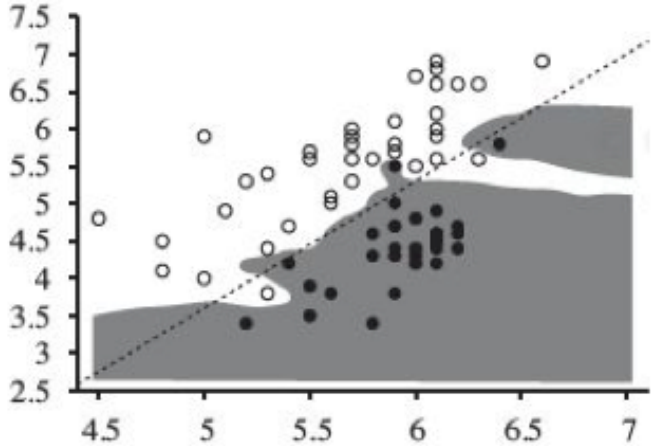
Limitation des classifieurs linéaires

- Si les données d'entraînement sont séparables linéairement, le perceptron et la régression logistique vont trouver cette séparation



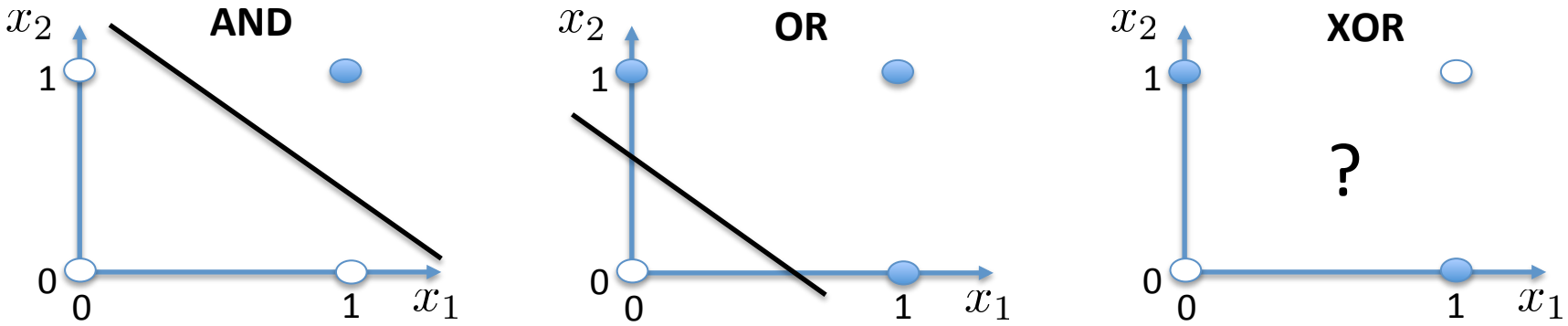
= prédiction classe • par *k* plus proche voisins

- *k* plus proche voisins est non-linéaire, mais coûteux en mémoire et temps de calcul (pas approprié pour des problèmes avec beaucoup de données)

Limitation des classifieurs linéaires

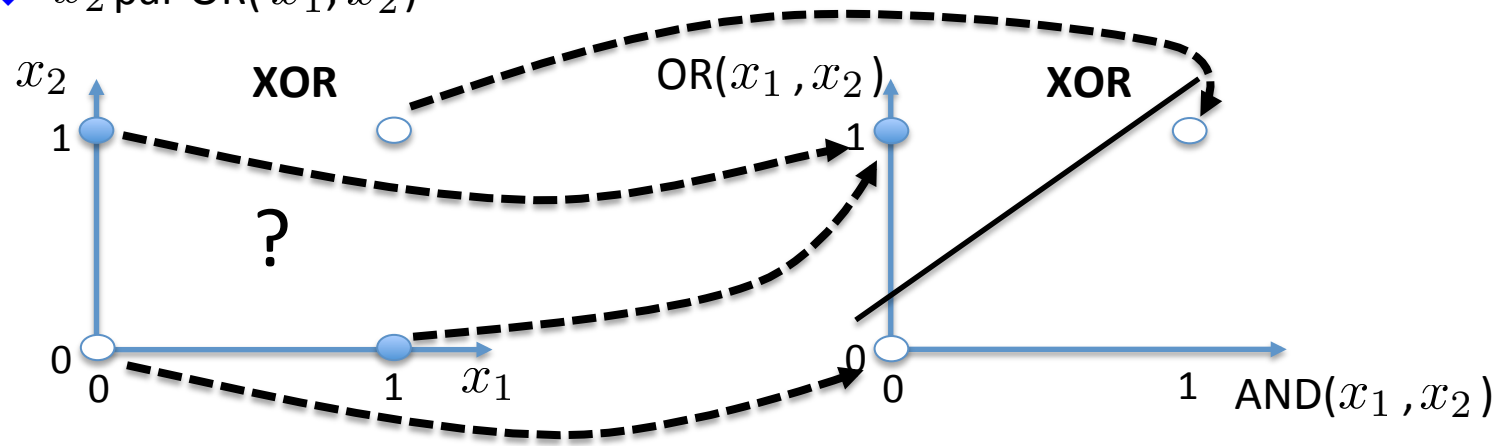
- Cependant, la majorité des problèmes de classification ne sont pas linéaires

- En fait, un classifieur linéaire ne peut même pas apprendre XOR!



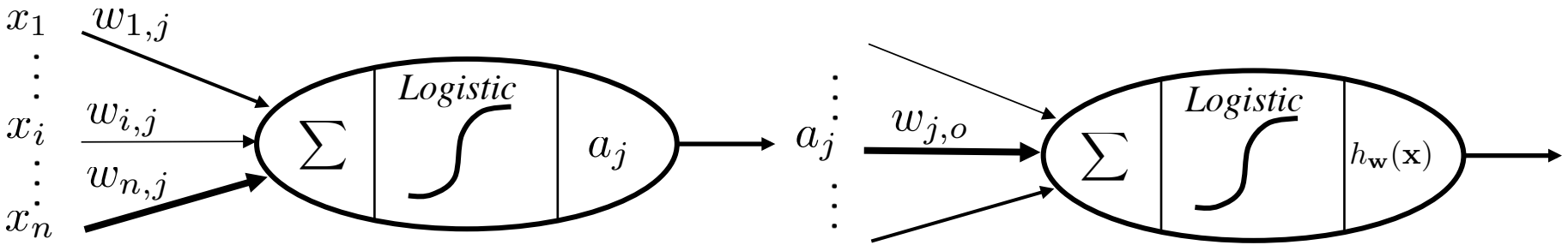
Limitation des classifieurs linéaires

- Par contre, on pourrait transformer l'entrée de façon à rendre le problème linéairement séparable sous cette nouvelle représentation
- Dans le cas de XOR, on pourrait remplacer
  - x_1 par AND(x_1, x_2) et
  - x_2 par OR(x_1, x_2)



Quatrième algorithme: réseau de neurones artificiel

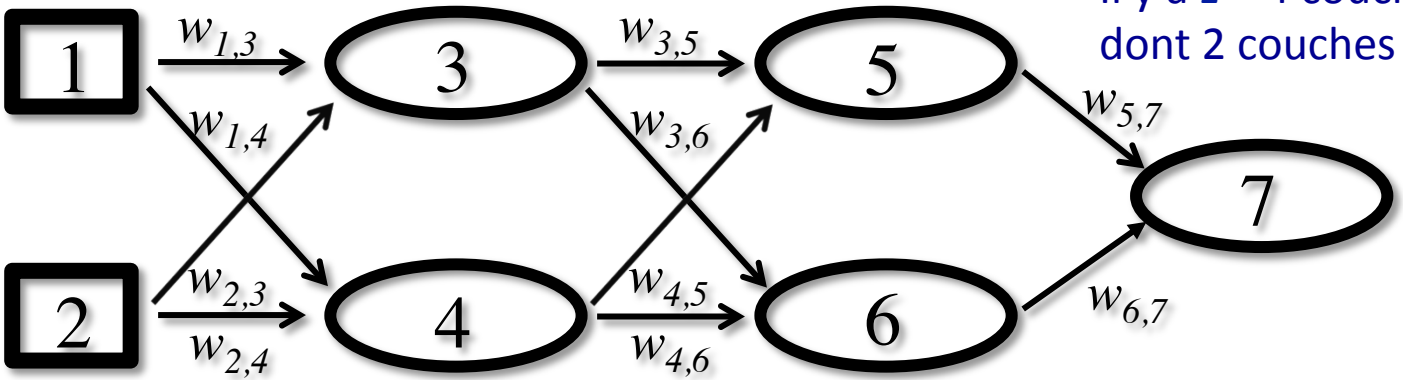
- **Idée**: apprendre les poids du classifieur linéaire **et** une transformation qui va rendre le problème linéairement séparable



Réseau de neurones à une seule couche cachée

Cas général à *L* couches

- Rien n'empêche d'avoir plus d'une couche cachée



Il y a *L* = 4 couches
dont 2 couches cachées

- On note a_j l'activité du *j*e « neurone », incluant les neurones d'entrée et de sortie. Donc on aura $a_i = x_i$
- On note in_j l'activité du *j*e neurone avant la non-linéarité logistique, c'est à dire

$$a_j = Logistic(in_j) = Logistic(\Sigma_i \; w_{i,j} a_i)$$