Leçon 6: Réseaux et apprentissage II

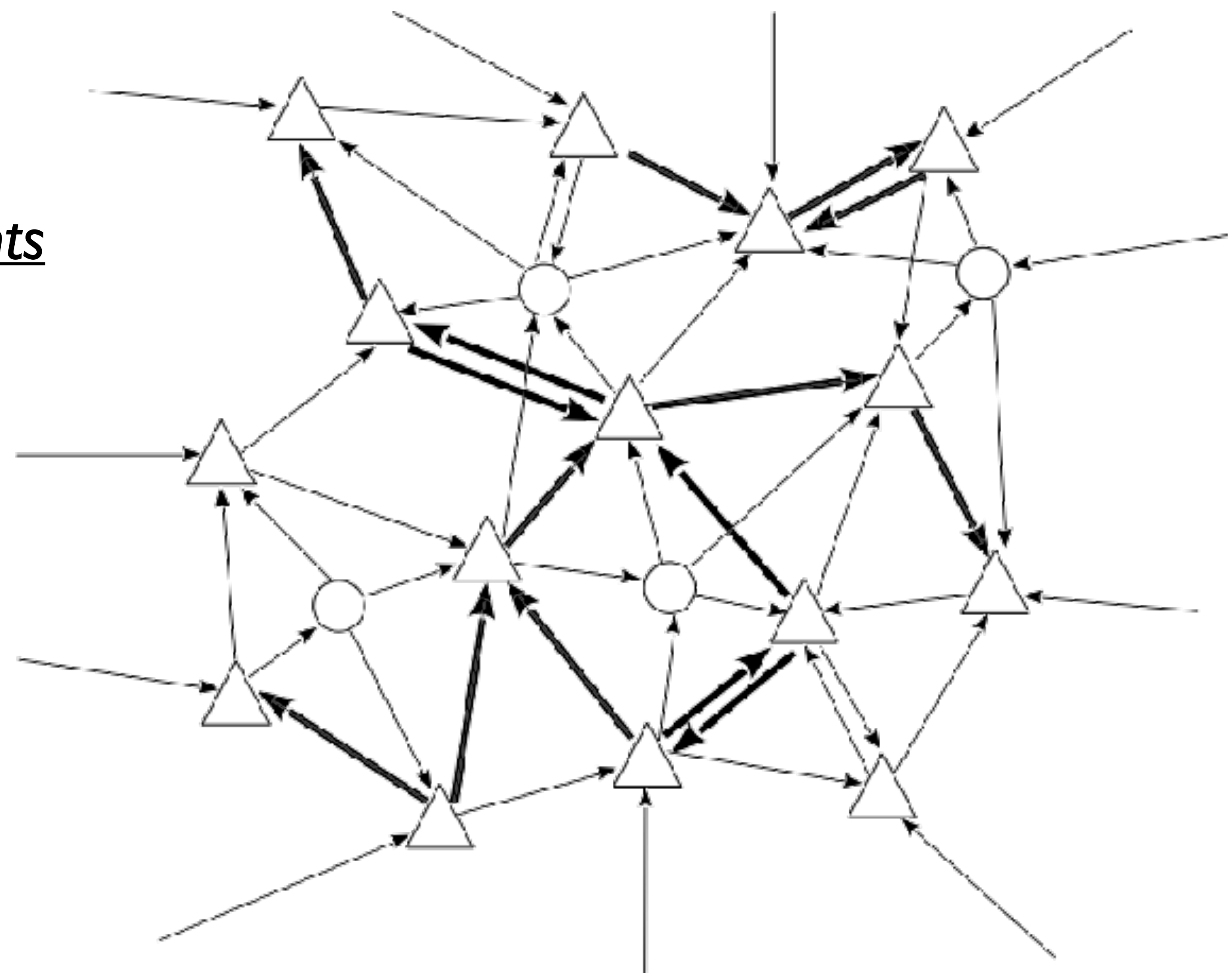
Jonas Ranft
Neurosciences computationnelles

Nov–Dec 2022

Mécanismes d’apprentissage

Les stimuli externes modifient l’activité neuronale dans le cerveau.

- Si un stimulus ne laisse pas de trace
  ⟶ aucun souvenir !
- Les changements d’activité engendrent des *changements de connectivité* (plasticité structurelle/synaptique) !
- Un autre stimulus peut modifier l’activité dans un sous-ensemble différent.
- La connectivité synaptique résulte de la superposition des traces laissées par les entrées externes.



Le modèle de Hopfield

Modèle pionnier d'un réseau de neurone avec attracteur

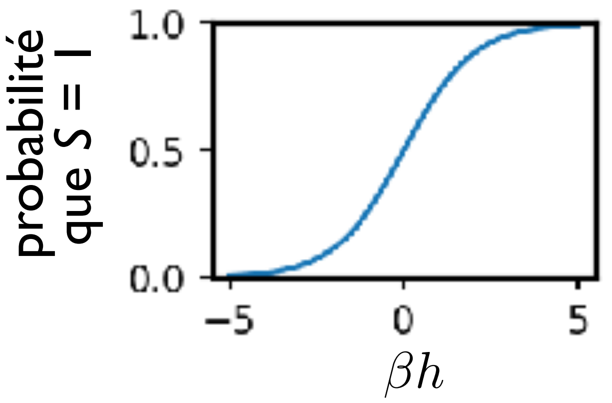
- Chaque neurone *i* est représenté par une variable *binaire*, $S_i \in \{-1, 1\}$, qui représente s'il est actif ou inactif dans un interval de temps Δt .

Le modèle de Hopfield

Modèle pionnier d'un réseau de neurone avec attracteur

- Chaque neurone *i* est représenté par une variable *binaire*, $S_i \in \{-1, 1\}$, qui représente s'il est actif ou inactif dans un interval de temps Δt .
- L'activité à $t + \Delta t$ dépend de l'état au temps précédent t selon

$$\mathrm{prob}[S_i(t+\Delta t) = 1] = \frac{1}{1+e^{-\beta h_i(t)}}$$



avec les entrées $h_i = \sum_{j=1 \ldots N} w_{ij} S_j$.

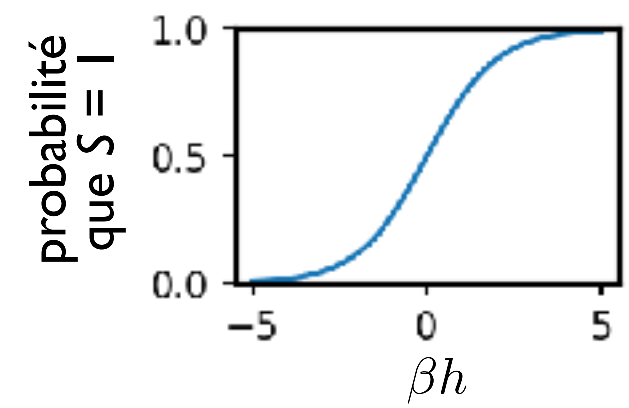
matrice de connexion

Le modèle de Hopfield

Modèle pionnier d'un réseau de neurone avec attracteur

- Chaque neurone *i* est représenté par une variable *binaire*, $S_i \in \{-1, 1\}$, qui représente s'il est actif ou inactif dans un interval de temps Δt .
- L'activité à $t + \Delta t$ dépend de l'état au temps précédent t selon

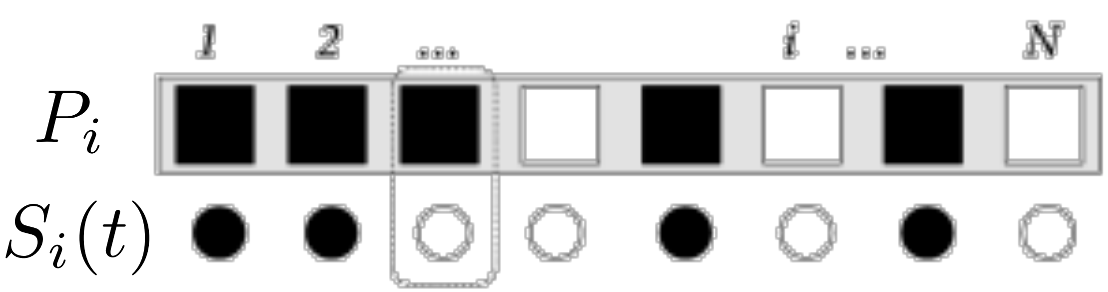
$$\mathrm{prob}[S_i(t+\Delta t) = 1] = \frac{1}{1 + e^{-\beta h_i(t)}}$$



avec les entrées $h_i = \sum_{j=1 \dots N} w_{ij} S_j$.

matrice de connexion

- La mémoire à retenir sont les motifs $P_i^\mu \in \{-1, 1\}$: $\mu = 1 \dots K$



motif (immuable)

état (peut évoluer)

<u>"*overlap*" (dégré d'accord) :</u>

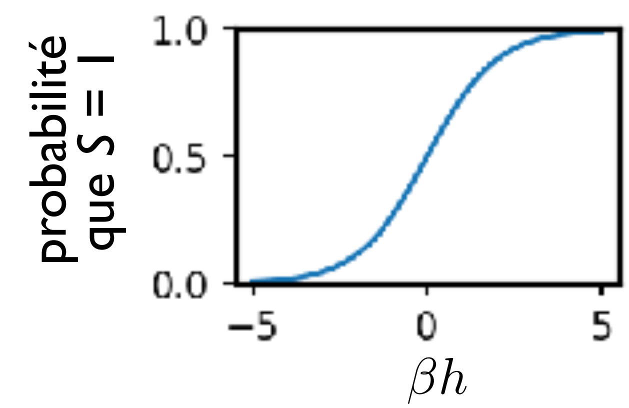
$$m^\mu(t) = \frac{1}{N} \sum_i P_i^\mu S_i(t)$$

Le modèle de Hopfield

Modèle pionnier d'un réseau de neurone avec attracteur

- Chaque neurone *i* est représenté par une variable *binaire*, $S_i \in \{-1, 1\}$, qui représente s'il est actif ou inactif dans un interval de temps Δt .
- L'activité à $t + \Delta t$ dépend de l'état au temps précédent t selon

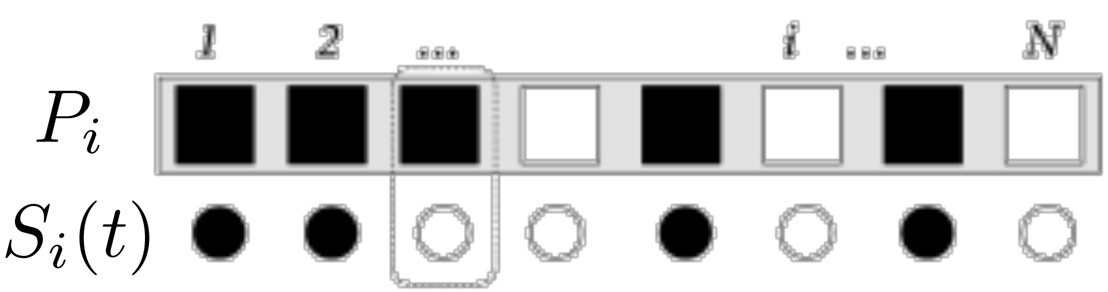
$$\text{prob}[S_i(t + \Delta t) = 1] = \frac{1}{1 + e^{-\beta h_i(t)}}$$



avec les entrées $h_i = \sum_{j=1 \ldots N} w_{ij} S_j$.

matrice de connexion

- La mémoire à retenir sont les motifs $P_i^\mu \in \{-1, 1\}$: $\mu = 1 \ldots K$



motif (immuable)

état (peut évoluer)

<u>"*overlap*" (dégré d'accord) :</u>

$$m^\mu(t) = \frac{1}{N} \sum_i P_i^\mu S_i(t)$$

- <u>Résultat théorique :</u> pour récupérer un motif à partir d'un petit overlap, utiliser $w_{ij} = \frac{1}{N} \sum_{\mu=1 \ldots K} P_i^\mu P_j^\mu$

"Fire together, wire together."

Modèle de Hopfield = minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

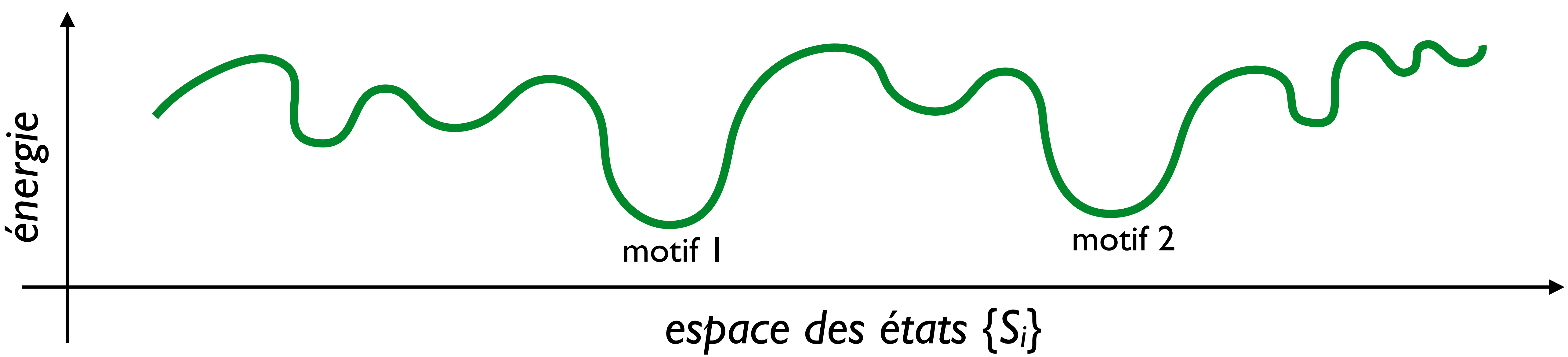
Modèle de Hopfield = minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



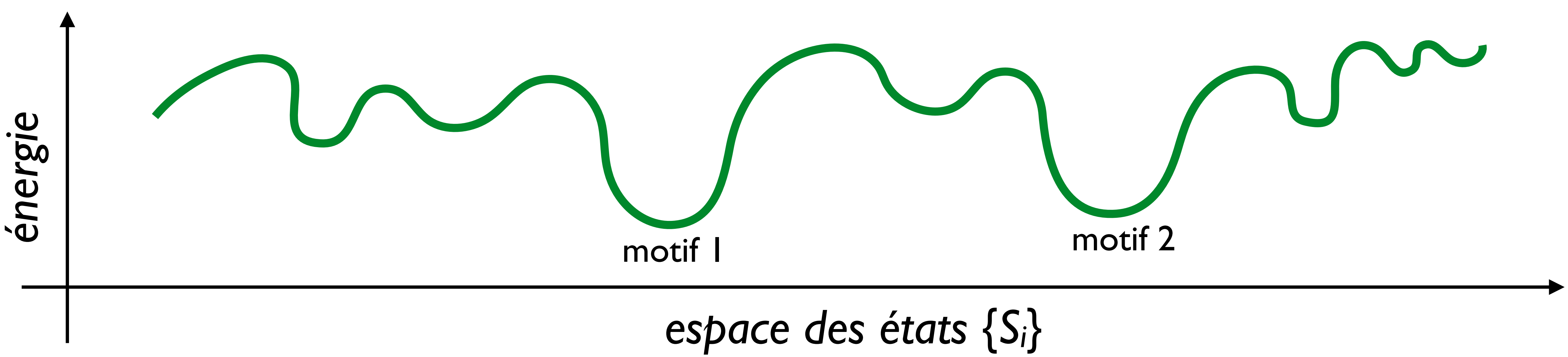
Modèle de Hopfield = minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



- La dynamique probabiliste (à température $1/\beta$) correspond à une dynamique de minimisation d'énergie libre du système.

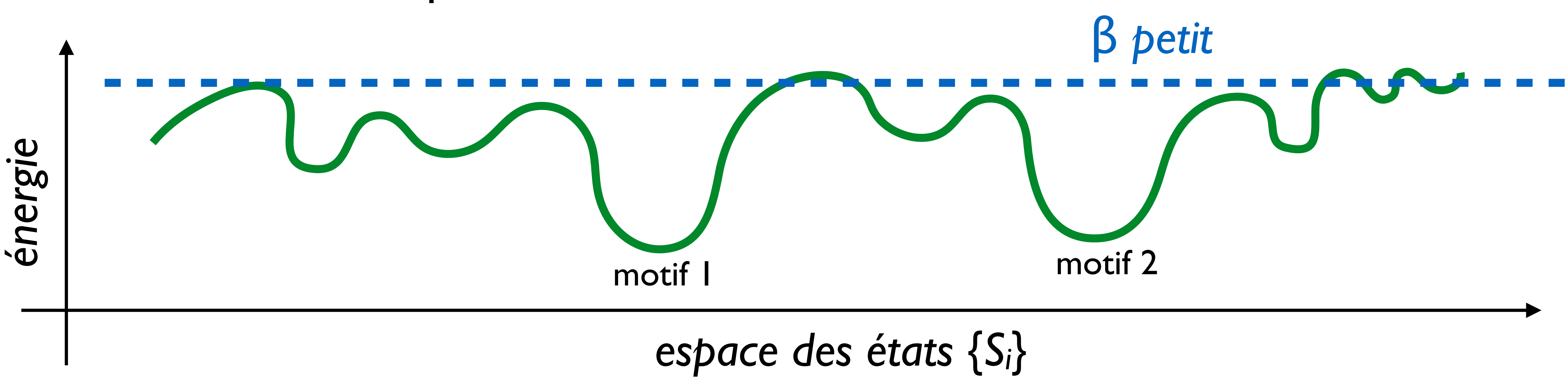
Modèle de Hopfield
= minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



- La dynamique probabiliste (à température 1/β) correspond à une dynamique de minimisation d'énergie libre du système.

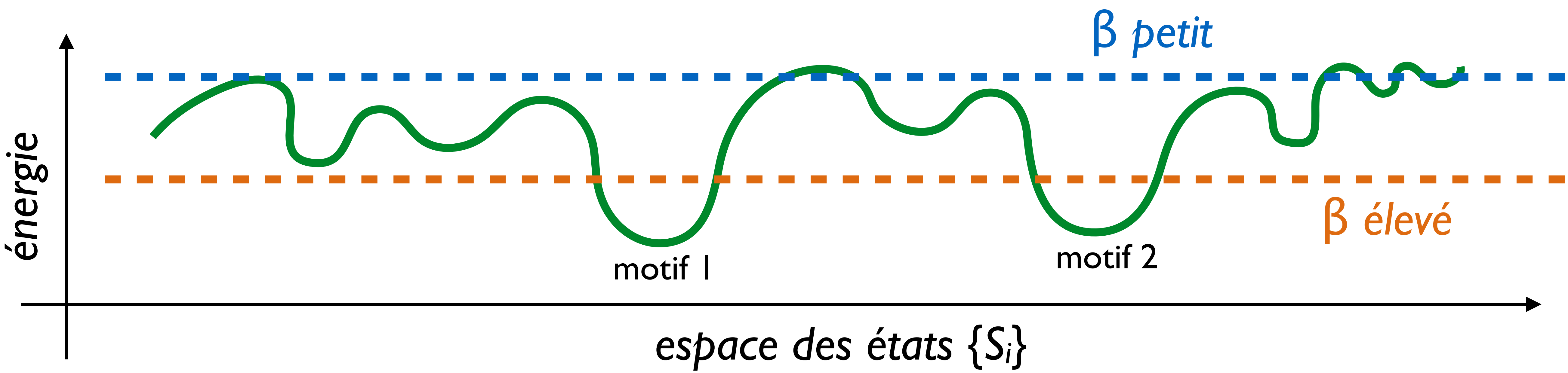
Modèle de Hopfield = minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



- La dynamique probabiliste (à température 1/β) correspond à une dynamique de minimisation d'énergie libre du système.

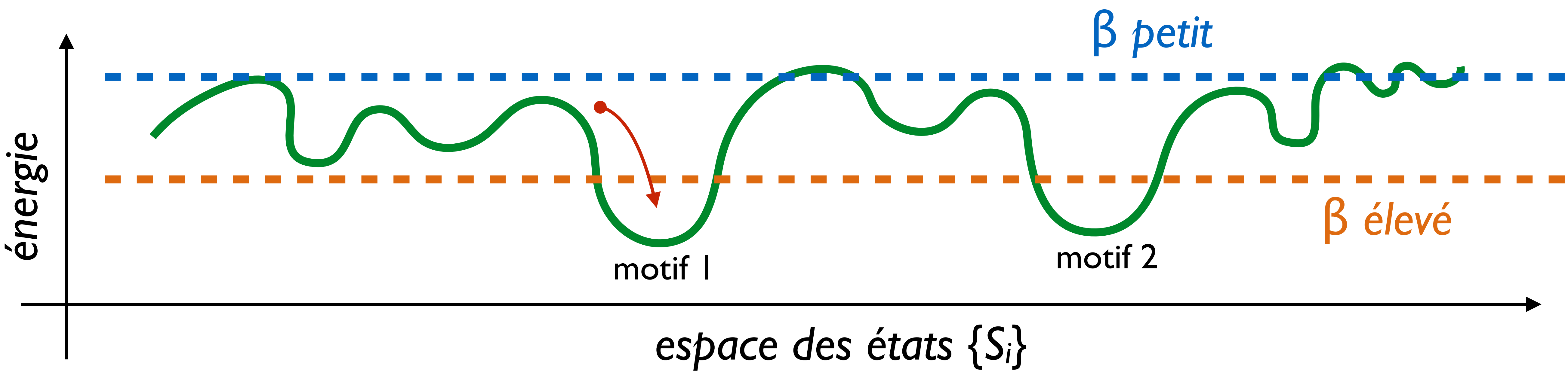
Modèle de Hopfield
= minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



- La dynamique probabiliste (à température 1/β) correspond à une dynamique de minimisation d'énergie libre du système.

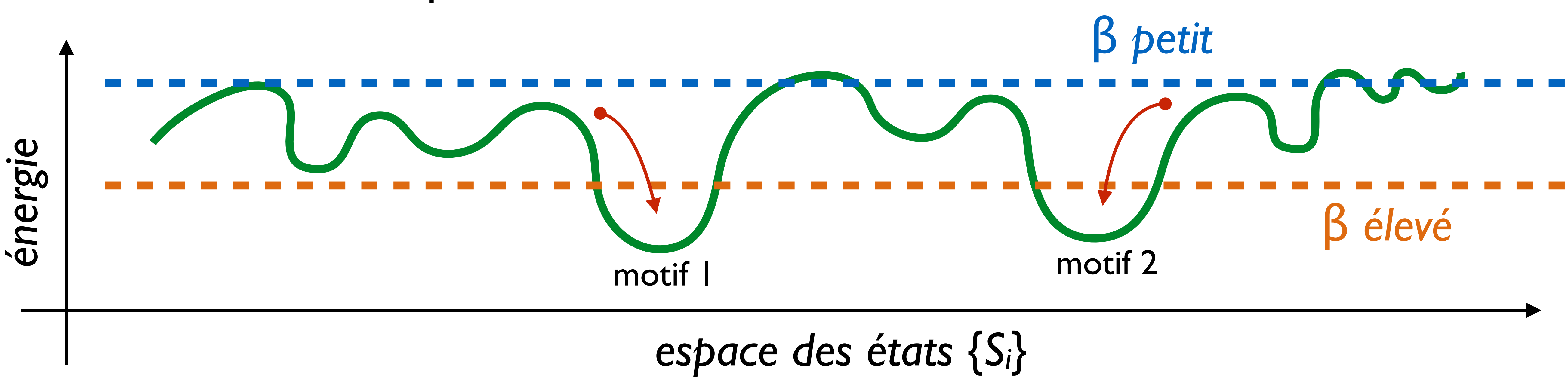
Modèle de Hopfield
= minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



- La dynamique probabiliste (à température 1/β) correspond à une dynamique de minimisation d'énergie libre du système.

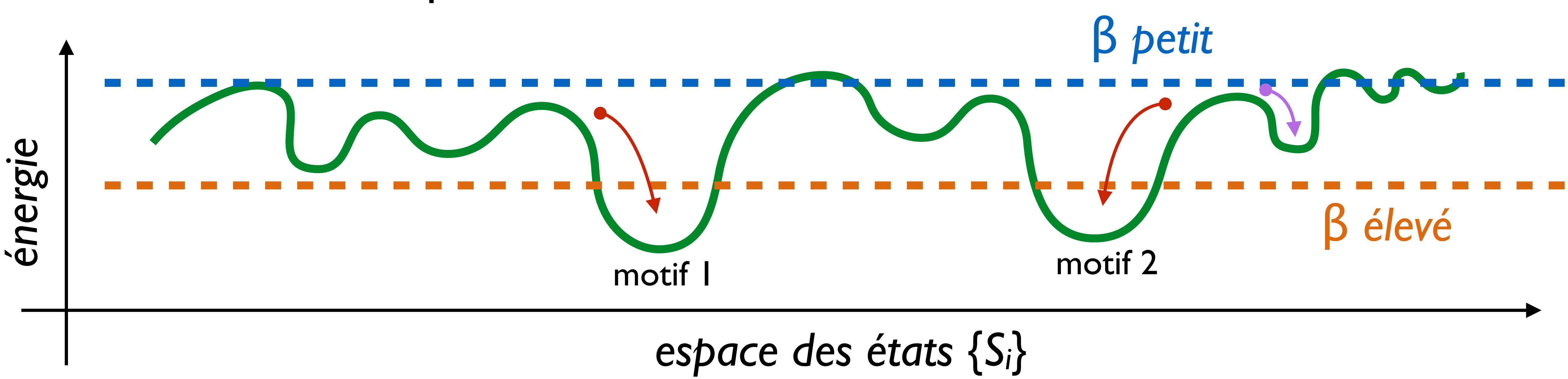
Modèle de Hopfield = minimisation d'énergie

Motifs stockés : bassins d'attraction d'une dynamique de minimisation d'énergie.

- Considérons la fonction qui représente l' "énergie" du réseau :

$$E(\{S_i\}) = -\sum_{i,j} w_{ij} S_i S_j$$

- Le "paysage énergétique" à *N* dimensions peut être très complexe, où les minima correspondent à des motifs :



- La dynamique probabiliste (à température 1/β) correspond à une dynamique de minimisation d'énergie libre du système.

Réseaux et apprentissage

Ce que nous savons déjà :

- Le plasticité structurel et/ou synaptique permet la modification des connexions synaptiques.
- Les connexions synaptiques peuvent servir comme *substrat de mémoire* : A partir d'un "indice", la dynamique du réseau (l'activité neuronale) évolue alors vers un *état attracteur* qui représente cette information. *Par ex. modélisé par le modèle de Hopfield.*

Réseaux et apprentissage

Ce que nous savons déjà :

- Le plasticité structurel et/ou synaptique permet la modification des connexions synaptiques.
- Les connexions synaptiques peuvent servir comme *substrat de mémoire* :
  A partir d'un "indice", la dynamique du réseau (l'activité neuronale) évolue alors vers un *état attracteur* qui représente cette information.
  Par ex. modélisé par le modèle de Hopfield.

Ce que nous n'avons pas encore étudié :

- Comment **apprendre** les connexions synaptiques qui permettent à un réseau de remplir une fonction donnée ?

Réseaux et apprentissage

Ce que nous savons déjà :

- Le plasticité structurel et/ou synaptique permet la modification des connexions synaptiques.
- Les connexions synaptiques peuvent servir comme *substrat de mémoire* : A partir d'un "indice", la dynamique du réseau (l'activité neuronale) évolue alors vers un *état attracteur* qui représente cette information. *Par ex. modélisé par le modèle de Hopfield.*

Ce que nous n'avons pas encore étudié :

- Comment **apprendre** les connexions synaptiques qui permettent à un réseau de remplir une fonction donnée ?

Ce qu'il faut distinguer :

- L'apprentissage *supervisé* de l'apprentissage *non-supervisé*, ainsi que l'apprentissage *par renforcement*.
- Les *modèles d'apprentissage biologiques* d'une part, et les *algorithmes* qui *en principe* permettent le bon apprentissage d'autre part.

Réseaux et apprentissage

Ce que nous savons déjà :

- Le plasticité structurel et/ou synaptique permet la modification des connexions synaptiques.
- Les connexions synaptiques peuvent servir comme *substrat de mémoire* :
  A partir d'un "indice", la dynamique du réseau (l'activité neuronale) évolue alors vers un *état attracteur* qui représente cette information.
  Par ex. modélisé par le modèle de Hopfield.

Ce que nous n'avons pas encore étudié :

- Comment **apprendre** les connexions synaptiques qui permettent à un réseau de remplir une fonction donnée ?

Ce qu'il faut distinguer :

aujourd'hui

- L'apprentissage *supervisé* de l'apprentissage *non-supervisé*, ainsi que l'apprentissage *par renforcement.*
- Les *modèles d'apprentissage biologiques* d'une part, et les *algorithmes* qui *en principe* permettent le bon apprentissage d'autre part.

Les différents types d’apprentissage

- Apprentissage ***non-supervisé*** : Les stimuli externes présentés au réseau entraînent, ensemble avec l’activité du réseau, des changements synaptiques sans d’autre intervention ou signal guidant la plasticité.
  - Exemple : construction d’une représentation neuronale d’un stimulus par apprentissage Hebbien

Les différents types d'apprentissage

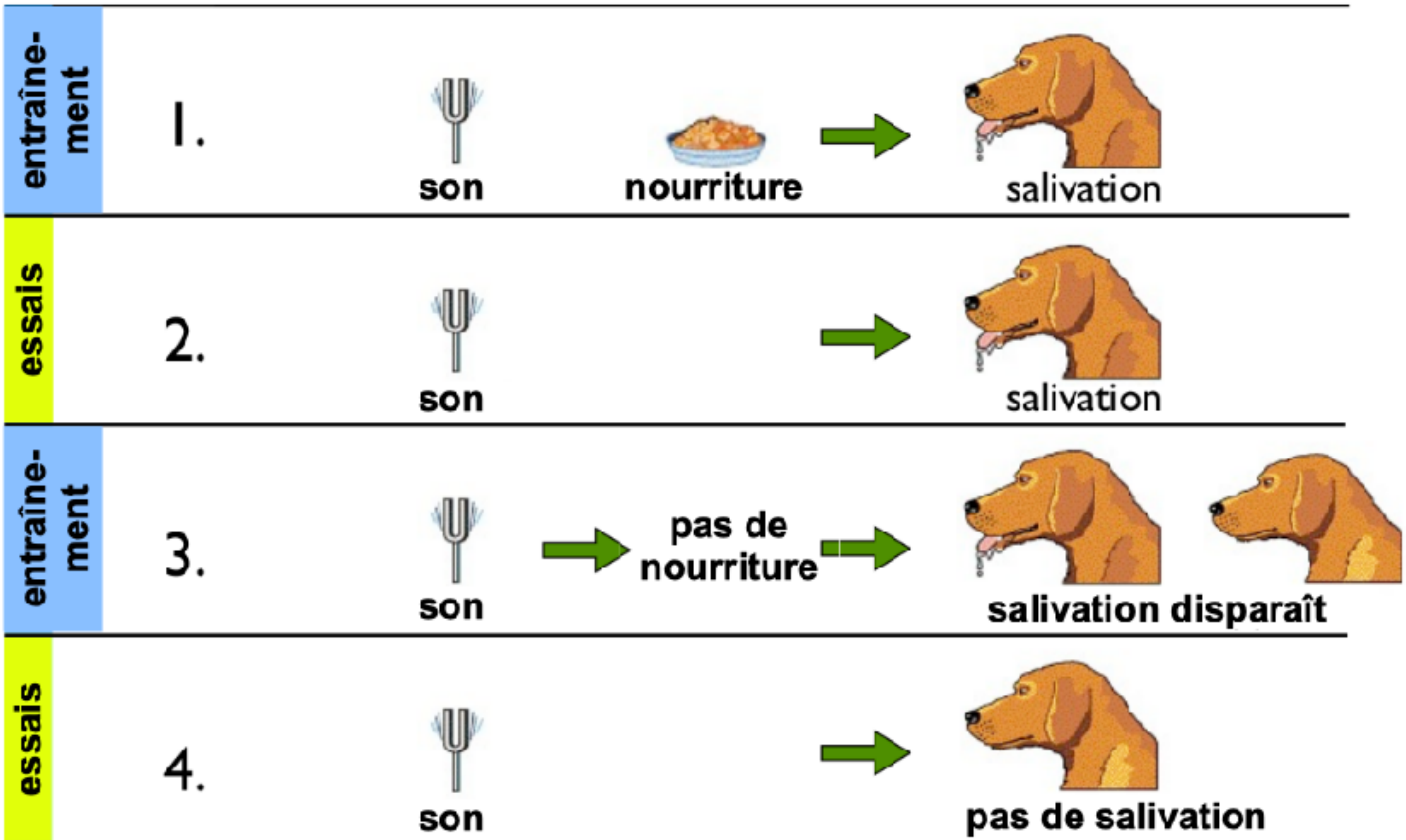
- Apprentissage ***non-supervisé*** : Les stimuli externes présentés au réseau entraînent, ensemble avec l'activité du réseau, des changements synaptiques sans d'autre intervention ou signal guidant la plasticité.
  - Exemple : construction d'une représentation neuronale d'un stimulus par apprentissage Hebbien
- Apprentissage ***supervisé*** : Les connexions synaptiques sont modifiés en fonction du résultat d'une classification, régression etc. d'un, en connaissance du résultat souhaité.
  - Exemples : "deep learning" (réseau multi-couche pour classifier par ex. des images, où l'image et l'objet à reconnaître sont connus lors de l'entrainement), *perceptron*

Les différents types d’apprentissage

- Apprentissage *non-supervisé* : Les stimuli externes présentés au réseau entraînent, ensemble avec l’activité du réseau, des changements synaptiques sans d’autre intervention ou signal guidant la plasticité.
  - Exemple : construction d’une représentation neuronale d’un stimulus par apprentissage Hebbien
- Apprentissage *supervisé* : Les connexions synaptiques sont modifiés en fonction du résultat d’une classification, régression etc. d’un, en connaissance du résultat souhaité.
  - Exemples : "deep learning” (réseau multi-couche pour classifier par ex. des images, où l’image et l’objet à reconnaître sont connus lors de l’entrainement), *perceptron*
- Apprentissage *par renforcement* : Apprendre la bonne réponse à un stimulus sur la base des récompenses ou punitions y associées.
  - Exemple : conditionnement classique

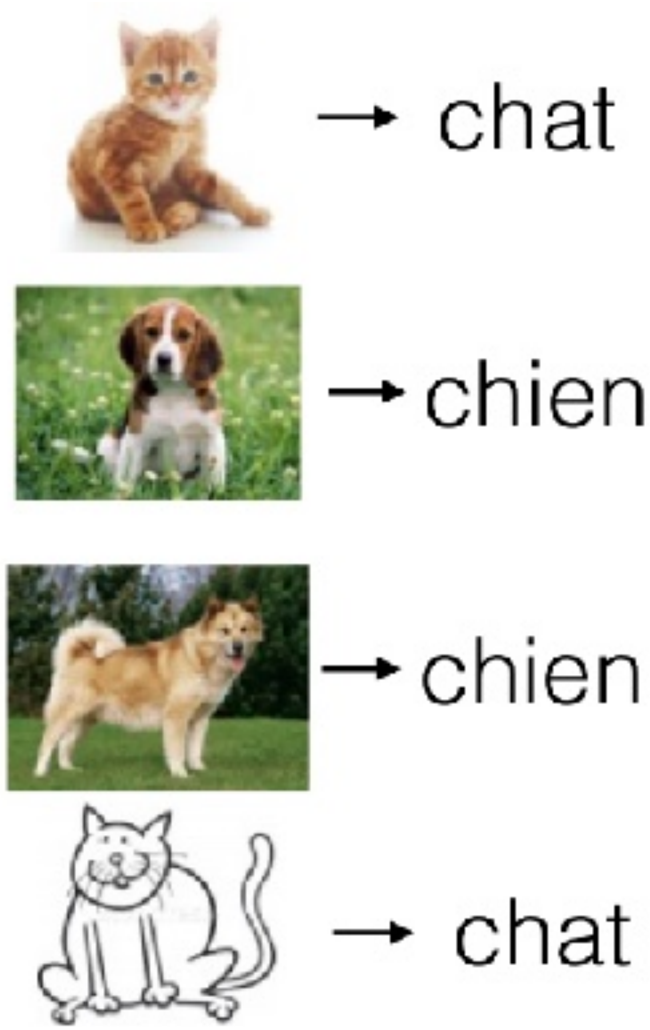
Apprentissage par renforcement

Exemple : l’apprentissage (conditionnement) du “réflexe Pavlovien”



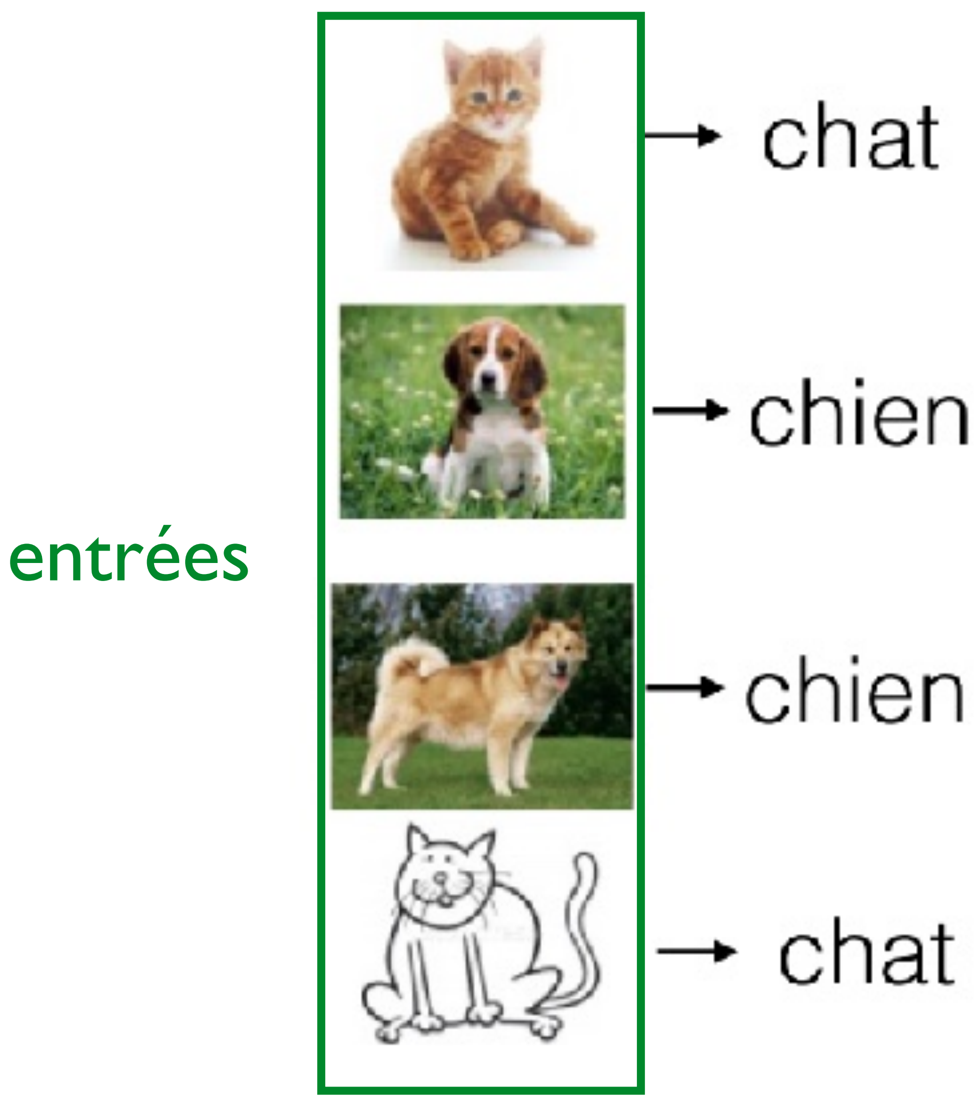
Apprentissage supervisé

- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.



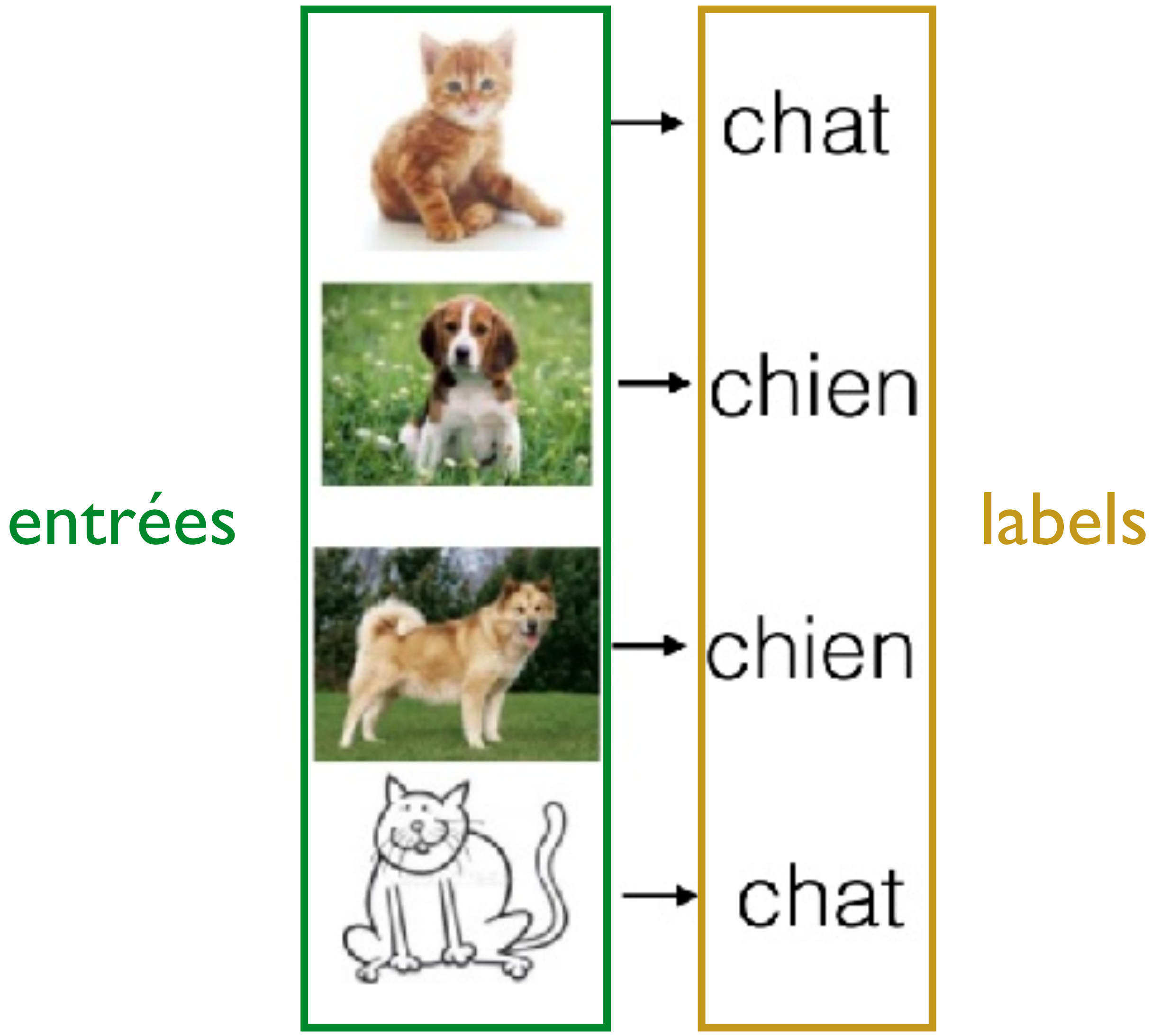
Apprentissage supervisé

- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.



Apprentissage supervisé

- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.

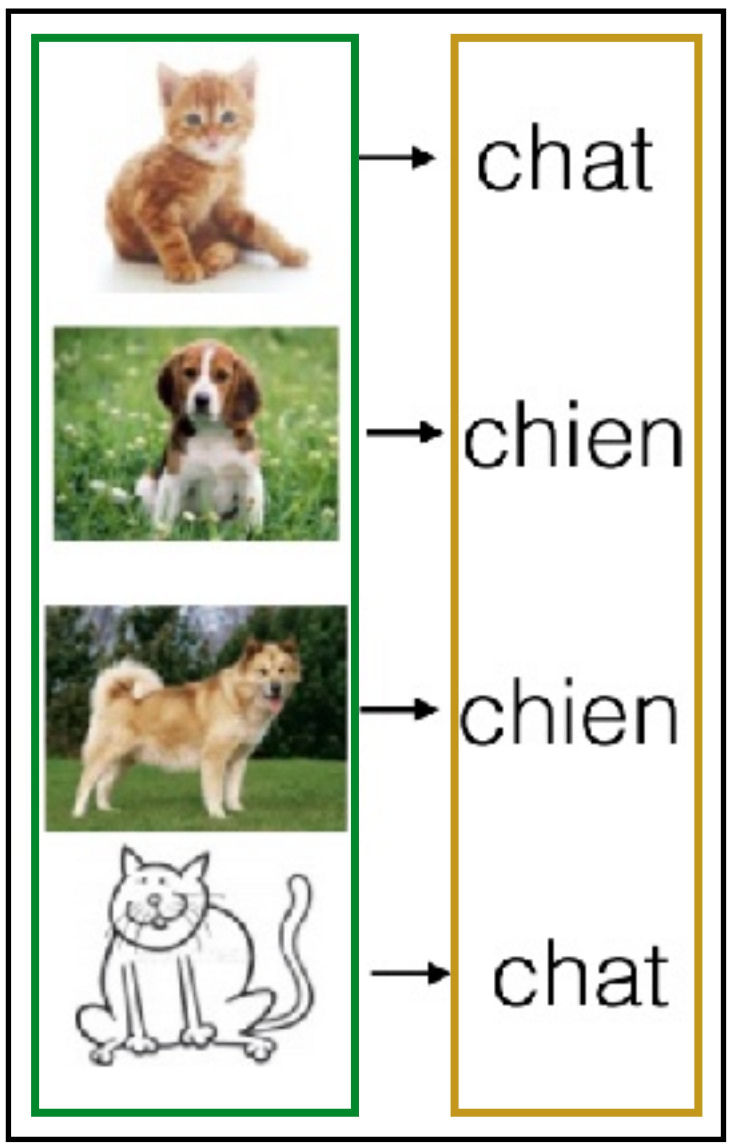


Apprentissage supervisé

- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.

ensemble d'entraînement

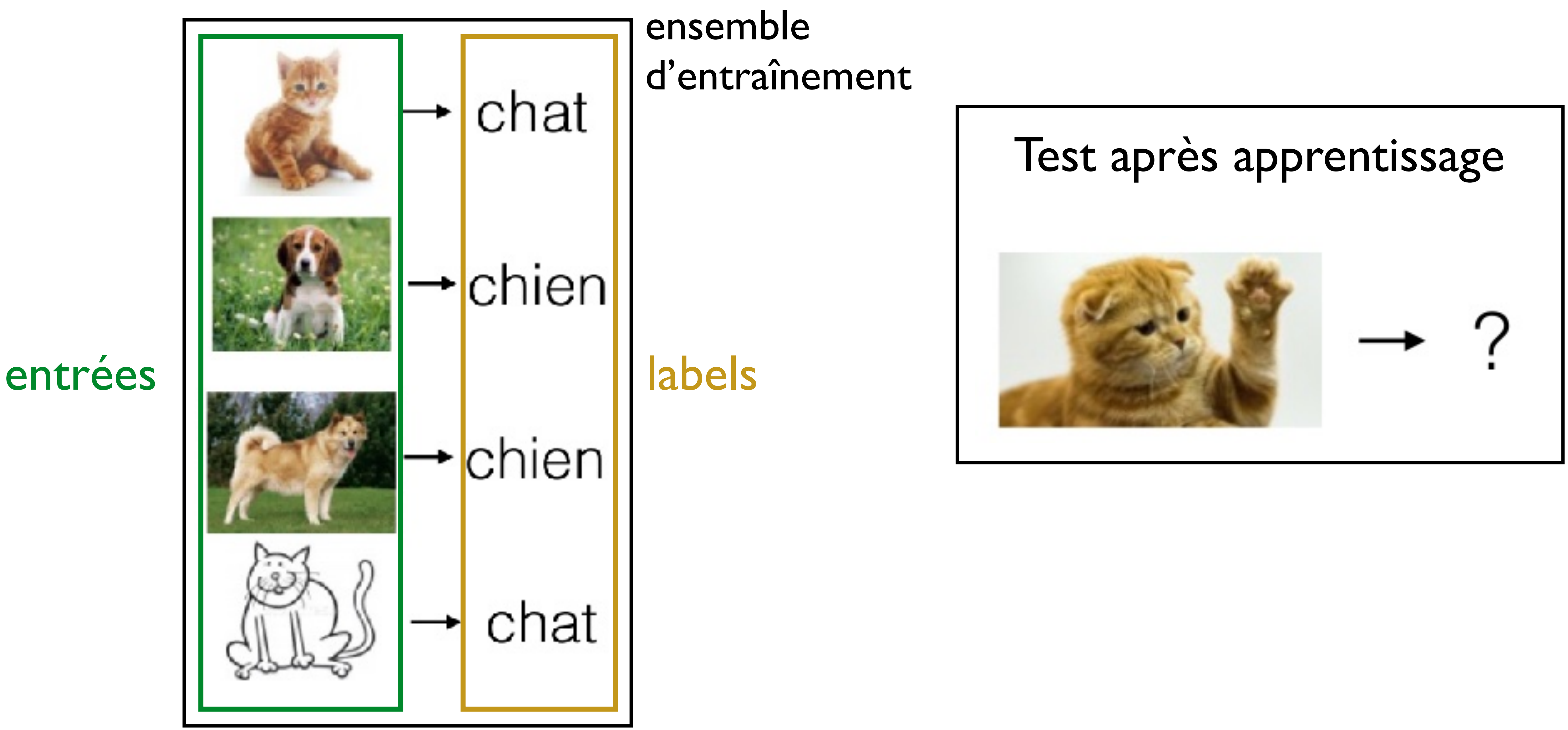
entrées



labels

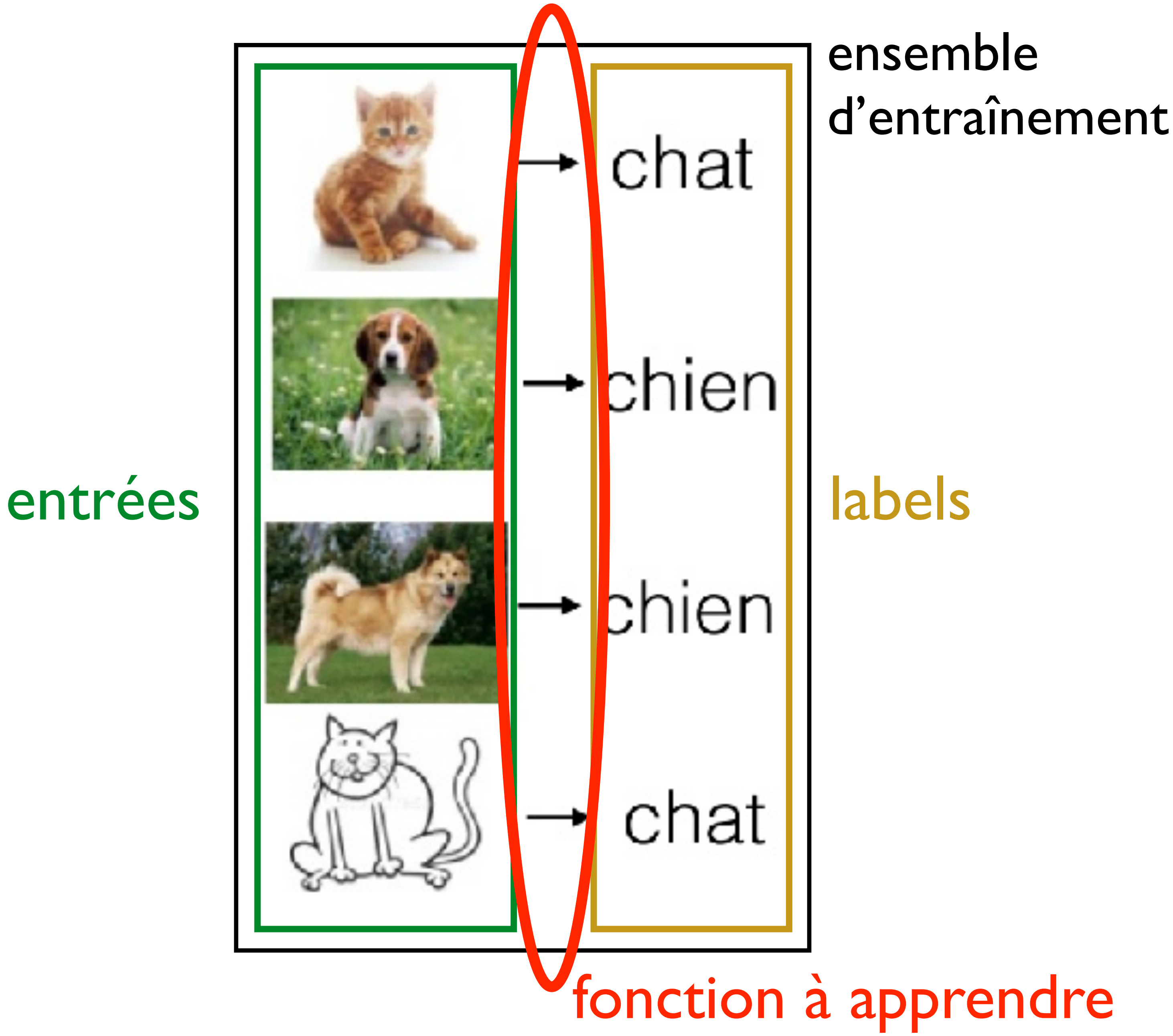
Apprentissage supervisé

- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.



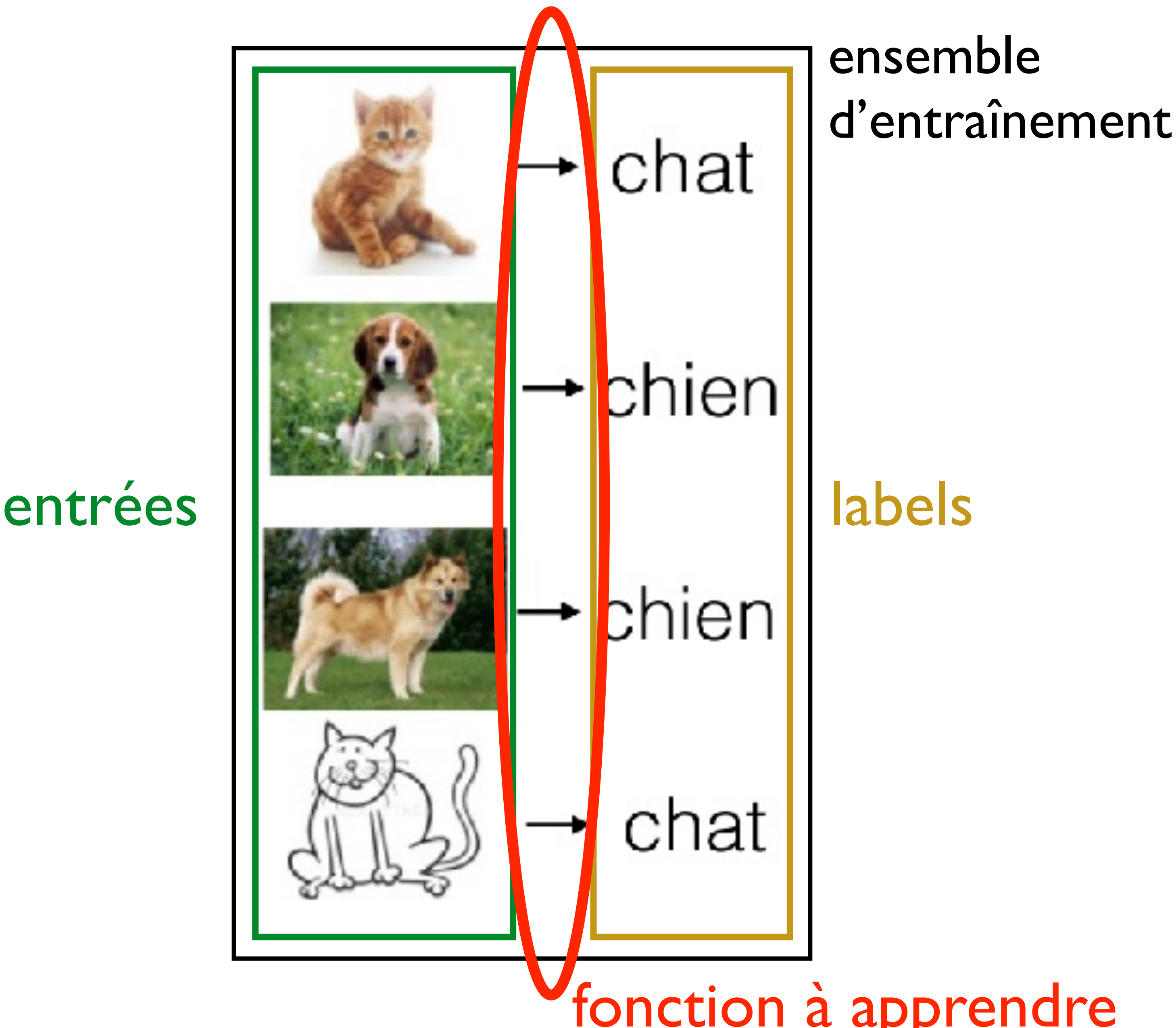
Apprentissage supervisé

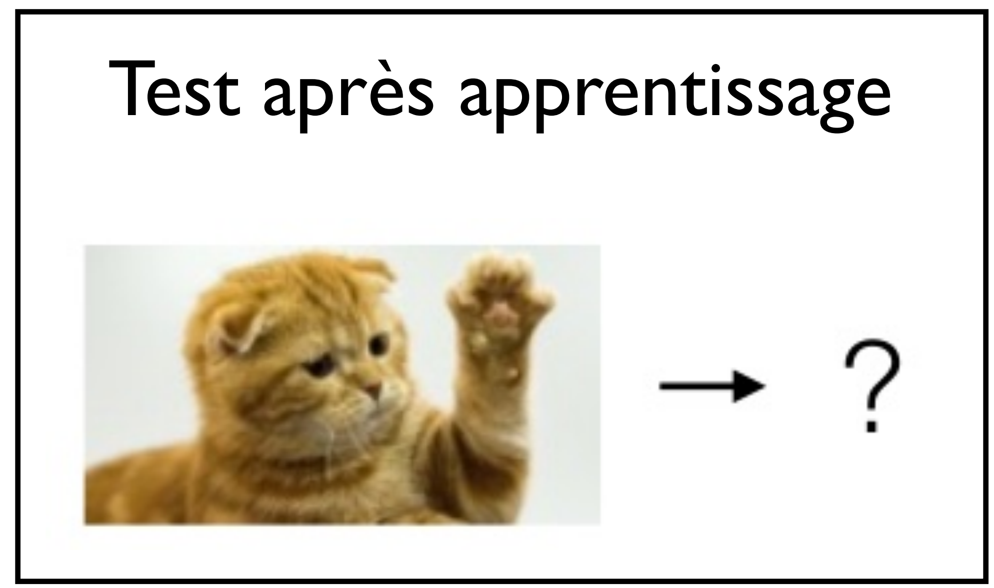
- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.



Apprentissage supervisé

- **But** : Apprendre une fonction *F* sur les entrées à partir d'un ensemble de données, par ex. pour classifier des images.
- **Intérêt** : Utiliser la fonction apprise sur de nouvelles entrées (jeux de données) qui n'ont pas été utilisées lors de l'entraînement.



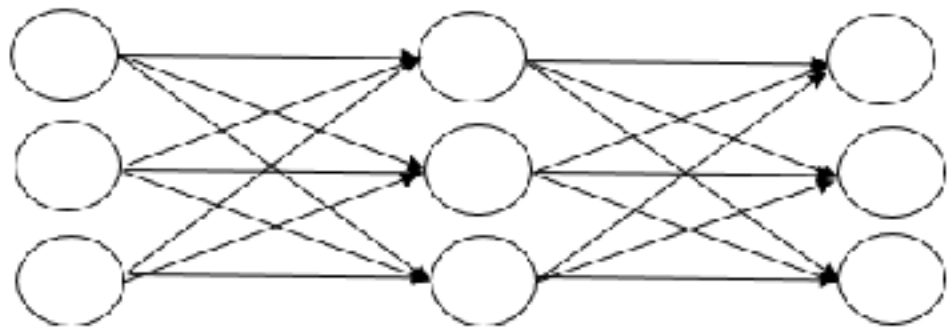


Difficulté : L'apprentissage peut nécessiter un ensemble de données d'entraînement très conséquent.

Les réseaux “feed-forward”

Structure des réseaux sans connexions récurrentes

- Lorsque les neurones sont organisés en couches sans connexions récurrentes, on parle d’un réseau “feed-forward”, où les sorties des neurones d’une couche sont les entrées de la couche suivante :

. . .  . . .

- Un réseau “feed-forward" à une couche comprend une *couche d’entrée* (N_{in} neurones) et une *couche de sortie* (N_{in} neurones).
- L’apprentissage dit “profond” consiste à apprendre les poids synaptiques des réseaux “feed-forward” à plusieurs couches.
- Par ex. dans le traitement initial des entrées visuelles par la rétine et le cortex visuelle on peut également parler de réseaux “feed-forward” tant les connexions récurrentes ne rentrent pas en jeu.

Apprentissage supervisé : Le perceptron

The New York Times

NEW NAVY DEVICE LEARNS BY DOING

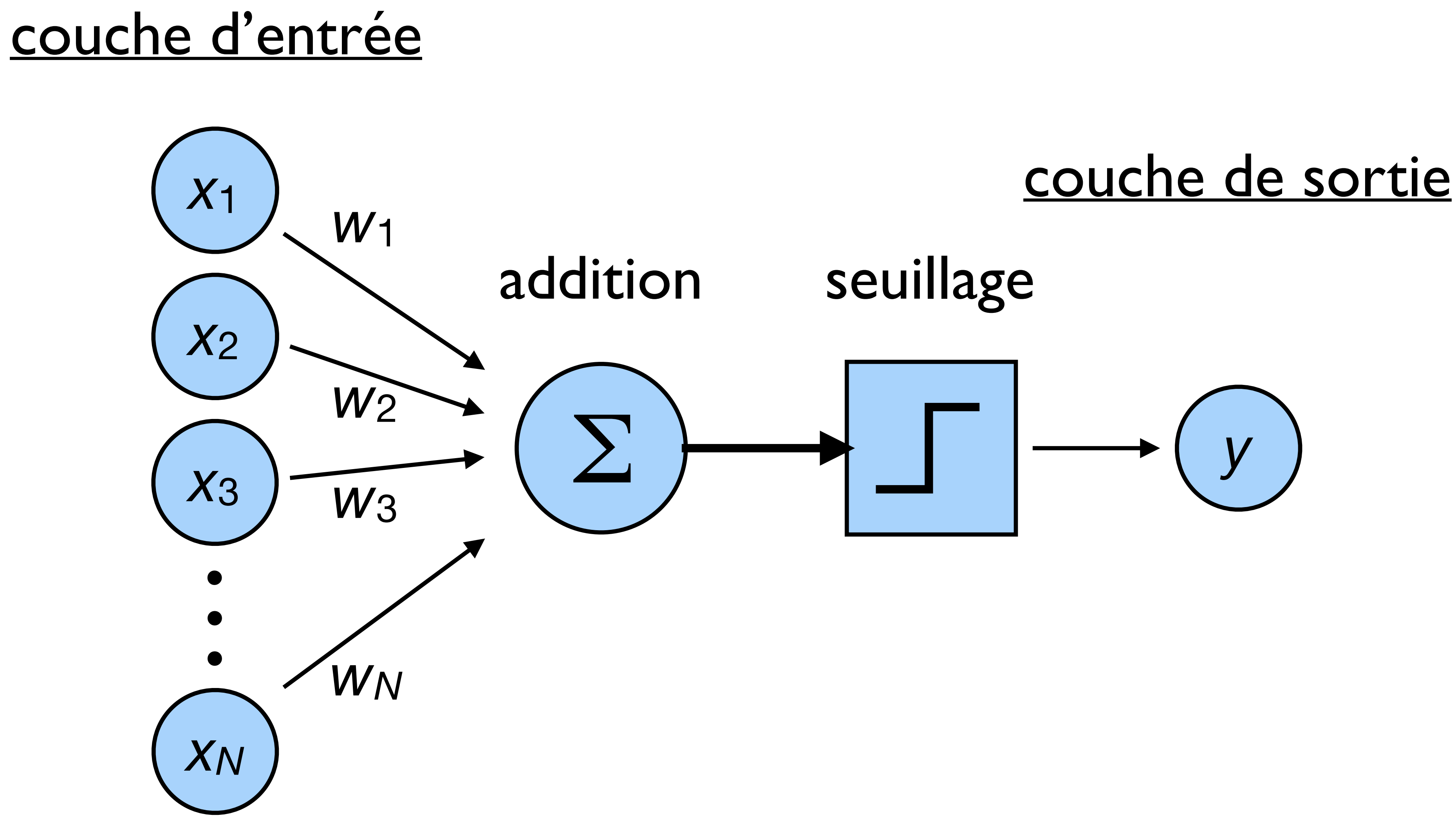
July 8, 1958

"The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence... Dr. Frank Rosenblatt, a research psychologist at the Cornell Aeronautical Laboratory, Buffalo, said Perceptrons might be fired to the planets as mechanical space explorers"

Frank Rosenblatt, 1958

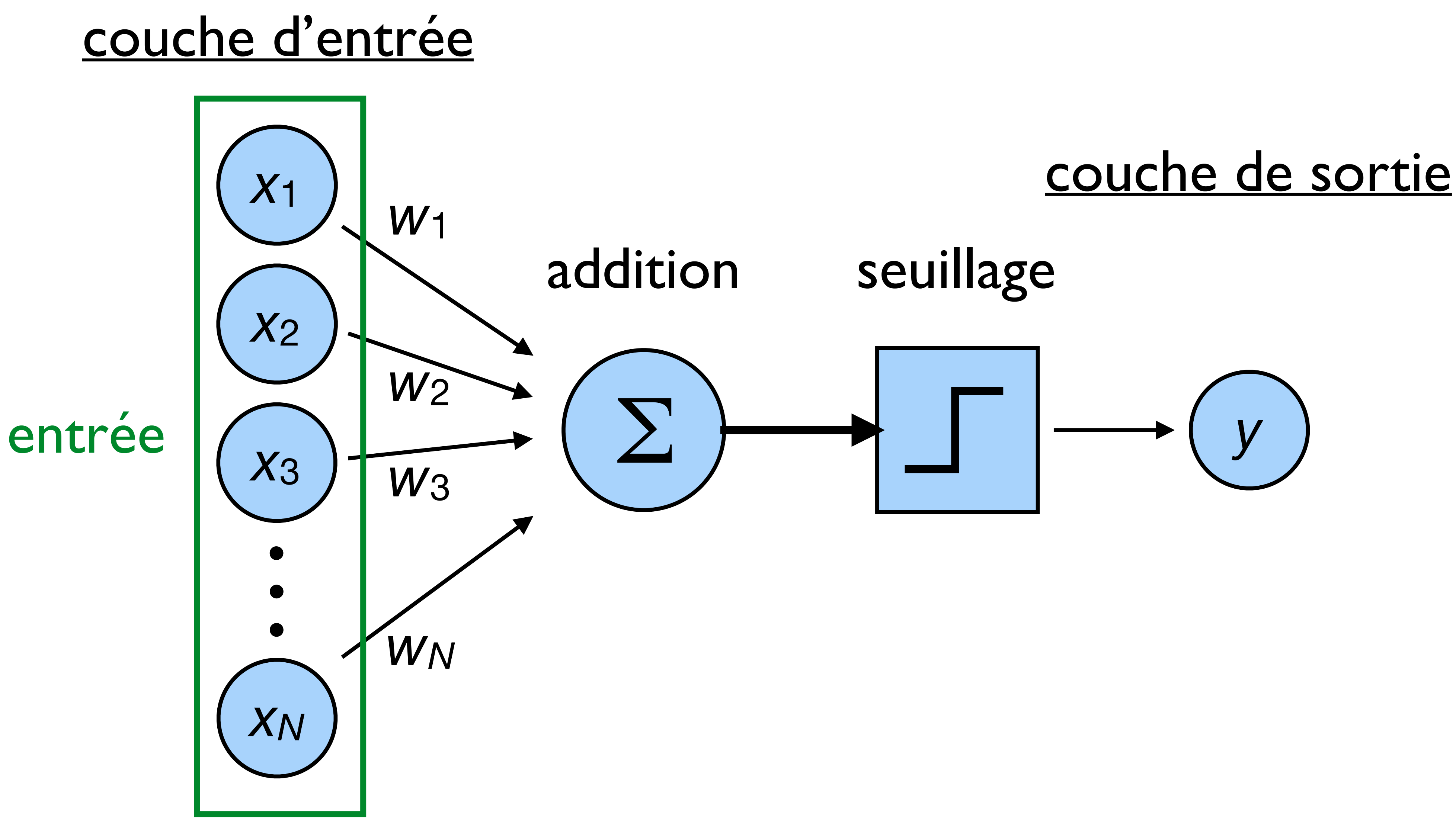
Apprentissage supervisé : Le perceptron

Une machine qui permet de faire des classifications binaires



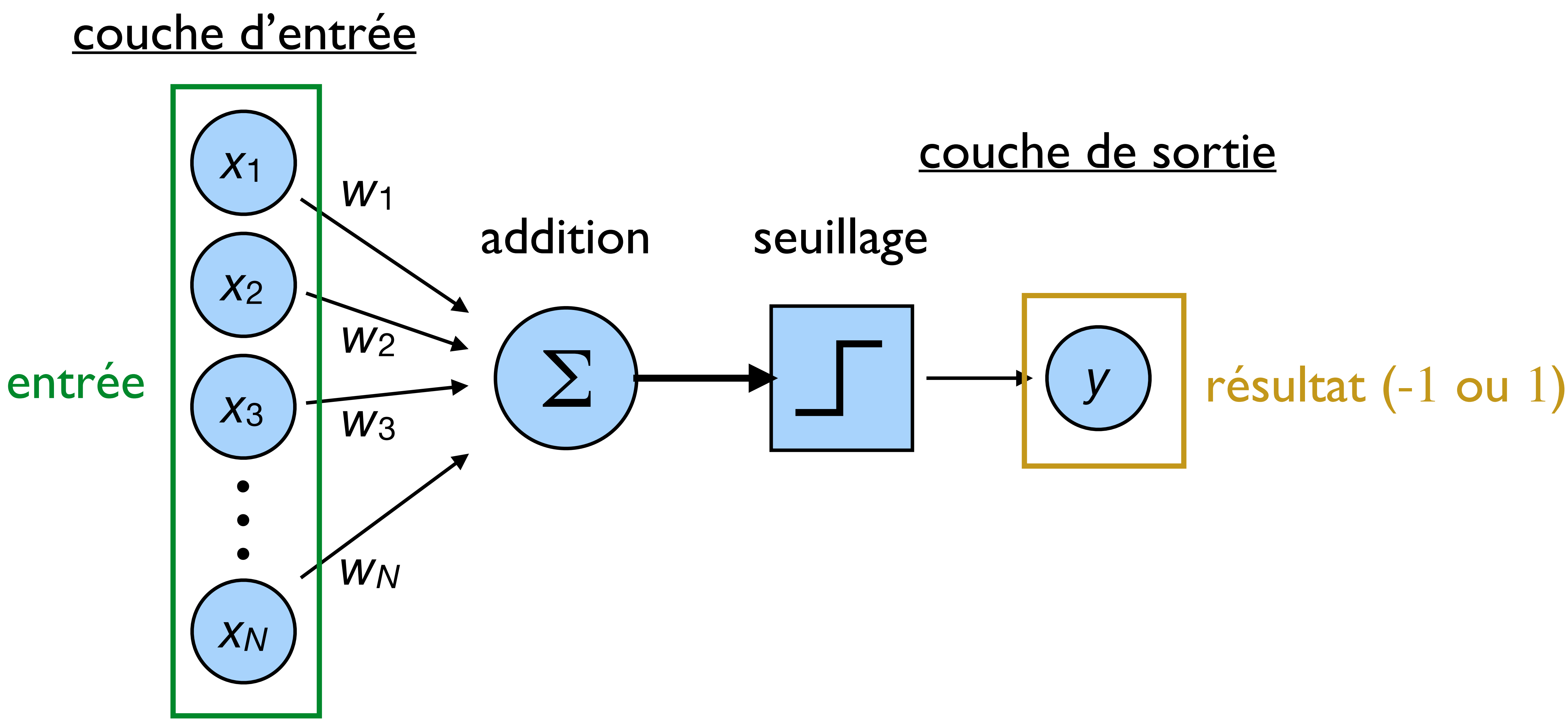
Apprentissage supervisé : Le perceptron

Une machine qui permet de faire des classifications binaires



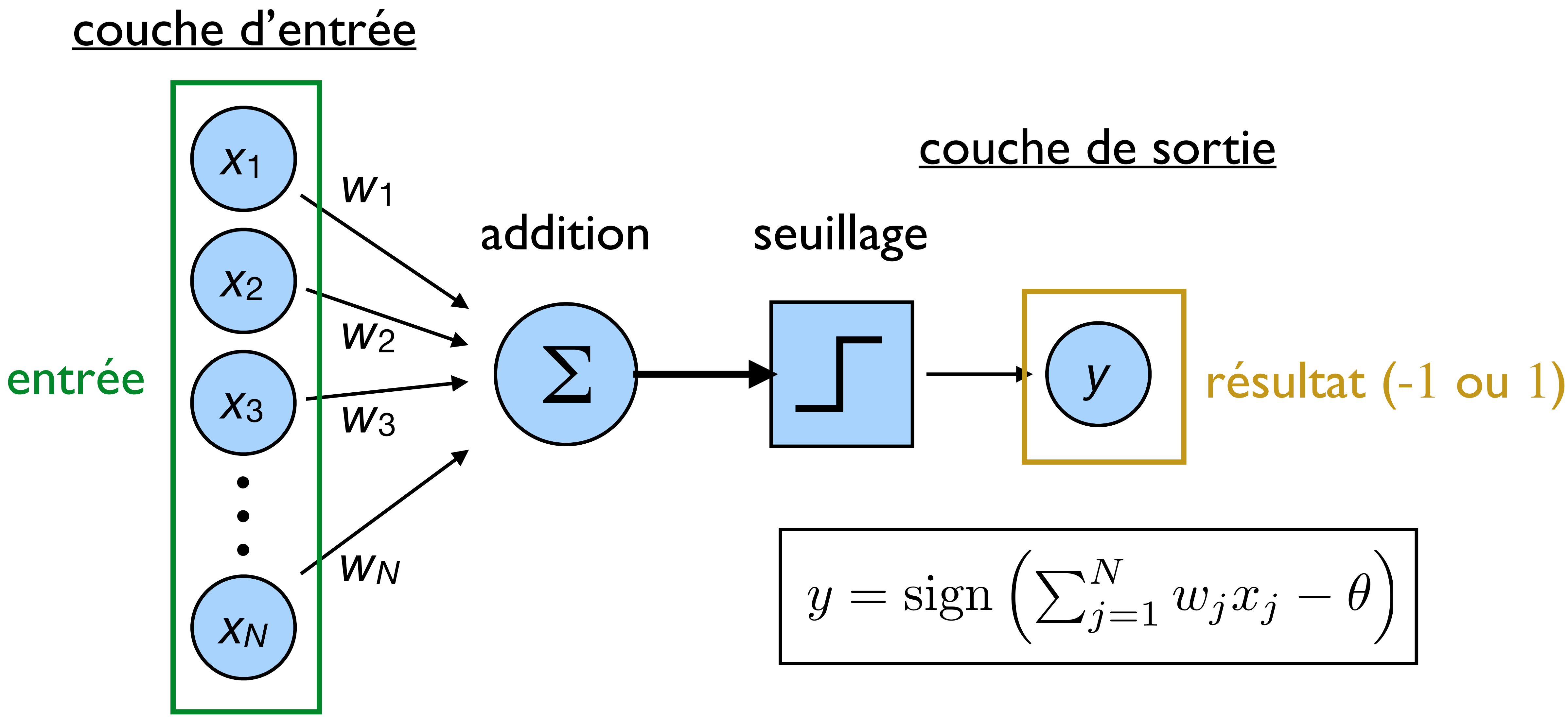
Apprentissage supervisé : Le perceptron

Une machine qui permet de faire des classifications binaires



Apprentissage supervisé : Le perceptron

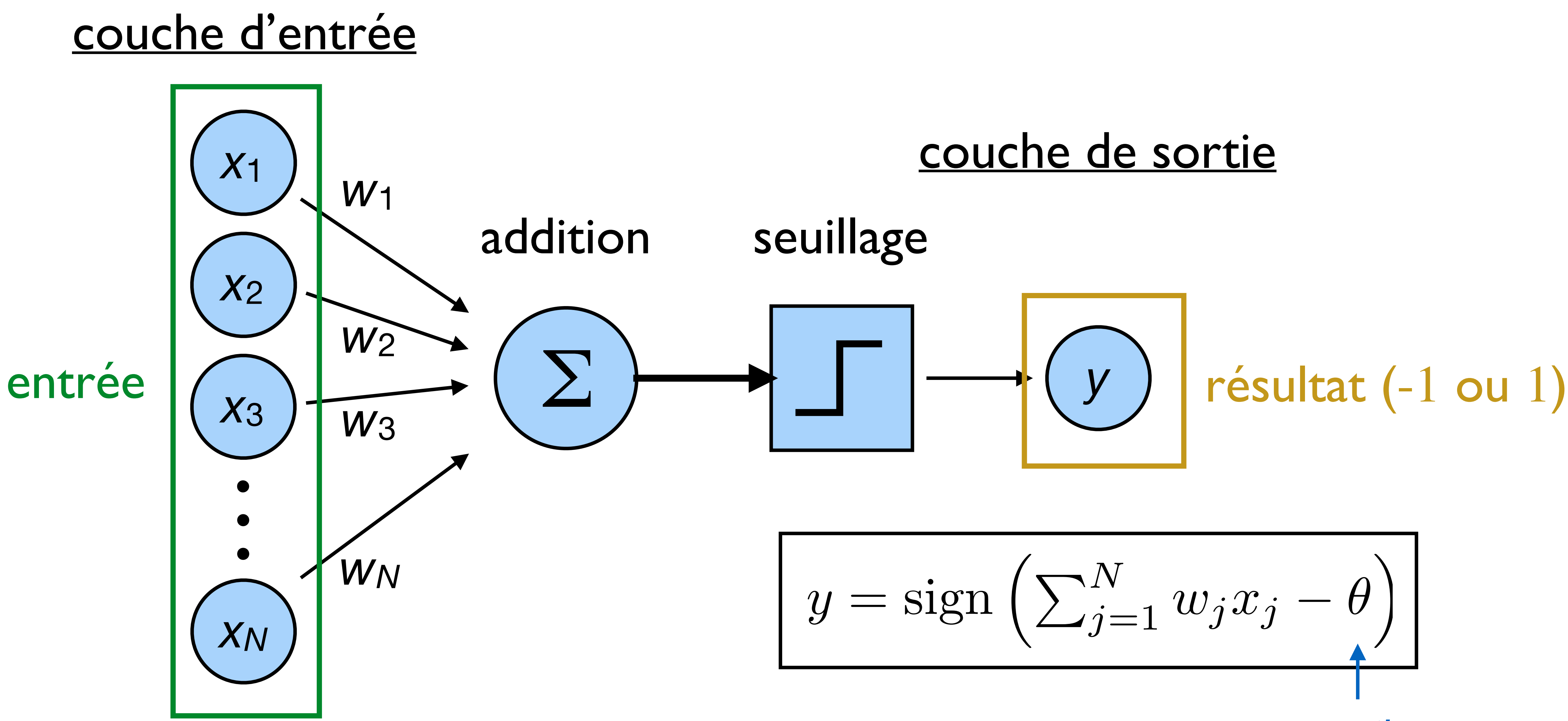
Une machine qui permet de faire des classifications binaires



$$y = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j - \theta\right)$$

Apprentissage supervisé : Le perceptron

Une machine qui permet de faire des classifications binaires

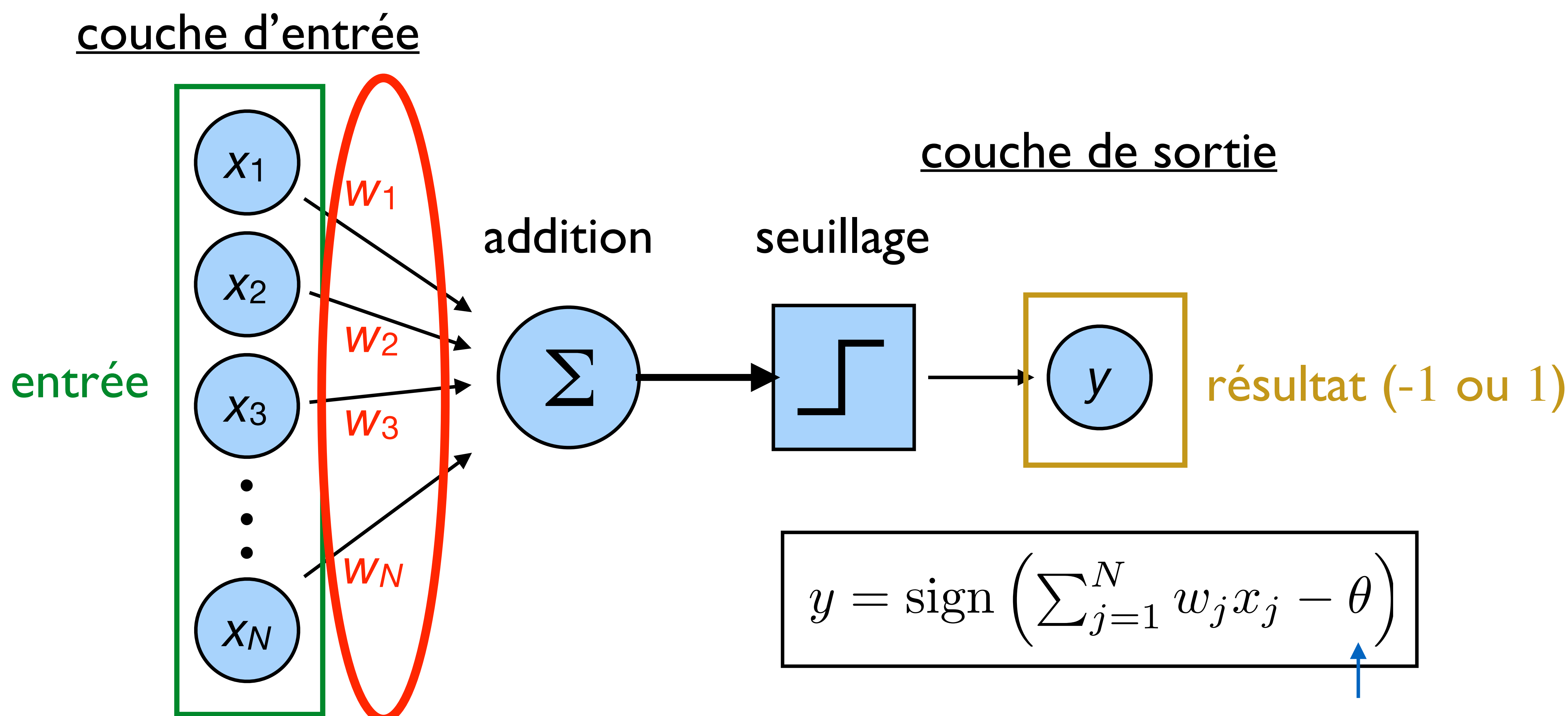


$$y = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j - \theta\right)$$

seuil

Apprentissage supervisé : Le perceptron

Une machine qui permet de faire des classifications binaires



$$y = \text{sign}\left(\sum_{j=1}^{N} w_j x_j - \theta\right)$$

seuil

poids à apprendre à partir de pairs "entrée—résultat" fournis

Apprentissage supervisé : Le perceptron

Une machine qui permet de faire des classifications binaires

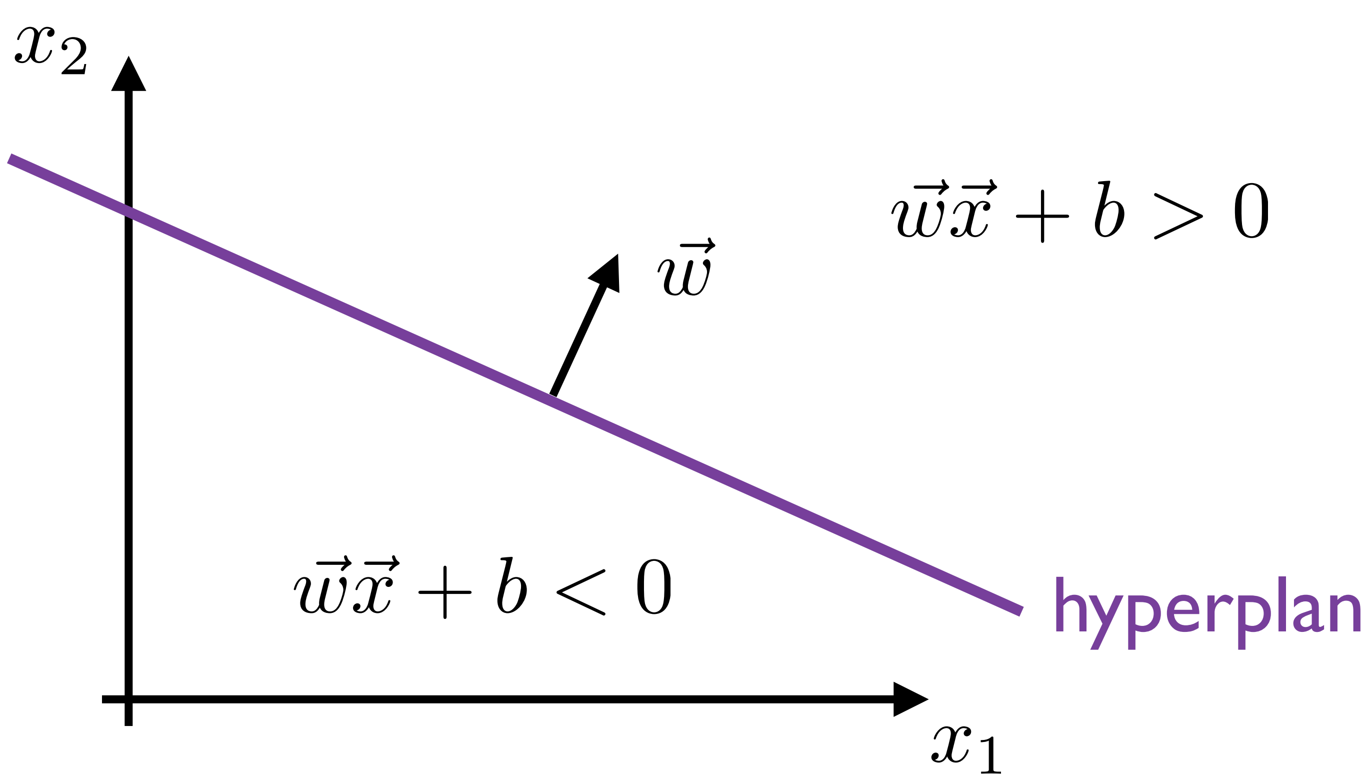
- Les perceptrons à une seule couche sont uniquement capables d'apprendre à distinguer des motifs *linéairement séparables*.

- Les réseaux "feed-forward" à deux ou plusieurs couches (aussi appelés *perceptrons multi-couches* si les neurones de chaque couche sont décrits par l'équation des perceptrons) sont beaucoup plus performants $\longrightarrow$ cf. apprentissage profond…

- Outil pour étudier les questions de capacité de stockage.

Le perceptron, interprétation géometrique

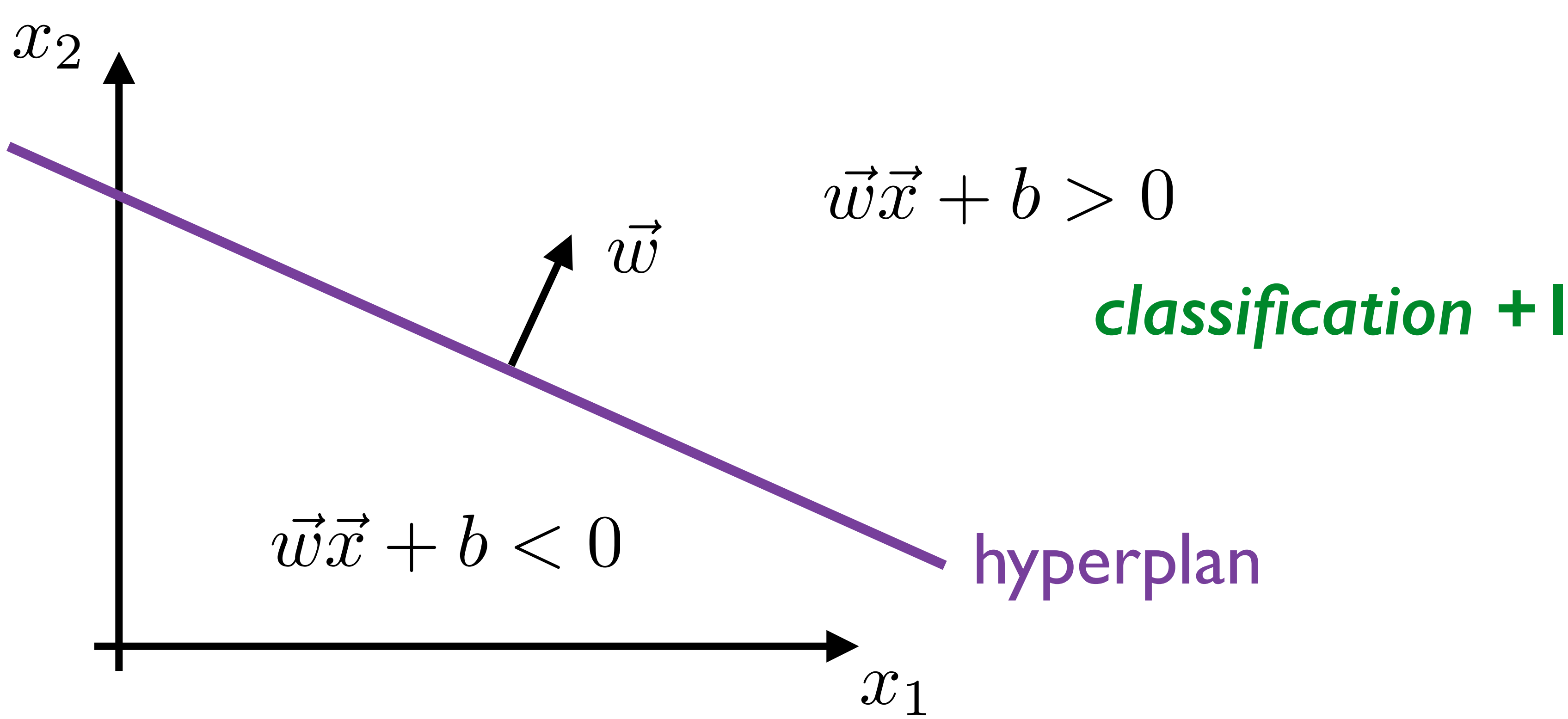
- Equation du perceptron : $y = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j - \theta\right)$
- Equation d'un hyperplan : $\vec{w}\vec{x} + b = 0$ $\quad (b \leftrightarrow -\theta)$



- Chaque entrée correspond à un point dans un espace de *N* dimensions.
- Deux ensembles de points sont *linéairement séparables* s'il existe un hyperplan (de dimension *N*-1) qui sépare les deux ensembles.

Le perceptron, interprétation géometrique

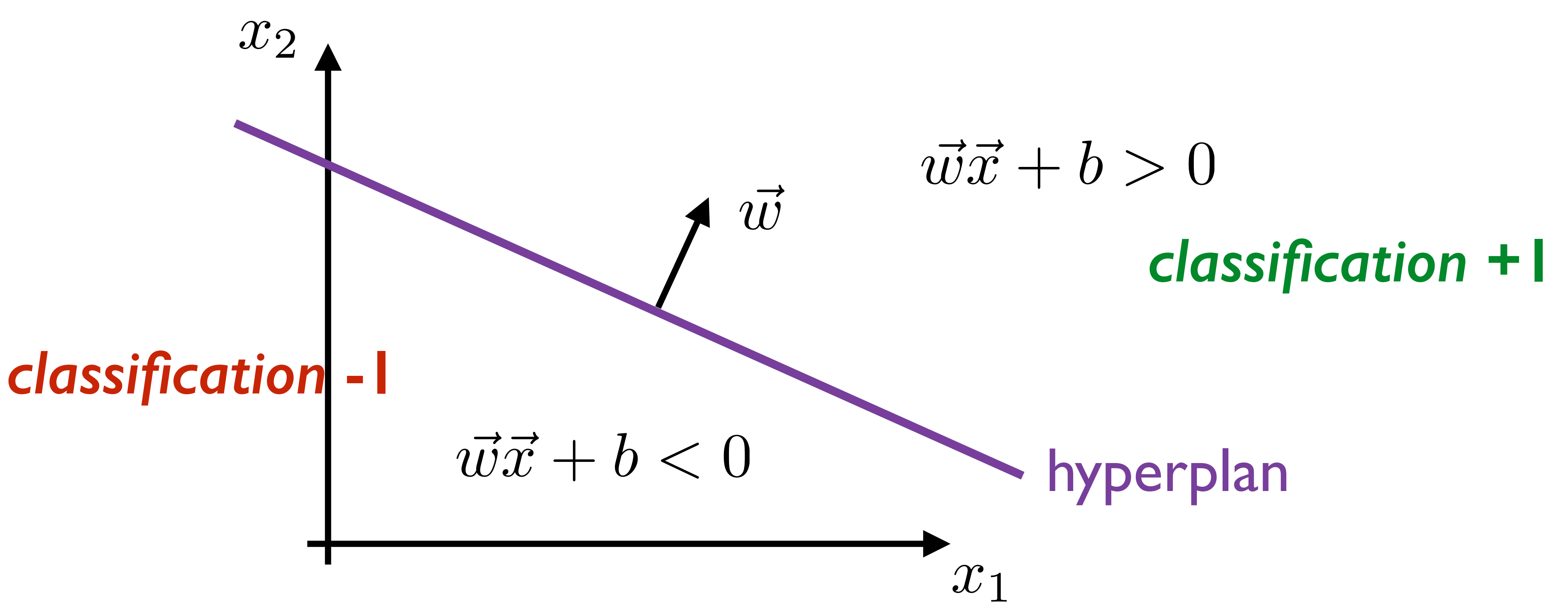
- Equation du perceptron : $y = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j - \theta\right)$
- Equation d'un hyperplan : $\vec{w}\vec{x} + b = 0$ $\quad (b \leftrightarrow -\theta)$



- Chaque entrée correspond à un point dans un espace de *N* dimensions.
- Deux ensembles de points sont *linéairement séparables* s'il existe un hyperplan (de dimension *N*-1) qui sépare les deux ensembles.

Le perceptron, interprétation géometrique

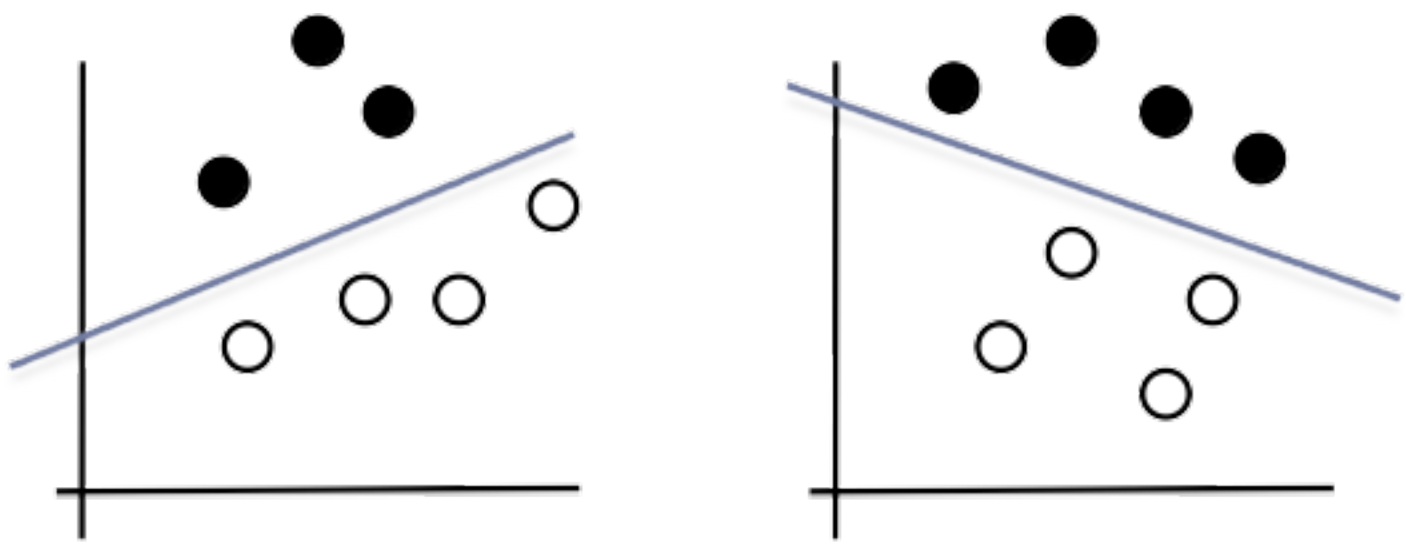
- Equation du perceptron : $y = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j - \theta\right)$
- Equation d'un hyperplan : $\vec{w}\vec{x} + b = 0$ $(b \leftrightarrow -\theta)$



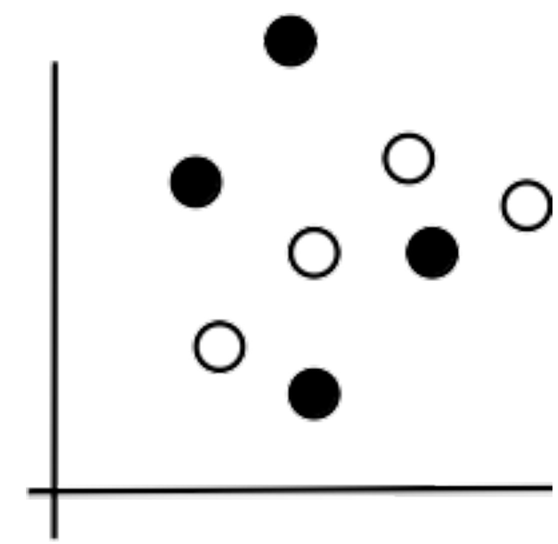
- Chaque entrée correspond à un point dans un espace de *N* dimensions.
- Deux ensembles de points sont *linéairement séparables* s'il existe un hyperplan (de dimension *N*-1) qui sépare les deux ensembles.

Séparabilité linéaire

Le perceptron peut distinguer des points linéairement séparables.



linéairement séparable



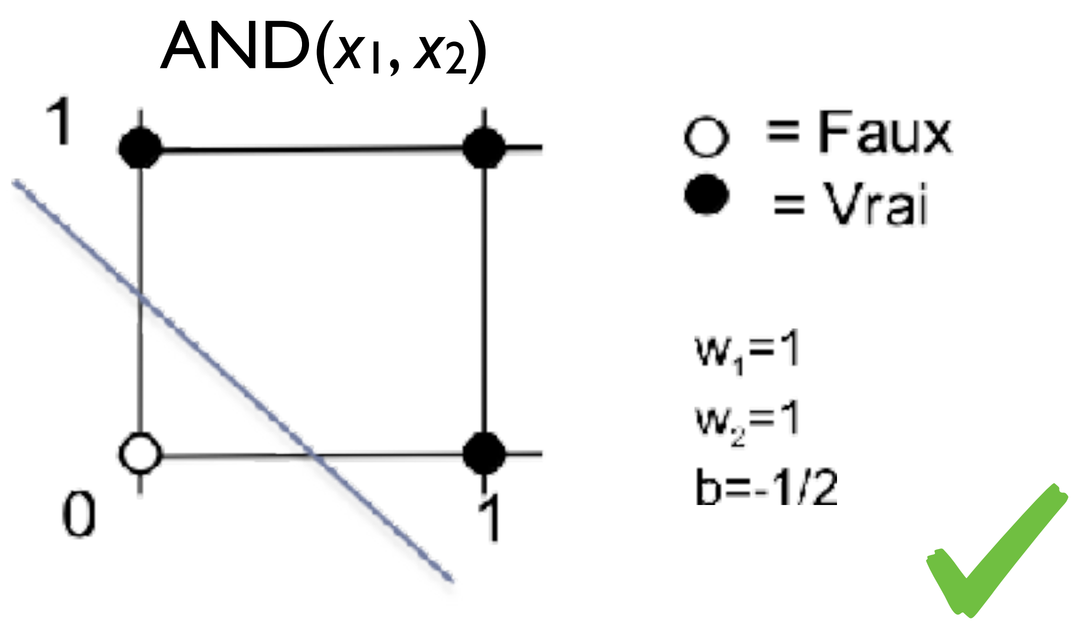
pas linéairement séparable

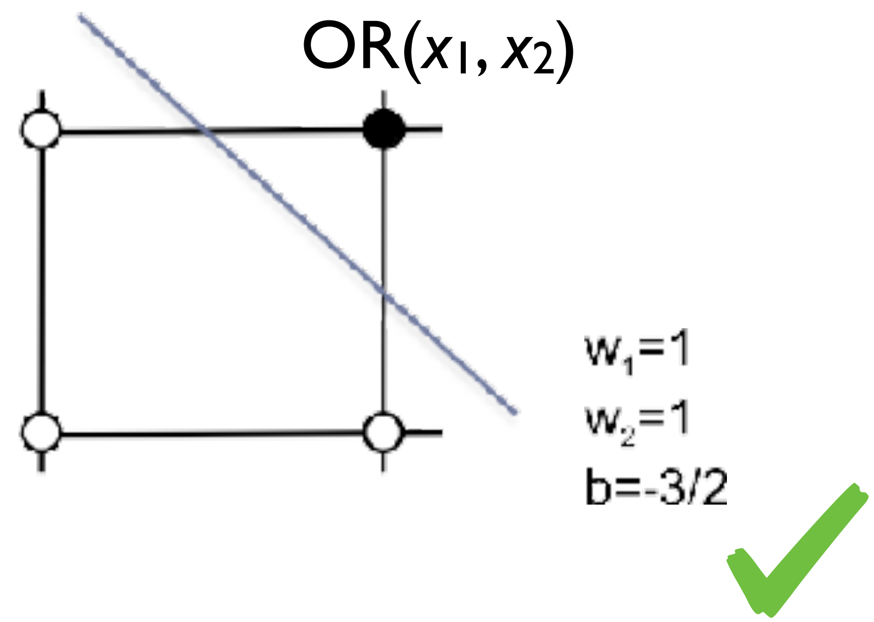
À remarquer qu'à très haute dimension ($N \gg 1$), des entrées sont généralement plus facilement séparables.

Exemples : AND, OR, et XOR

Le cas binaire : x_1, x_2 = 0 ou 1

- Pour des entrées binaires x_1, x_2, est-ce que le perceptron peut apprendre les fonctions logiques AND (et), OR (ou) et XOR (ou exclusif) ?



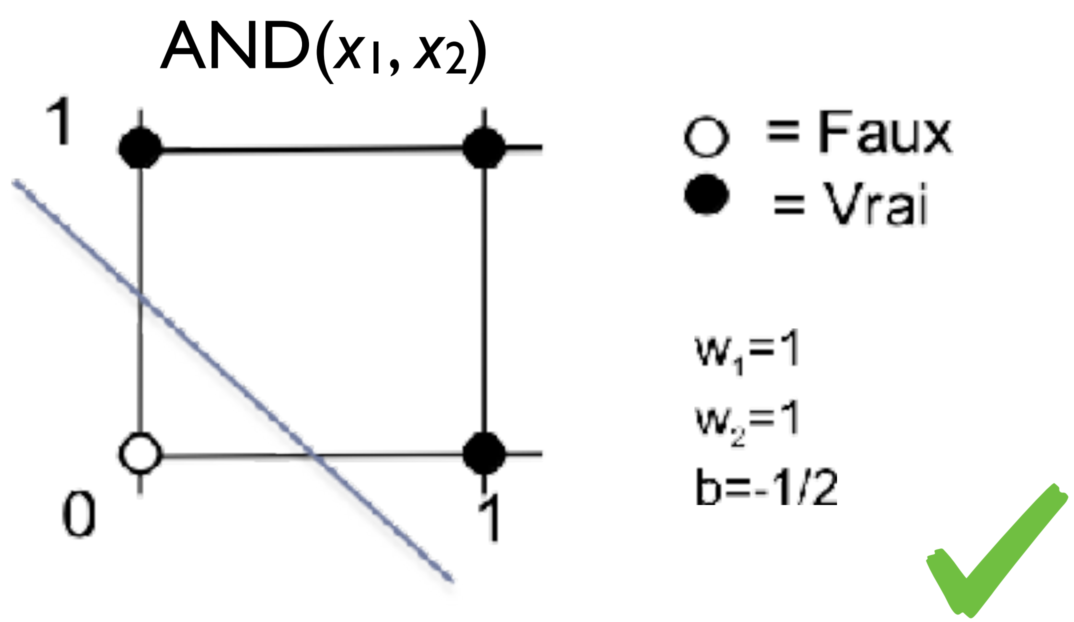


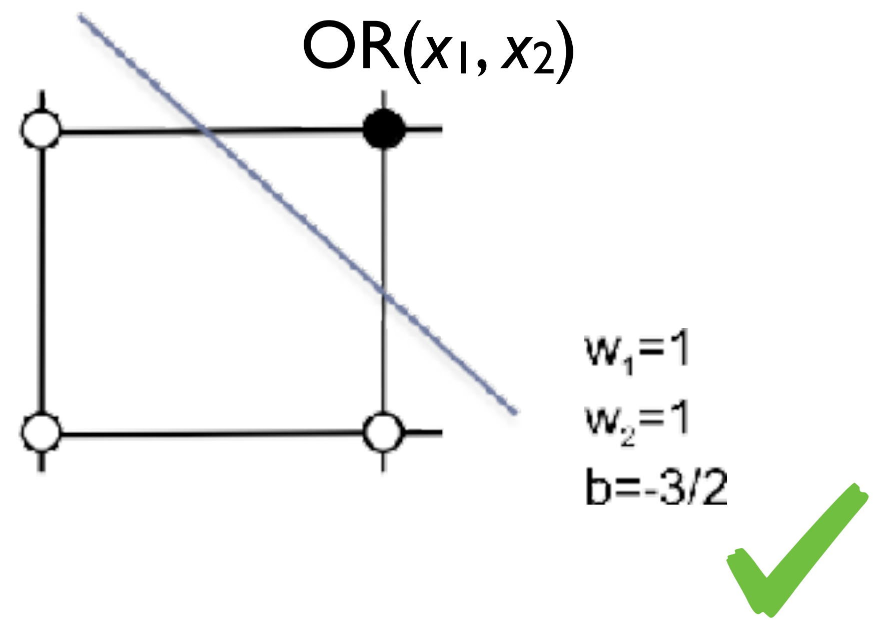
Un perceptron <u>peut exécuter</u> un *et* et un *ou*.

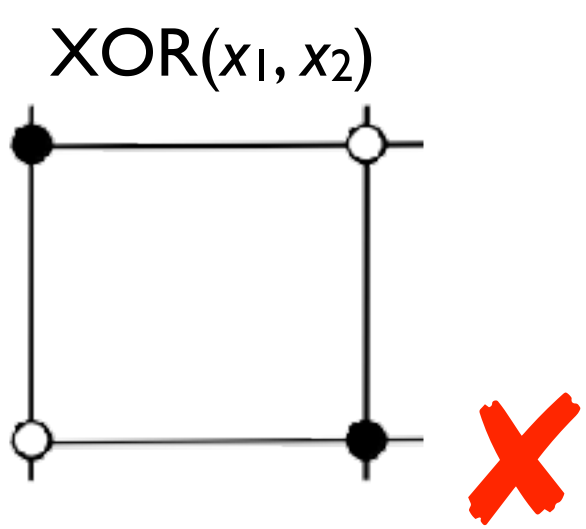
Exemples : AND, OR, et XOR

Le cas binaire : x_1, x_2 = 0 ou 1

- Pour des entrées binaires x_1, x_2, est-ce que le perceptron peut apprendre les fonctions logiques AND (et), OR (ou) et XOR (ou exclusif) ?







Un perceptron <u>peut exécuter</u> un *et* et un *ou*.

Un perceptron <u>ne peut pas exécuter</u> un *ou exclusif*.

Algorithme d'apprentissage du perceptron

1. **Initialiser les poids et le seuil :** choisir w_i, b aléatoirement dans l'intervalle [-1,1]

Algorithme d'apprentissage du perceptron

1. **Initialiser les poids et le seuil :** choisir w_i, b aléatoirement dans l'intervalle [-1,1]

2. **Sélectionner aléatoirement une entrée** k (x_i^k) aléatoirement parmi l'ensemble des K paires "entrée–sortie" d'entrainement (connaissant le résultat souhaité y^k). **Utiliser les poids et le seuil actuels pour faire la classification :**

$$\hat{y} = \operatorname{sign}\left(\sum_{j=1}^{N} w_j x_j^k + b\right)$$

Algorithme d'apprentissage du perceptron

1. **Initialiser les poids et le seuil :** choisir w_i, b aléatoirement dans l'intervalle [-1,1]

2. **Sélectionner aléatoirement une entrée** k (x_i^k) aléatoirement parmi l'ensemble des K paires "entrée–sortie" d'entrainement (connaissant le résultat souhaité y^k). **Utiliser les poids et le seuil actuels pour faire la classification :**

$$\hat{y} = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j^k + b\right)$$

3. **Si la classification est correcte** ($\hat{y} = y^k$), continuer avec 2.

Algorithme d'apprentissage du perceptron

1. **Initialiser les poids et le seuil :** choisir w_i, b aléatoirement dans l'intervalle [-1,1]

2. **Sélectionner aléatoirement une entrée** k (x_i^k) aléatoirement parmi l'ensemble des K paires "entrée–sortie" d'entrainement (connaissant le résultat souhaité y^k). **Utiliser les poids et le seuil actuels pour faire la classification :**

$$\hat{y} = \mathrm{sign}\left(\sum_{j=1}^{N} w_j x_j^k + b\right)$$

3. **Si la classification est correcte** ($\hat{y} = y^k$), continuer avec 2.

4. **Si la classification est fausse, mettre à jour les poids et le seuil** selon la règle suivante :

$$b \to b + \eta y^k$$

$$w_i \to w_i + \eta y^k x_i^k$$

$\eta > 0$ … taux d'apprentissage

Théorème de convergence

Est-ce qu'on peut être sûr que le perceptron va bien apprendre la classification ?

<u>Théorème</u> (prouvé) :
P*our tout ensemble de données linéairement séparable*, la règle d'apprentissage du perceptron ***est garantie de trouver une solution*** dans un nombre fini d'itérations.

<u>Exemple de mise-à-jours consécutifs d'un perceptron</u>