

# INF8175 - Intelligence artificielle

*Méthodes et algorithmes*

## Module 6: Apprentissage supervisé



POLYTECHNIQUE  
MONTRÉAL

Quentin Cappart

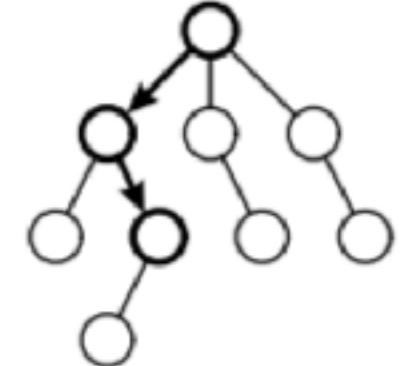
# Contenu du cours

## Raisonnement par recherche (essais-erreurs avec de l'intuition)

**Module 1:** Stratégies de recherche

**Module 2:** Recherche en présence d'adversaires

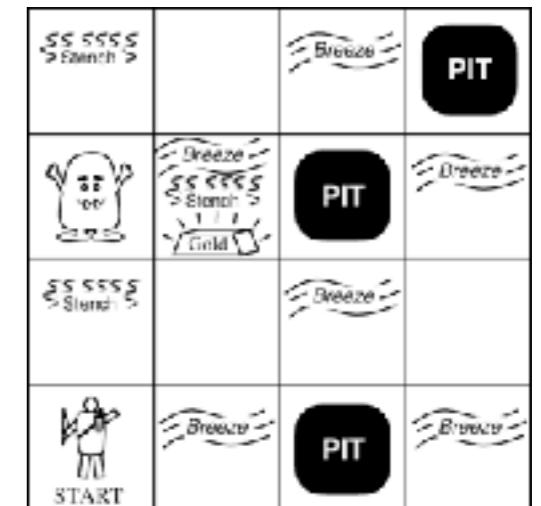
**Module 3:** Recherche locale



## Raisonnement logique

**Module 4:** Programmation par contraintes

**Module 5:** Agents logiques



## Raisonnement par apprentissage

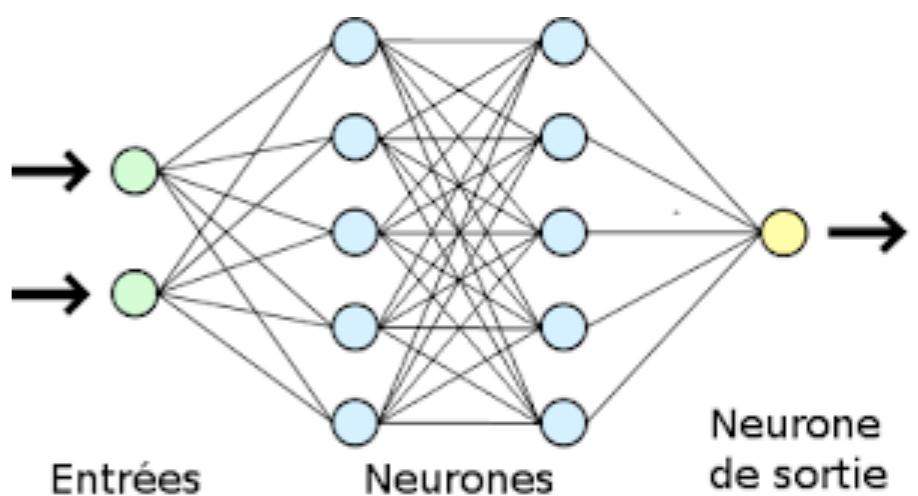
**Module 6:** Apprentissage supervisé



**Module 7:** Réseaux de neurones et apprentissage profond

**Module 8:** Apprentissage non-supervisé

**Module 9:** Apprentissage par renforcement



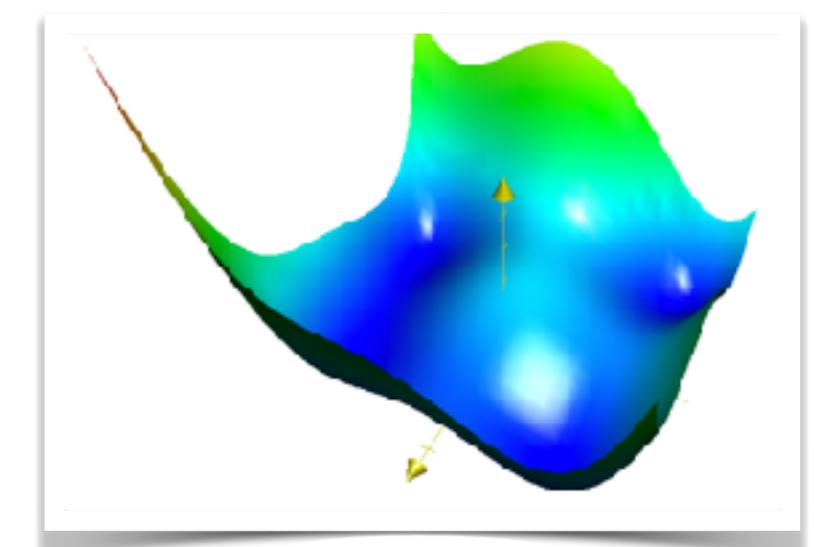
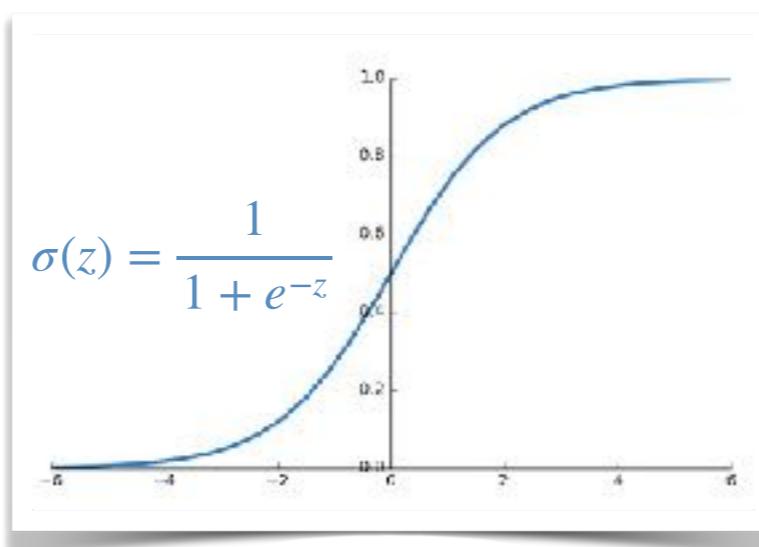
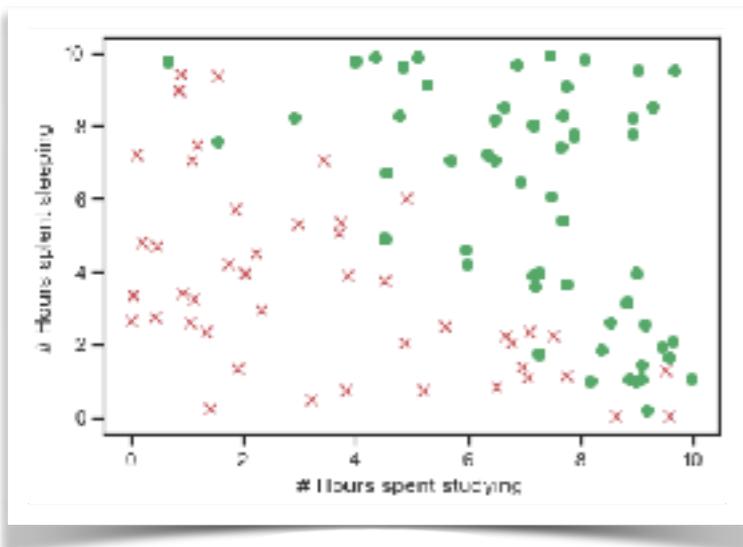
## Considérations pratiques et sociétales

**Module 10:** Utilisation en industrie, éthique, et philosophie

# Table des matières

## Apprentissage supervisé

1. Motivation et intérêt de l'apprentissage automatique
2. Classification des principaux types d'apprentissage
3. Définition de l'apprentissage supervisé
4. Méthode de la régression linéaire simple
5. Apprentissage par descente de gradient
6. Méthode de la régression linéaire multiple
7. Méthode de la régression logistique
8. Graphe de dépendance, *forward pass*, et *backward pass*



# Apprentissage automatique

Nous utilisons régulièrement des outils basés sur de l'apprentissage automatique dans notre vie quotidienne

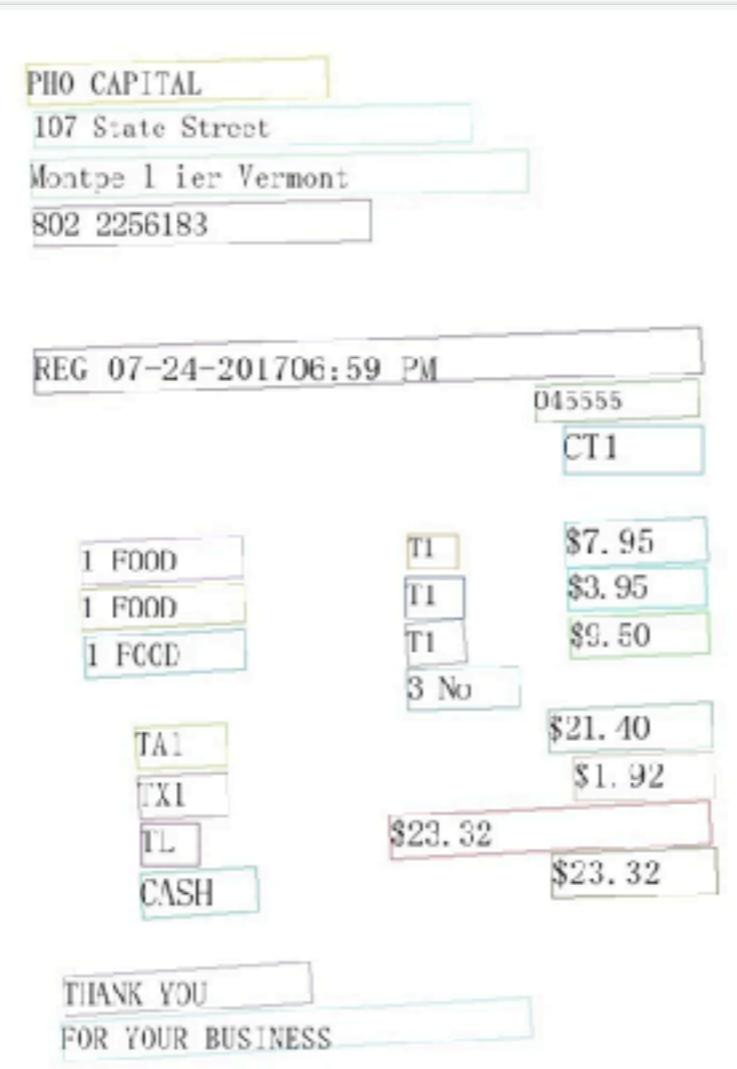
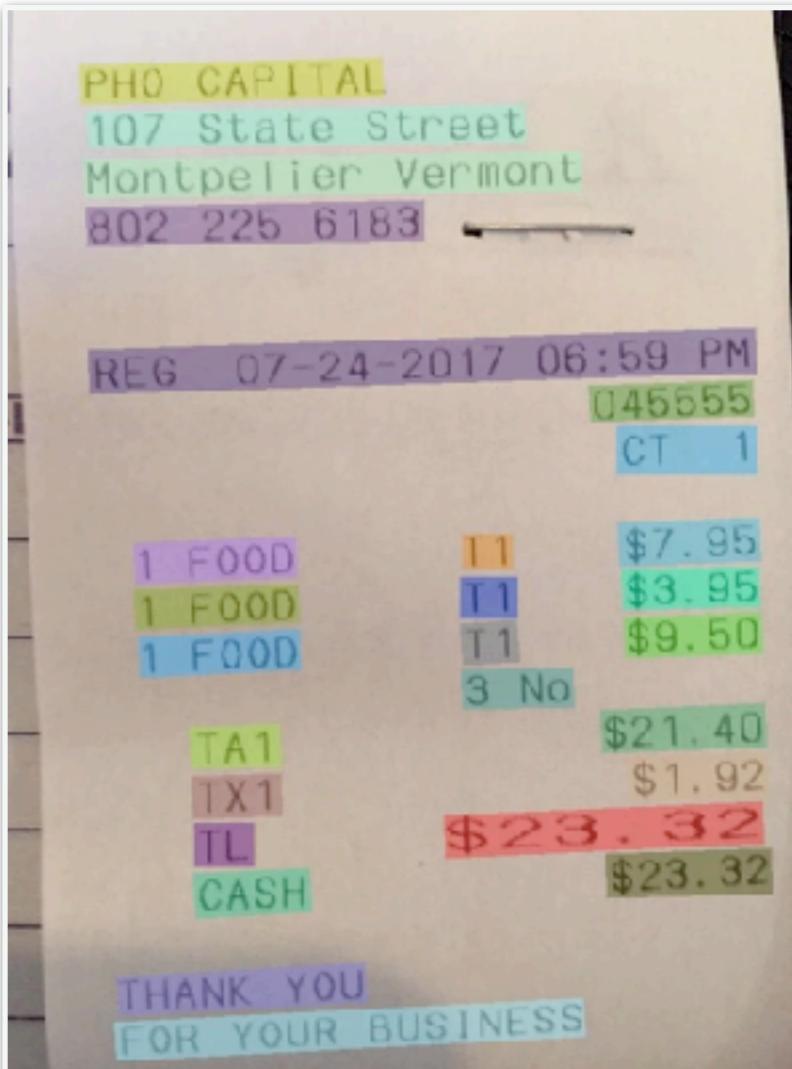


Pouvez-vous donner quelques exemples d'applications usuelles ?



**Application:** reconnaissance faciale pour vos appareils personnels

# Reconnaissance de caractères (OCR - *optical character recognition*)



**Application 1:** reconnaissance de caractères dans une facture

**Application 2:** reconnaissance de réponses dans un formulaire administratif

**Application 3:** dispatching automatique de lettres postales

Objectif simple, mais il est très difficile de construire manuellement un algorithme pour cette tâche

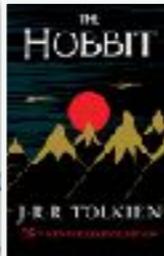
# Conduite autonome



# Traduction automatique du langage

DÉTECTOR LA LANGUE ANGLAIS FRANÇAIS ARABE

In a hole in the ground there lived a hobbit. Not a nasty, dirty, wet hole, filled with the ends of worms and an oozy smell, nor yet a dry, bare, sandy hole with nothing in it to sit down on or to eat: it was a hobbit-hole, and that means comfort.

247/51 

2022

ANGLAIS FRANÇAIS ARABE

Dans un trou dans le sol vivait un hobbit. Pas un trou sale, sale, humide, rempli de bouts de vers et d'une odeur suintante, ni encore un trou sec, nu, sablonneux, sans rien dedans pour s'asseoir ou pour manger : c'était un trou de hobbit, et ça signifie confort

Speaker icon, Print icon, Copy icon, Share icon

FRANÇAIS ANGLAIS ARABE

2020

Dans un trou dans le sol vivait un hobbit. Pas un trou désagréable, sale et humide, rempli d'extrémités de vers et d'une odeur suintante, ni encore un trou sec, nu et sablonneux sans rien pour s'asseoir ou pour manger: c'était un hobbit-trou, et que signifie confort.

Speaker icon, Print icon, Copy icon, Share icon

2023

Français Anglais Arabe

Dans un trou dans le sol vivait un hobbit. Pas un trou méchant, sale, humide, rempli de bouts de vers et d'une odeur suintante, ni encore un trou sec, nu, sablonneux, sans rien dedans pour s'asseoir ou manger : c'était un trou de hobbit, et cela signifie confort

Rechercher des détails

Speaker icon, Print icon, Copy icon, Share icon

**En 2020: la traduction n'est pas parfaite, mais donne néanmoins l'idée générale**

**En 2021, 2022 et 2023: la traduction s'est continuellement améliorée**

**Intérêt:** peut servir de base à de meilleures traductions

# Moteur de recherche



- Apprentissage auto
- apprentissage automatique
- apprentissage autonome
- apprentissage autorégulé
- apprentissage autonome adolescent
- apprentissage autorégulé définition
- apprentissage autonome ulaval
- apprentissage autodidacte
- apprentissage autonome robot
- apprentissage automobile
- apprentissage automatique mémoire

Recherche Google

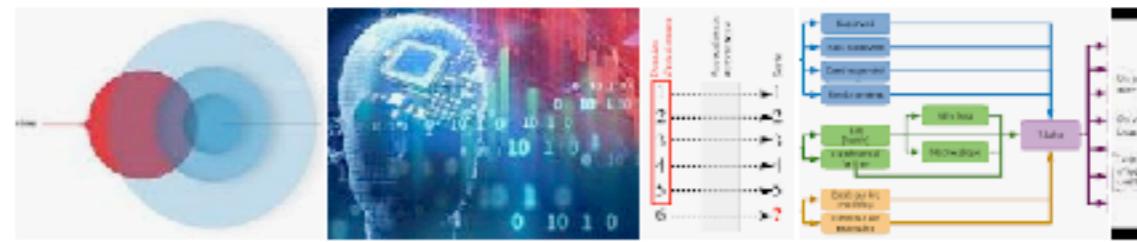
J'ai de la chance

Signaler des prédictions imprécises

apprentissage automatique

Tous Vidéos Images Actualités Shopping Plus Paramètres Outils

Environ 33 200 000 résultats (0,45 secondes)



« apprentissage machine »), apprentissage artificiel ou apprentissage statistique est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches mathématiques et statistiques pour donner aux ordinateurs la capacité d'« apprendre à partir de données, c'est-à-dire d'améliorer leurs performances à ...

fr.wikipedia.org › wiki › Apprentissage\_automatique

**Apprentissage automatique — Wikipédia**

À propos des extraits optimisés

Commentaires

www.oracle.com › ... › Solutions › Intelligence artificielle

**Qu'est-ce que l'apprentissage automatique? | Oracle Canada**

L'apprentissage automatique est un sous-domaine de l'intelligence artificielle (IA) qui se concentre sur la conception de systèmes qui apprennent — ou ...

admission.umontreal.ca › programmes › dess-en-appre...

**D.E.S.S. en apprentissage automatique - Université de Montréal**

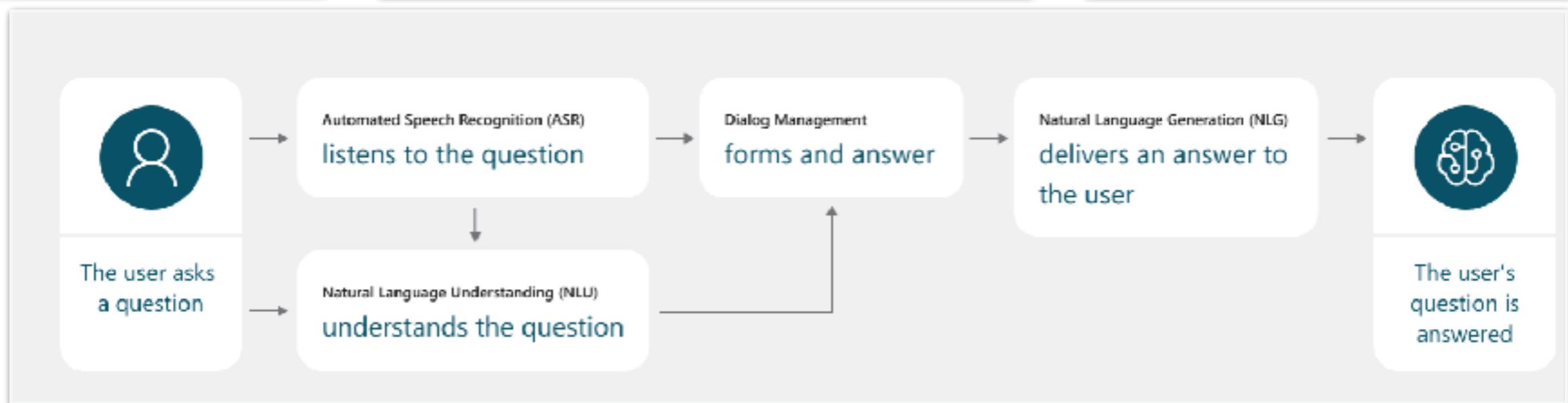
en apprentissage automatique fournit la formation minimale nécessaire pour appliquer les avancées en intelligence artificielle basées sur l'apprentissage ...

**Tâche 1:** l'auto-complétion de ma recherche

**Tâche 2:** donner les pages Web les plus probables

**Spécificité:** pour cela, on souhaite tenir compte du profil de la personne (p.ex., sa localisation)

# Chatbots vocaux



Tâche 1: reconstruction d'une phrase à partir d'une onde acoustique

Tâche 2: analyse de la requête

Tâche 3: réponse à donner ou prise de décision

Applications: support pour des clients, utilisation domestique, etc.

# Chatbots textuels

QU

Quelles sont les notions les plus importantes en programmation par contraintes ?

1. **Variable** : Les variables représentent les inconnues du problème. Chaque variable a un domaine de valeurs possibles.
2. **Domaine** : Le domaine d'une variable spécifie les valeurs possibles que cette variable peut prendre. Il peut s'agir d'un ensemble fini ou infini de valeurs.
3. **Contrainte** : Les contraintes sont des relations qui limitent les combinaisons de valeurs possibles pour les variables. Les contraintes décrivent les relations et les règles du problème.
4. **Satisfaction des contraintes** : Une solution est un ensemble de valeurs assignées aux variables qui respecte toutes les contraintes du problème. La satisfaction des contraintes est le processus de recherche de ces solutions.
5. **Recherche** : La recherche consiste à explorer l'espace des solutions possibles pour trouver une solution qui satisfait toutes les contraintes. Il existe de nombreuses techniques de recherche, y compris la recherche arborescente, la recherche locale, etc.
6. **Inférence** : L'inférence consiste à déduire de nouvelles contraintes à partir des contraintes existantes. Cela peut réduire l'espace des solutions possibles.
7. **Heuristiques** : Les heuristiques sont des techniques pour guider la recherche vers des solutions potentielles plus rapidement. Elles incluent le choix de la prochaine variable à assigner et la valeur à lui attribuer.
8. **Propagation de contraintes** : La propagation de contraintes est un processus automatique qui réduit le domaine des variables en fonction des contraintes existantes.

**Application 1:** support d'aide générale personnalisée

**Application 2:** support clientèle sur un site web

**Application 3:** générateur d'histoires à des fins récréatives



Data Privacy Bot

Please view our privacy policy below.

Yes, I agree

No, Not Now

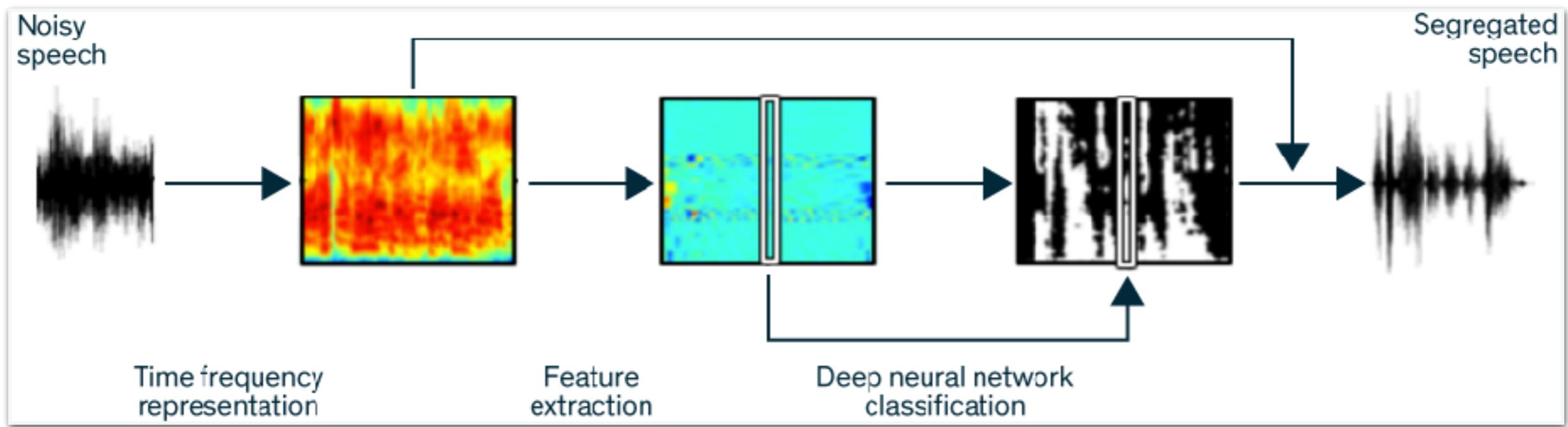
This chat may be monitored and the chat conversations may be retained, including for training purposes. See our Privacy Statement for details about how we process personal data.



# Application en robotique



# Applications dans le domaine médical



<https://spectrum.ieee.org/consumer-electronics/audiovideo/deep-learning-reinvents-the-hearing-aid>



**Application 1:** prothèse auditive pour mal-entendants

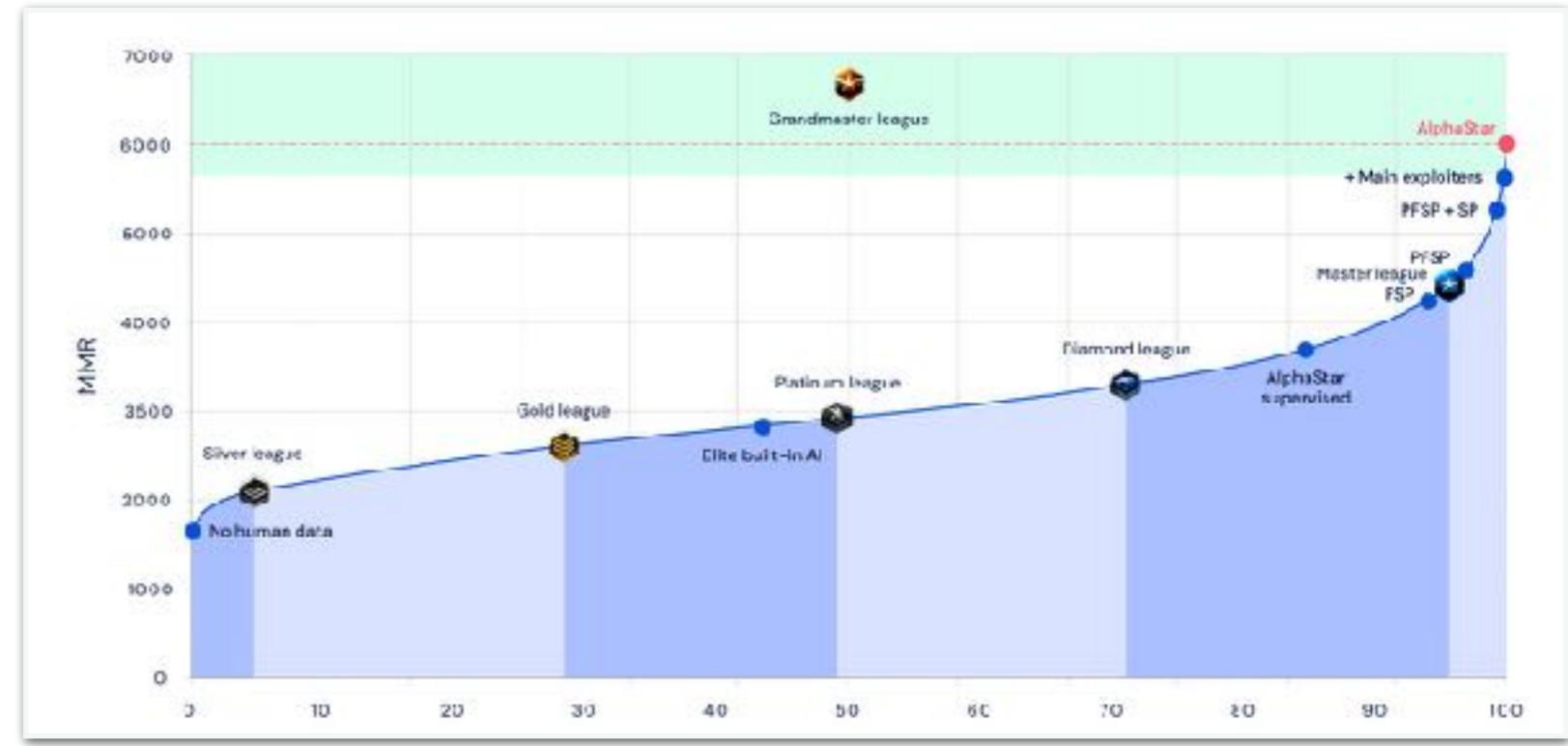
**Application 2:** amélioration du processus de traitement pour une radiothérapie

# Apprentissage pour les jeux



**Starting out - 10 minutes of training**

The algorithm tries to hit the ball back, but it is yet too clumsy to manage.



**Objectif:** utiliser les jeux comme environnement pour développer et tester des IAs (module 2)

Playing Atari with Deep Reinforcement Learning [DeepMind, 2013]

Mastering the game of Go with deep neural networks and tree search [DeepMind, 2016]

AlphaStar: Mastering the real-time strategy game StarCraft II [DeepMind, 2019]

# Générateur d'images - créations artistiques



# Générateur d'images - créations artistiques



## Générateurs d'images

**Objectif:** générer des images à partir d'une description textuelle

**Logiciels:** DALL-E, Midjourney, Stable diffusion, etc.

**Popularité:** génération d'images rigolotes, plausibles et liées à la requête

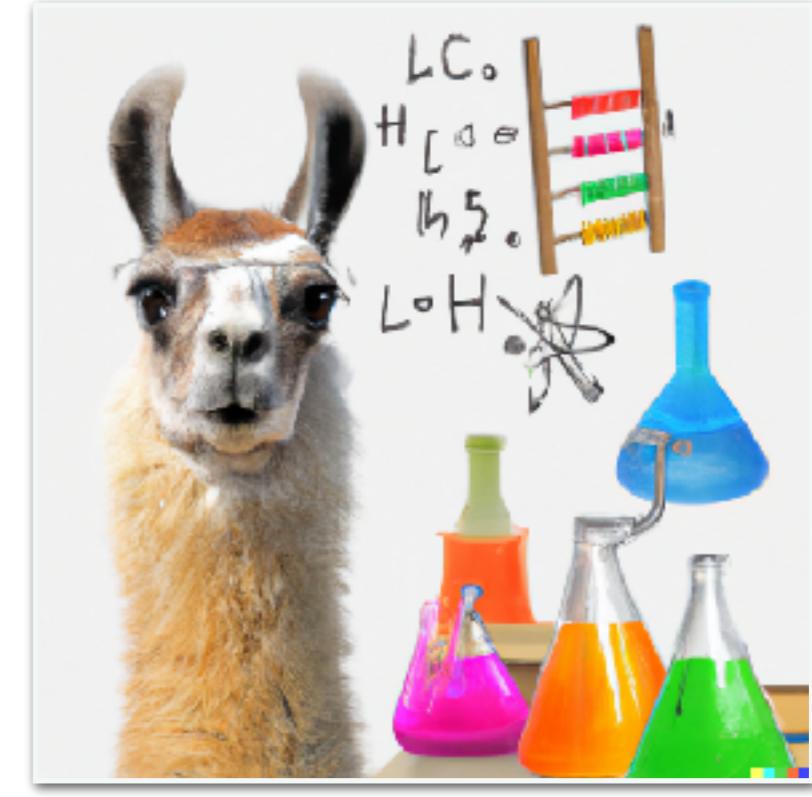
A university professor in the style of van Gogh



A university professor in the style of Picasso

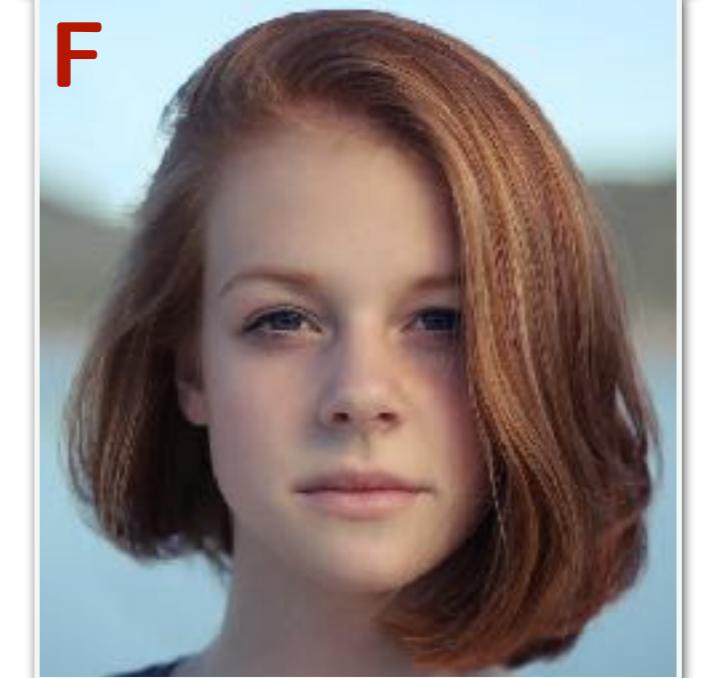


A llama following a course on chemistry



**Génération:** via la version payante de DALL-E

# Génération automatique

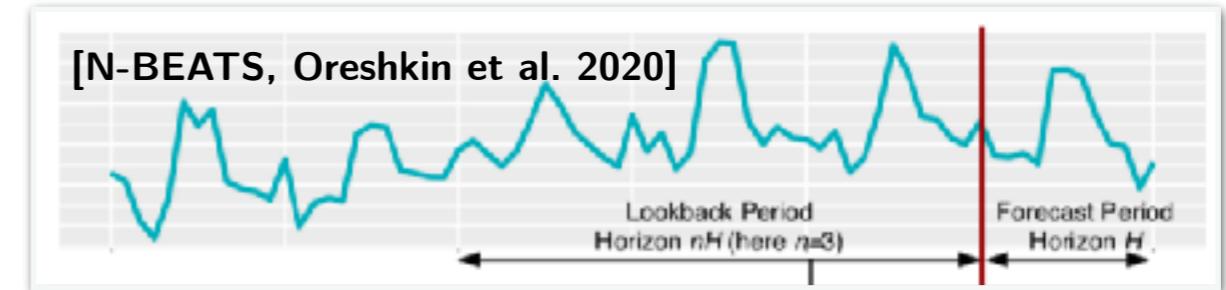
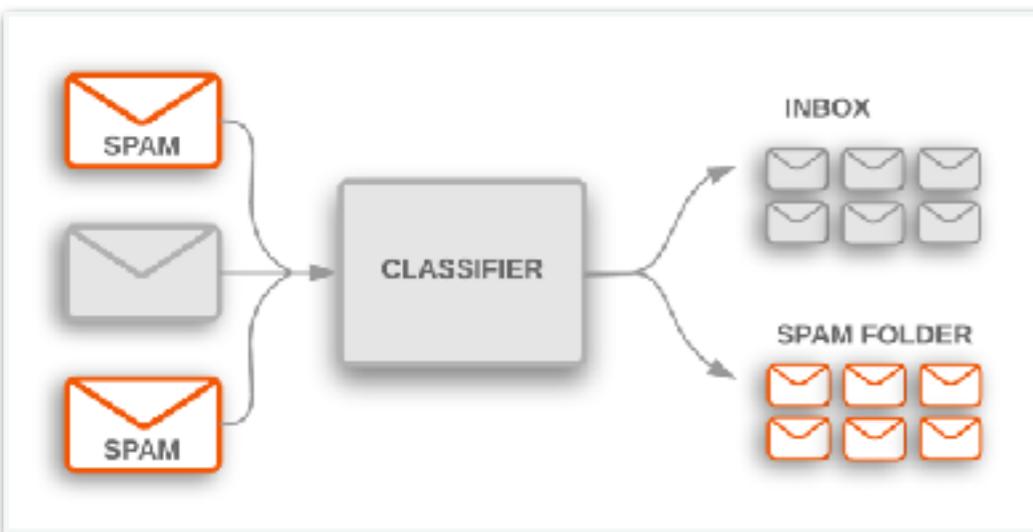


Une seule de ces photos provient d'une vraie personne, laquelle ?

Et je n'en suis pas sûr!

**Danger:** ouvre la porte à de nombreuses dérives éthiques (dernier module)

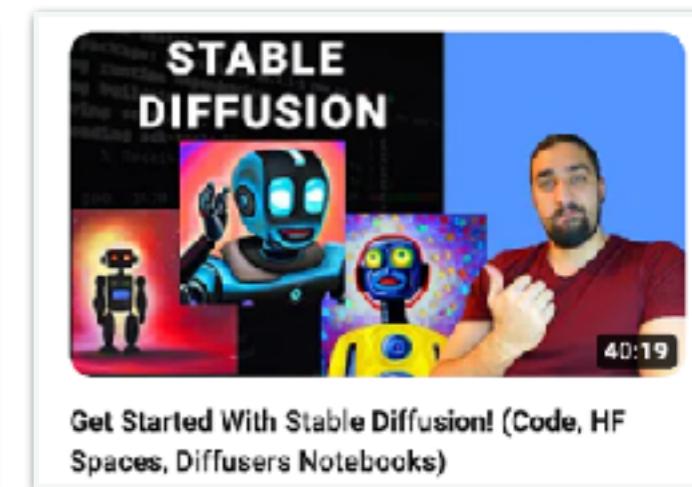
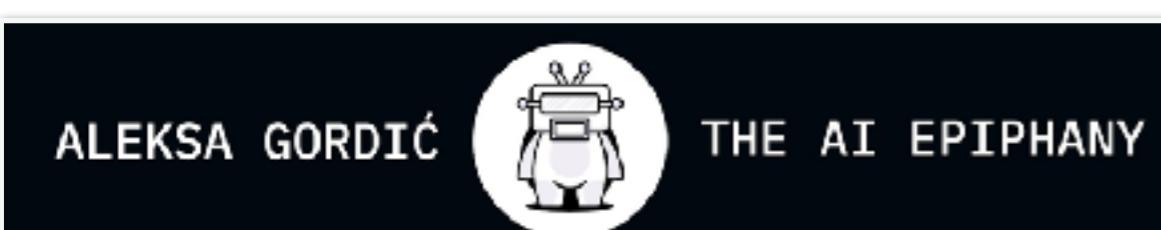
# Autres applications et ressources



Autre application: détection de spam

Autre application: prédition de ventes

Et encore un grand nombre d'applications, avec énormément d'exemples sur le Web



Objectif: rendre disponible différentes applications et leurs données relatives

Particularité: tâches présentées sous la forme de compétitions

Observation: de plus en plus d'entreprises passent par Kaggle pour la résolution de leurs problèmes

# Définition de l'apprentissage automatique

## Définition moderne de l'apprentissage automatique (parmi d'autres)



Tom Mitchell (1997)

*"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E"*

<http://www.cs.cmu.edu/afs/cs.cmu.edu/user/mitchell/ftp/mlbook.html>

Trois éléments fondamentaux apparaissent:

- (1) L'apprentissage est dédié pour une tâche précise (T)
- (2) L'apprentissage a besoin d'expériences passées (E - données)
- (3) Besoin d'avoir un moyen d'évaluer les performances d'un modèle (P)

## Confusion avec l'intelligence artificielle



Michael Jordan (2019)

*"Most of what is labeled AI today, particularly in the public sphere, is actually machine learning (ML), a term in use for the past several decades"*

<https://hdsr.mitpress.mit.edu/pub/wot7mkc1/release/8>

L'apprentissage automatique n'est qu'un sous-domaine de l'intelligence artificielle

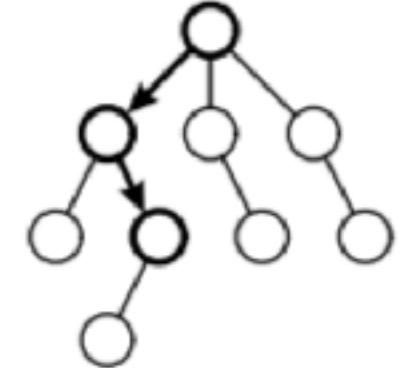
# Contenu du cours

## Raisonnement par recherche (essais-erreurs avec de l'intuition)

**Module 1:** Stratégies de recherche

**Module 2:** Recherche en présence d'adversaires

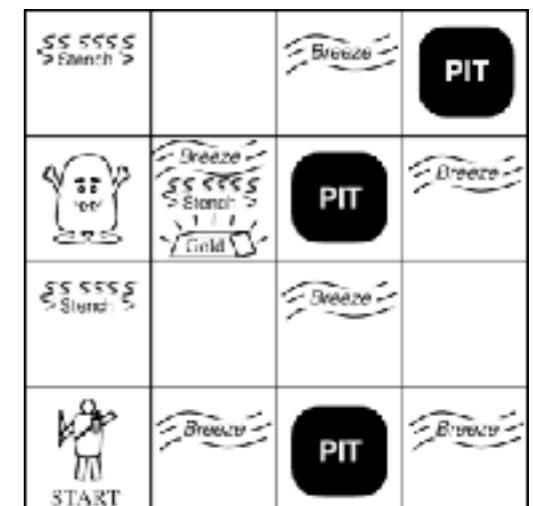
**Module 3:** Recherche locale



## Raisonnement logique

**Module 4:** Programmation par contraintes

**Module 5:** Agents logiques



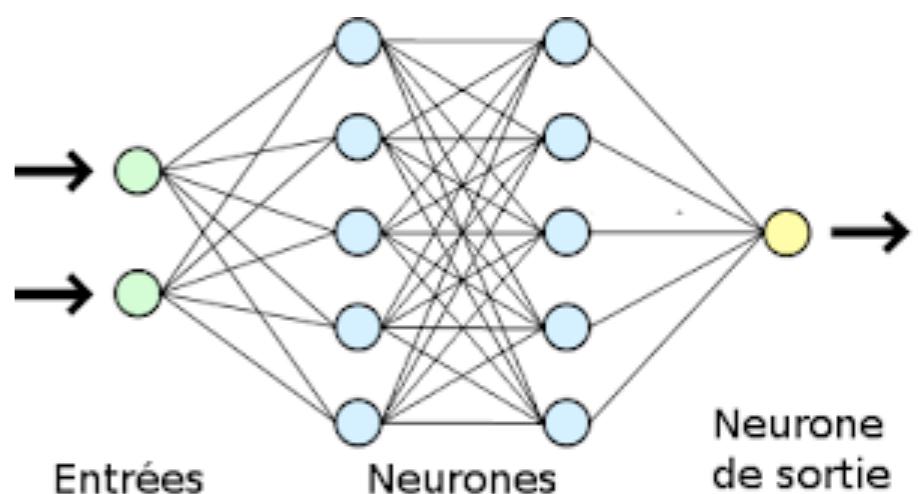
## Raisonnement par apprentissage

**Module 6:** Apprentissage supervisé

**Module 7:** Réseaux de neurones et apprentissage profond

**Module 8:** Apprentissage non-supervisé

**Module 9:** Apprentissage par renforcement



## Considérations pratiques et sociétales

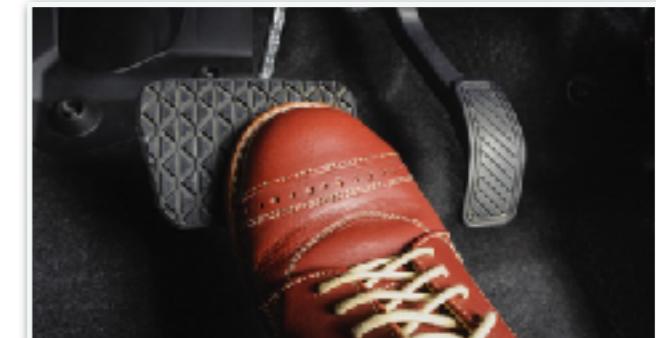
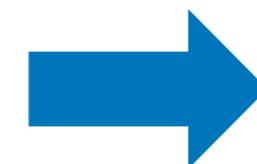
**Module 10:** Utilisation en industrie, éthique, et philosophie

# Objectif de l'apprentissage automatique

L'objectif est de construire une fonction capable d'effectuer une prédiction, pour une situation spécifique



Dashcam



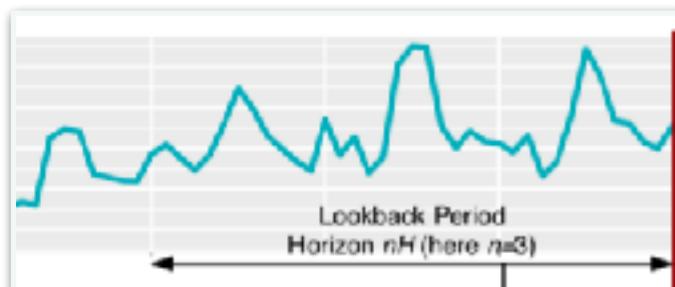
Freiner

*"This movie was far better than the trailer made it look."*

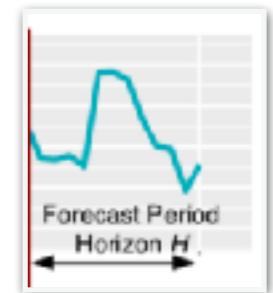
Review



Score



Ventes précédentes



Ventes futures

Notez la grande variété des situations pouvant être considérées (utile dans énormément de contextes)

# Intérêt de l'apprentissage automatique



Pourquoi ne pas résoudre ces problèmes avec les techniques déjà vues ?

Raison 1: certaines tâches sont très difficiles à résoudre avec les techniques précédentes



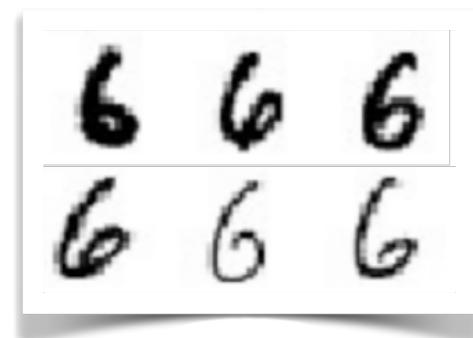
**Exemple:** effectuer une reconnaissance d'images

**Approche traditionnelle:** déceler manuellement des caractéristiques

**Apprentissage:** exploiter les similarités avec d'autres photos déjà vues

Cette deuxième option est plus facile (et efficace) à mettre en oeuvre

Raison 2: on souhaite avoir une approche applicable à des situations qui ne sont pas encore rencontrées



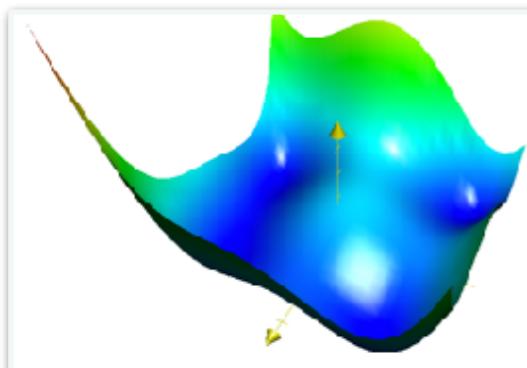
**Exemple:** effectuer une reconnaissance de chiffres écrits à la main



**Approche naïve:** mémoriser tous les chiffres vus dans une base de données

**Difficulté:** incapable de généraliser à des nouveaux chiffres, similaires, mais non-vus

Raison 3: en quelques sortes, on va utiliser des algorithmes déjà vus



**Exemple:** un clustering peut se faire avec de la recherche locale

**Réalité:** plusieurs méthodes d'apprentissage utilisent des algorithmes connus

**Conclusion:** la frontière entre les différents modules n'est pas si stricte...

# Principaux types d'apprentissage

## Apprentissage supervisé

Caractéristique: apprentissage qui utilise des données labellisées (la vraie valeur est connue - *ground truth*)



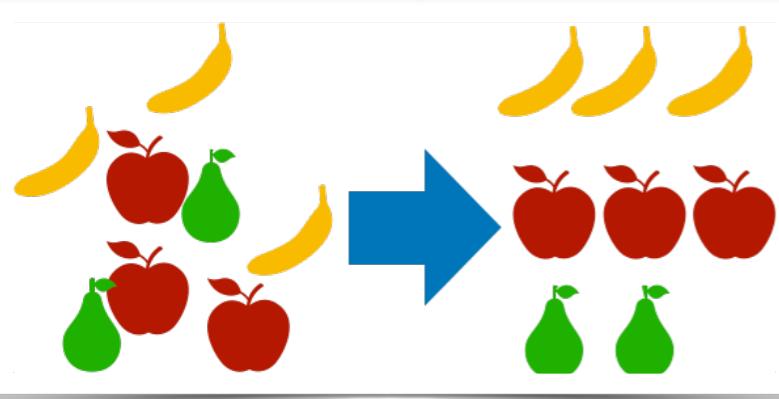
Analogie de l'apprentissage: un professeur qui enseigne des notions

Analogie de l'exécution: l'élève reproduit dans de nouveaux contextes

Difficulté: on doit avoir ces données à disposition en suffisance

## Apprentissage non-supervisé

Caractéristique: apprentissage qui n'utilise aucune donnée labellisée



Objectif: apprendre des associations de similarité entre les données

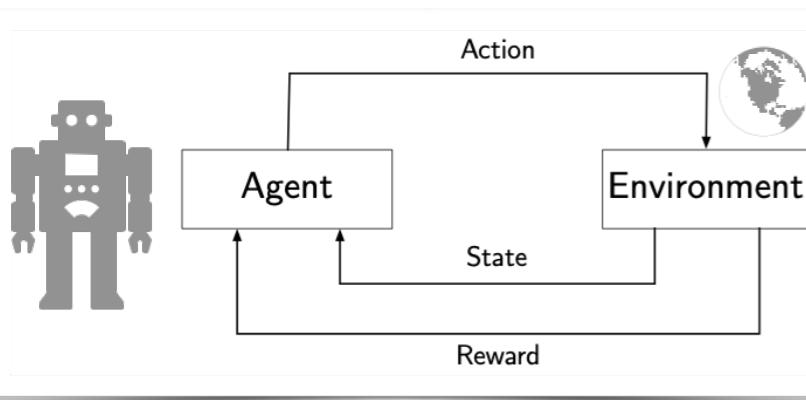
Exemple: détecter une anomalie (*malware* parmi des programmes sains)

Exemple: regrouper les données semblables (*clustering*)

Difficulté: on a pas de label de référence pour aider l'apprentissage

## Apprentissage par renforcement

Caractéristique: apprendre par expériences et essais-erreurs via un signal de récompense



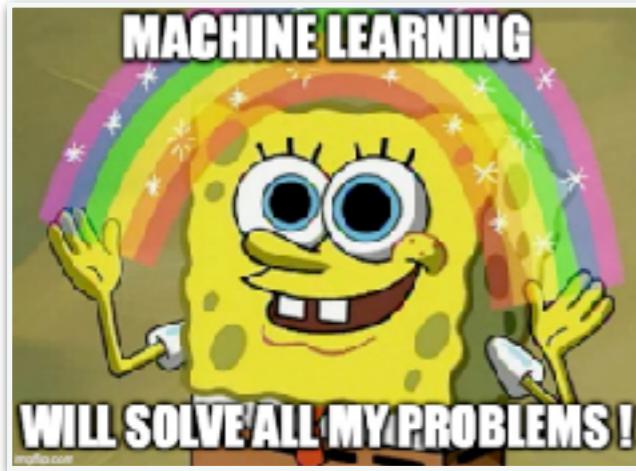
Fonctionnement: l'agent réalise toute une série d'actions

Signal de récompense: positif en cas de bonne action, négatif sinon

Objectif: obtenir le plus de récompenses lors de ces actions

Difficulté: important de savoir comment récompenser les actions faites

# Limitations de l'apprentissage automatique



**Limitations:** certaines tâches ne sont pas propices à l'apprentissage automatique



**Critères à considérer:** efficacité, éthique, données disponibles, etc.

## (1) Tâches très bien résolues avec des algorithmes traditionnels



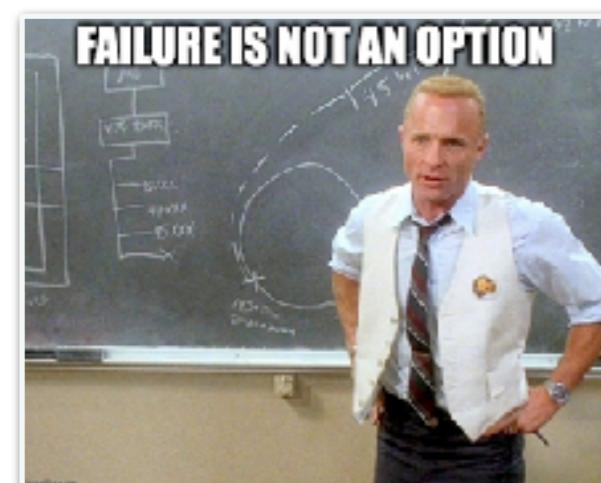
**Exemple:** trier une liste

**Exemple:** trouver le plus court chemin entre deux noeuds

**Exemple:** résoudre efficacement un problème combinatoire

**Approches liées:** *neural reasoning, neural combinatorial optimization*

## (2) Tâches nécessitant des garanties



**Difficulté:** l'apprentissage automatique ne réalise que des prédictions

**Conséquence:** il y a un risque que les prédictions soient erronées

**Non-conseillé pour des systèmes critiques où une erreur est coûteuse à réparer**

**Approches liées:** *machine learning with guarantees, safe machine learning*

# Limitations de l'apprentissage automatique

## (3) Tâches trop coûteuse pour utiliser un modèle



**Difficulté:** entraîner un modèle peut demander des ressources conséquentes

**Ressources:** temps d'exécution, temps d'entraînement, mémoire spatiale, etc.

**Exemple:** systèmes temps-réels

**Approches liées:** *deep learning on FPGA*

## (4) Tâches avec peu ou aucune donnée disponible



**Réalité:** entraîner un modèle demande des données (idéalement labellisées)

**Difficulté:** ces données ne sont pas toujours disponibles

**Exemple:** prédiction d'une maladie rare

**Approches liées:** *few-shot learning, zero-shot learning, détection d'anomalies*

## (5) Situations éthiques délicates



**Difficulté:** on doit faire attention à ce genre de situation et empêcher la discrimination

**Exemple:** prédiction si un étudiant doit être admis à une université ou non

**Exemple:** prédiction si une personne a droit à un prêt hypothécaire

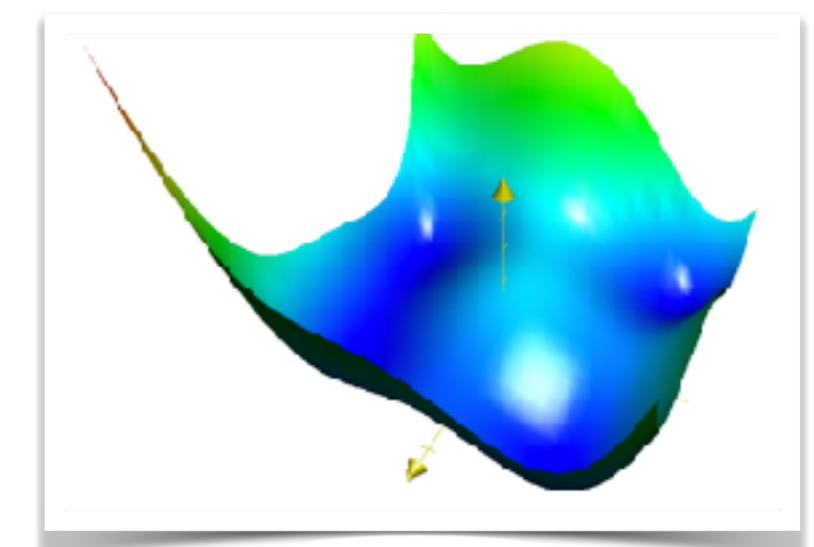
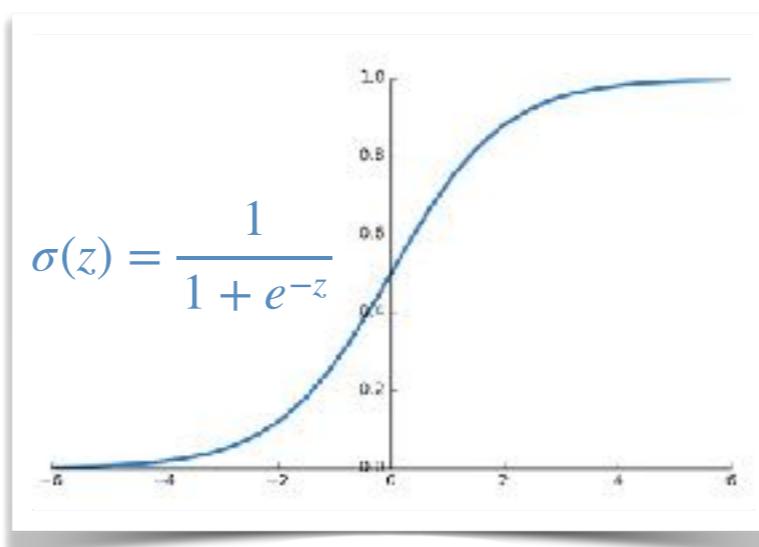
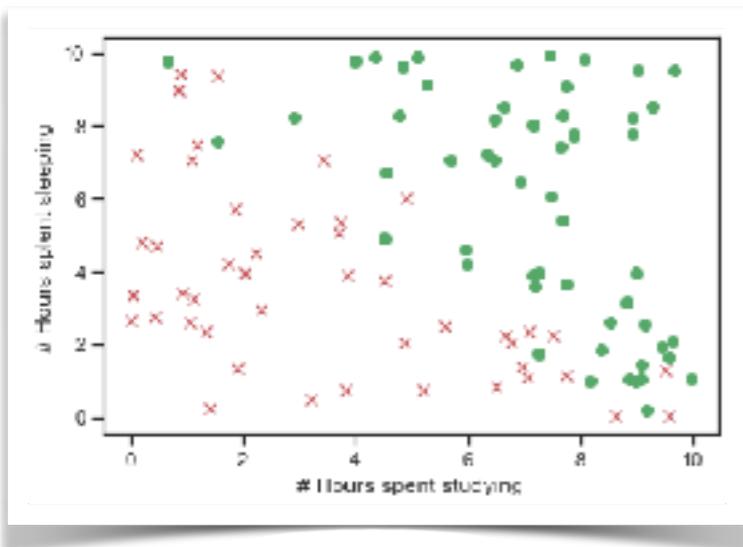
**Approches liées:** *Fairness-aware AI, Trustworthy machine learning*

**Pour un grand nombre de situations, l'apprentissage automatique n'est pas l'approche à privilégier**

# Table des matières

## Apprentissage supervisé

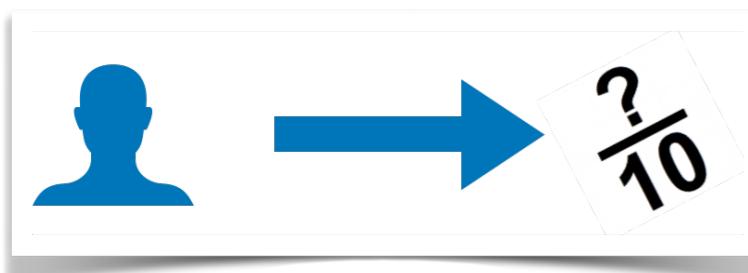
- ✓ 1. Motivation et intérêt de l'apprentissage automatique
- ✓ 2. Classification des principaux types d'apprentissage
- 3. Définition de l'apprentissage supervisé
- 4. Méthode de la régression linéaire simple
- 5. Apprentissage par descente de gradient
- 6. Méthode de la régression linéaire multiple
- 7. Méthode de la régression logistique
- 8. Graphe de dépendance, *forward pass*, et *backward pass*



# Mise en situation: le problème de l'étudiant

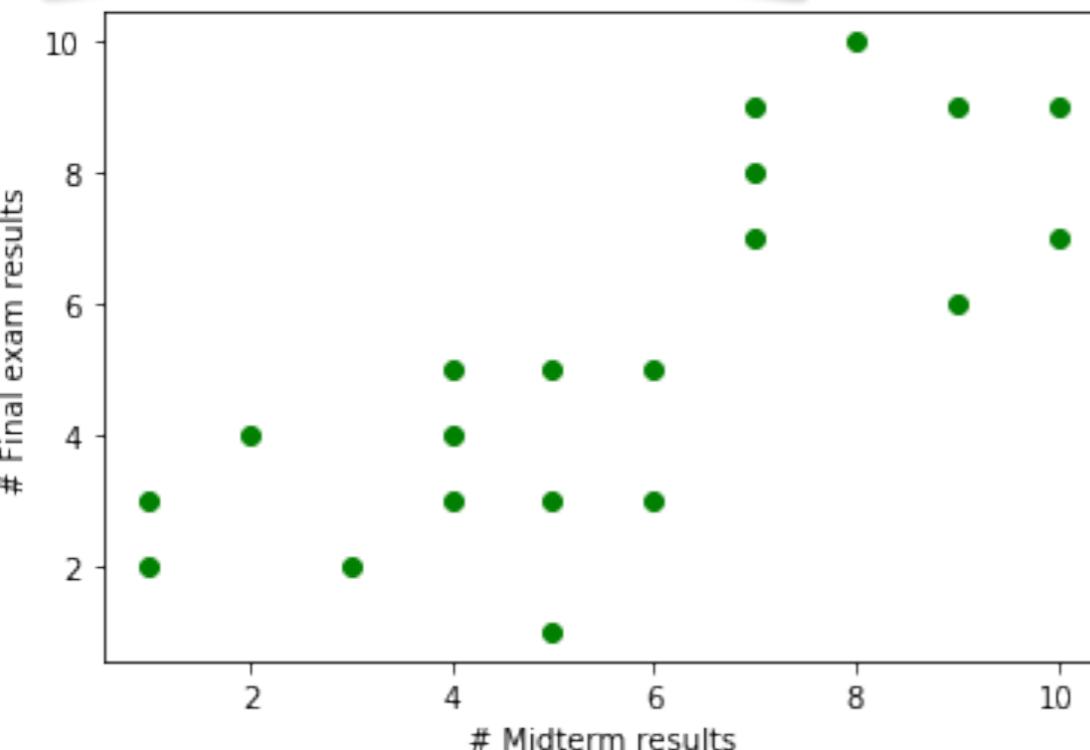
## Problème de l'étudiant

**Mise en situation:** suite à ces notes de l'intra, un étudiant se demande combien il obtiendra à son examen



**Son idée:** il se base sur les résultats des étudiants de l'année précédente

**Première étape:** construire une formalisation du problème



(1) Un étudiant est caractérisé par sa note à l'intra

$$x \in [0..10]$$

(2) Le résultat est la note à l'examen final

$$y \in [0..10]$$

(3) On souhaite construire une fonction de prédiction

$$f: [0..10] \rightarrow [0..10]$$



## Problème de régression

Problème visant à construire une fonction de prédiction renvoyant une valeur continue

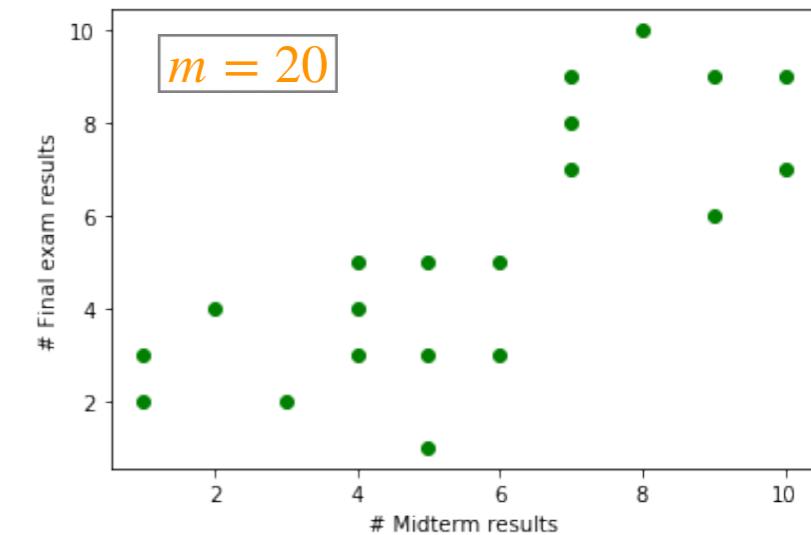
$$f: X \rightarrow \mathbb{R}$$

**Note:** on peut également restreindre la prédiction à un intervalle de valeurs continues

# Formalisation du problème

## Principes de l'apprentissage supervisé

- (1) Présence de **données historiques**, dont on connaît **la vraie valeur**
- (2) Les données sont utilisées pour construire **une fonction de prédiction**
- (3) Utilisation de la fonction pour prédire **la valeur de nouvelles situations**



## Notations et vocabulaire général

**Ensemble d'entraînement (training set):** les données utilisées pour construire la fonction

$m$  : taille de l'ensemble

$$D : \left\{ (x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \dots, (x^{(m)}, y^{(m)}) \right\}$$

**Donnée (data):** une association entre en ensemble de caractéristiques (features) et un label

$(x^{(i)}, y^{(i)})$ , avec  $x \in [0..10]$  et  $y \in [0..10]$

$x^{(i)}$  : caractéristique de la donnée  $i$  (on vera par la suite qu'on peut avoir plusieurs caractéristiques)

$y^{(i)}$  : label de la donnée  $i$  (la vraie valeur, ou encore l'étiquette)

**Fonction de prédiction:** fonction donnant une valeur à partir des caractéristiques en entrée

$f : [0..10] \rightarrow [0..10]$

**Valeur prédictive:** valeur de sortie de la fonction pour de nouvelles données

$\hat{y} = f(x)$  (le chapeau indique qu'on a une valeur estimée)

# Fonction de prédiction



Notre objectif est de construire une fonction de prédiction.

Quelle forme de fonction utiliser ? (linéaire, polynomiale, exponentielle, etc.)

**Hypothèse:** concept fondamental en apprentissage automatique

**Principe:** préconception que l'on fait sur notre fonction

**Votre tâche:** vous devez définir l'hypothèse qui vous semble pertinente

**Exemple:** fonction linéaire



**Approximation par une fonction linéaire**

Fonction de prédiction ayant la forme suivante:

$$\hat{y} = f(x) = w \cdot x + b$$

$x$  : caractéristique (feature)

$w, b$  : paramètres de la fonction (parameters)

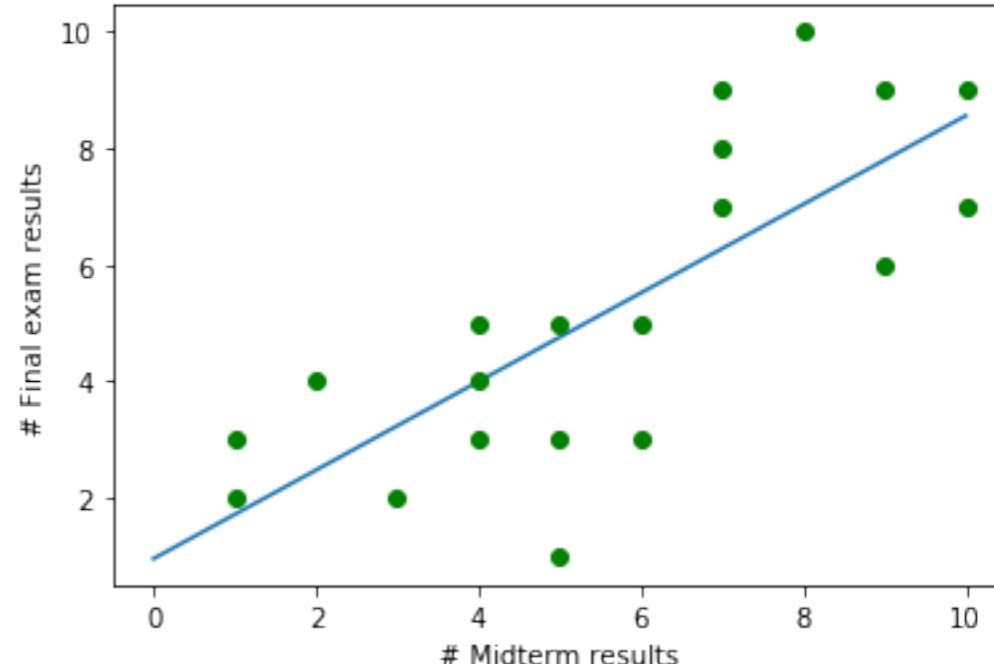
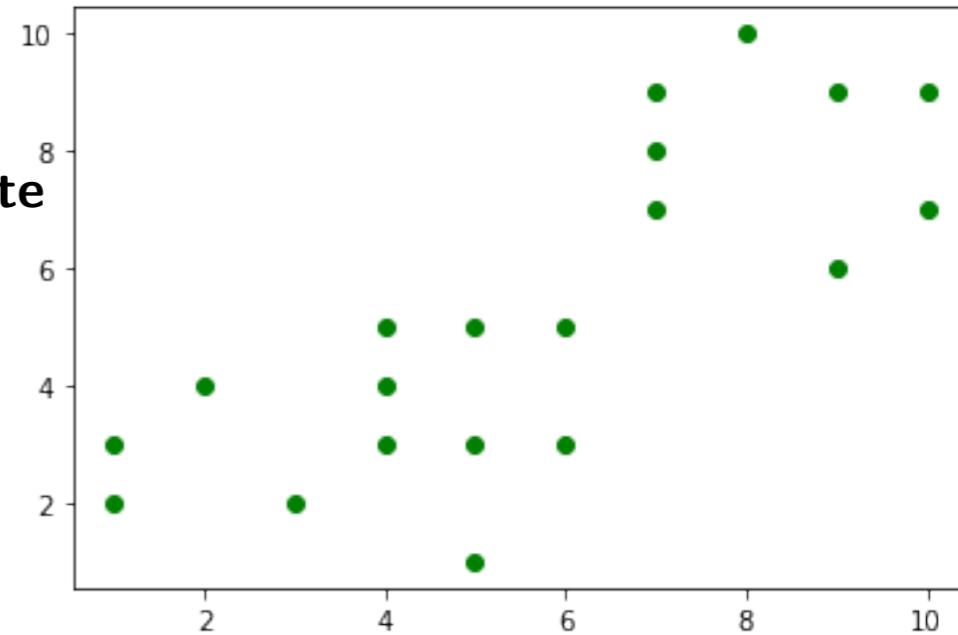
$w$  : poids de la caractéristique  $x$  (weight)

$b$  : biais (bias or intercept)

**Note:** hypothèse très simple mais néanmoins très utile

**Paramètres:** définissent le type de la droite

**Objectif:** trouver des valeurs adéquates pour ces paramètres



# Régression linéaire simple

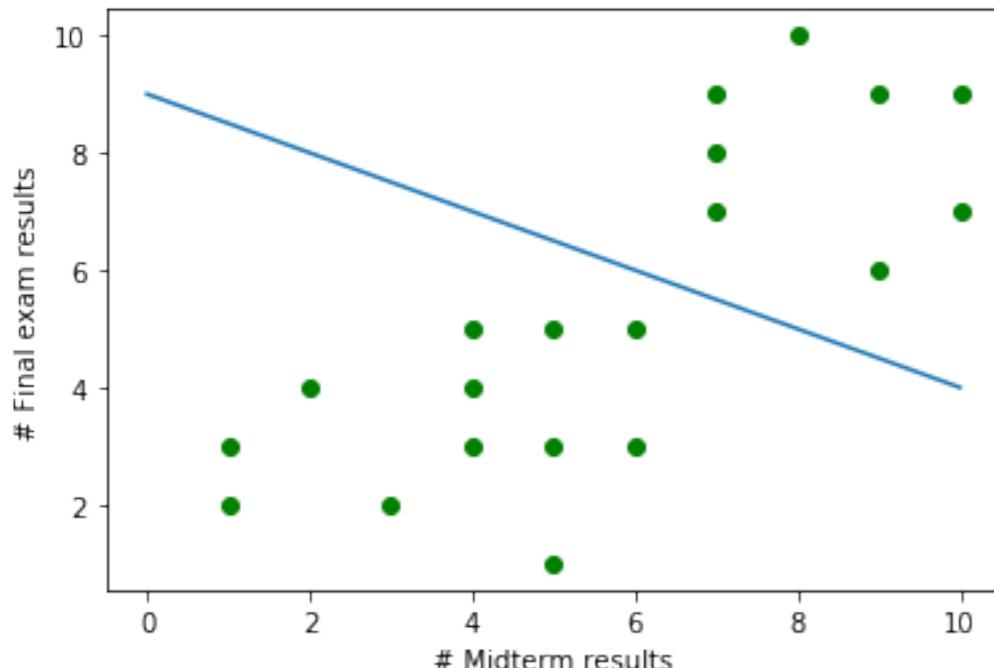


Comment déterminer ces paramètres ( $w$  et  $b$ ) ?

**Rappel:** cela revient à déterminer la droite (pente et *intercept*)

**Intuition:** on souhaite obtenir la droite qui donne une bonne approximation de nos données

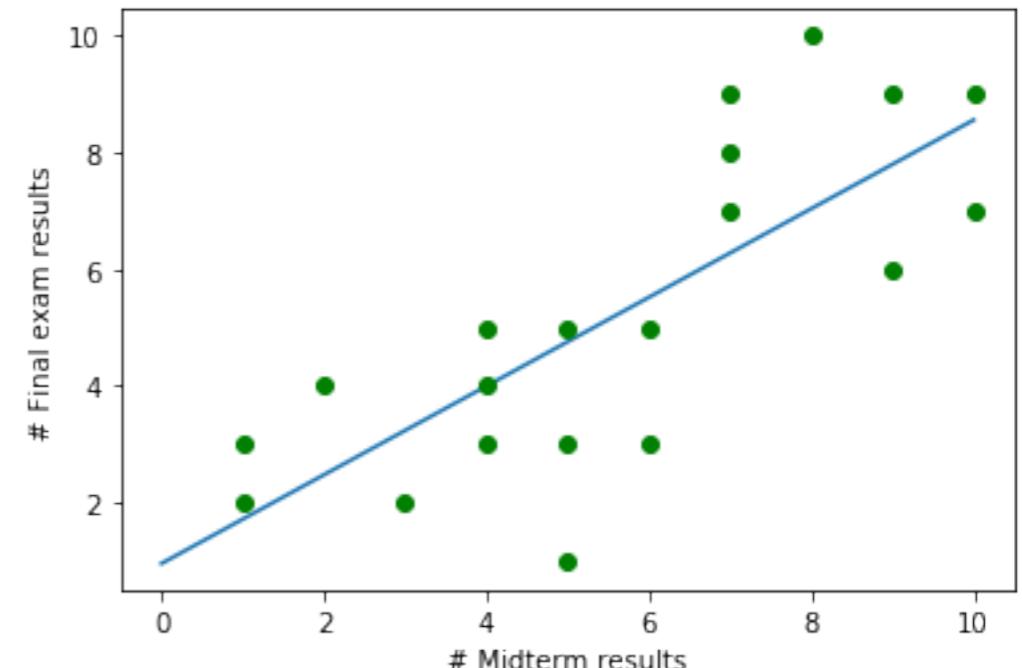
$$f(x) = -0.5x + 9$$



**Approximation linéaire**

$$\hat{y} = f(x) = w \cdot x + b$$

$$f(x) = 0.76x + 0.96$$



**Observation:** la fonction de droite semble être de meilleure qualité

**Régression linéaire**

Tâche consistant à déterminer la meilleure approximation linéaire entre une variable ( $x$ ) et son label ( $y$ )

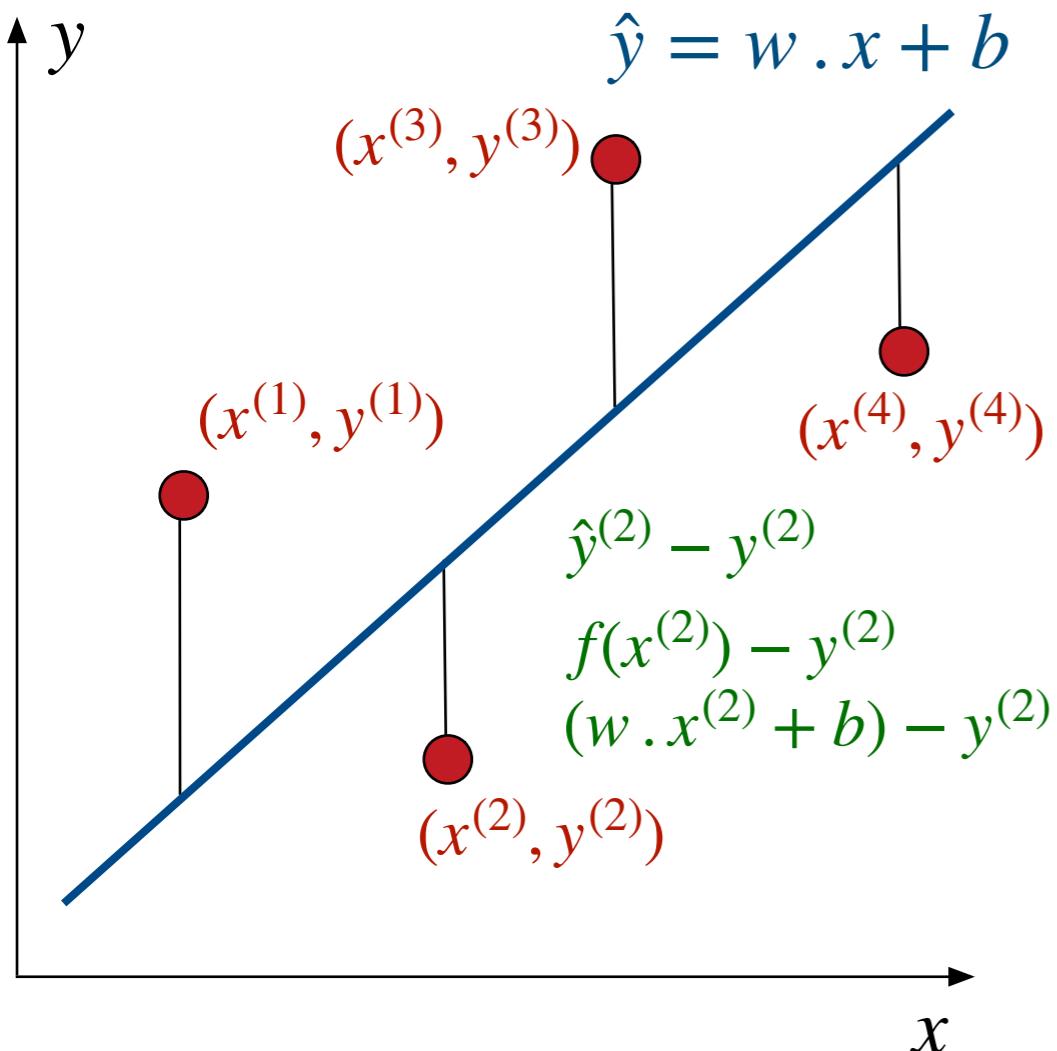
# Fonction d'écart



Comment formaliser la qualité d'une fonction par rapport à nos données ?

**Intuition:** on veut construire la fonction (linéaire) qui approxime le mieux nos données

**Métrique possible:** considérer l'écart entre la fonction et les points connus



**Difficulté:** la notion d'écart n'est pas bien définie

En pratique, il existe plusieurs façons de calculer cet écart



**Fonction d'écart (*loss function*)**  
Fonction définissant l'écart qu'un modèle de prédiction a avec un certain point de l'espace

$$L(\hat{y}^{(i)}, y^{(i)}) \rightarrow \text{écart}$$

Exemple: l'écart quadratique  $\rightarrow L(\hat{y}^{(i)}, y^{(i)}) = (\hat{y}^{(i)} - y^{(i)})^2$

**Intérêt:** évite que les écarts positifs et négatifs se compensent

**Intérêt:** donne une forte pénalité pour les grands écarts

**Intérêt:** cet écart est une fonction différentiable

Un objectif possible est de minimiser la somme de tous ces écarts

# Formalisation de l'apprentissage supervisé



## Fonction de coût (*cost function*)

Fonction combinant les écarts avec tous les points de l'ensemble considéré

$$J(w, b) : \bigoplus_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) \rightarrow \text{écart total}$$

$\bigoplus_{i=1}^m$  : opérateur combinant les différents écarts obtenues via la fonction d'écart

$w, b$  : paramètres de la fonction (parameters)

**Intuition:** il s'agit d'une mesure d'à quel point une prédiction est loin de l'ensemble des données

**Note:** par cette formulation, on remarque que l'évaluation du coût dépend des valeurs des paramètres



## Apprentissage supervisé

Tâche consistant à déterminer les paramètres ( $w$  et  $b$ ) qui minimisent une fonction de coût

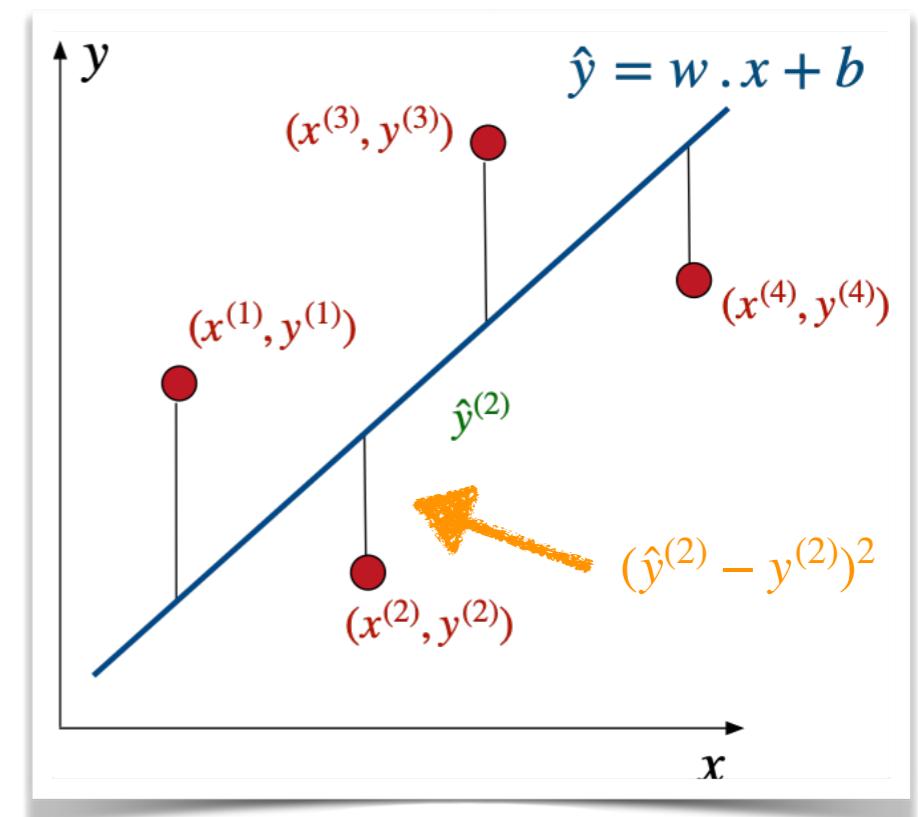
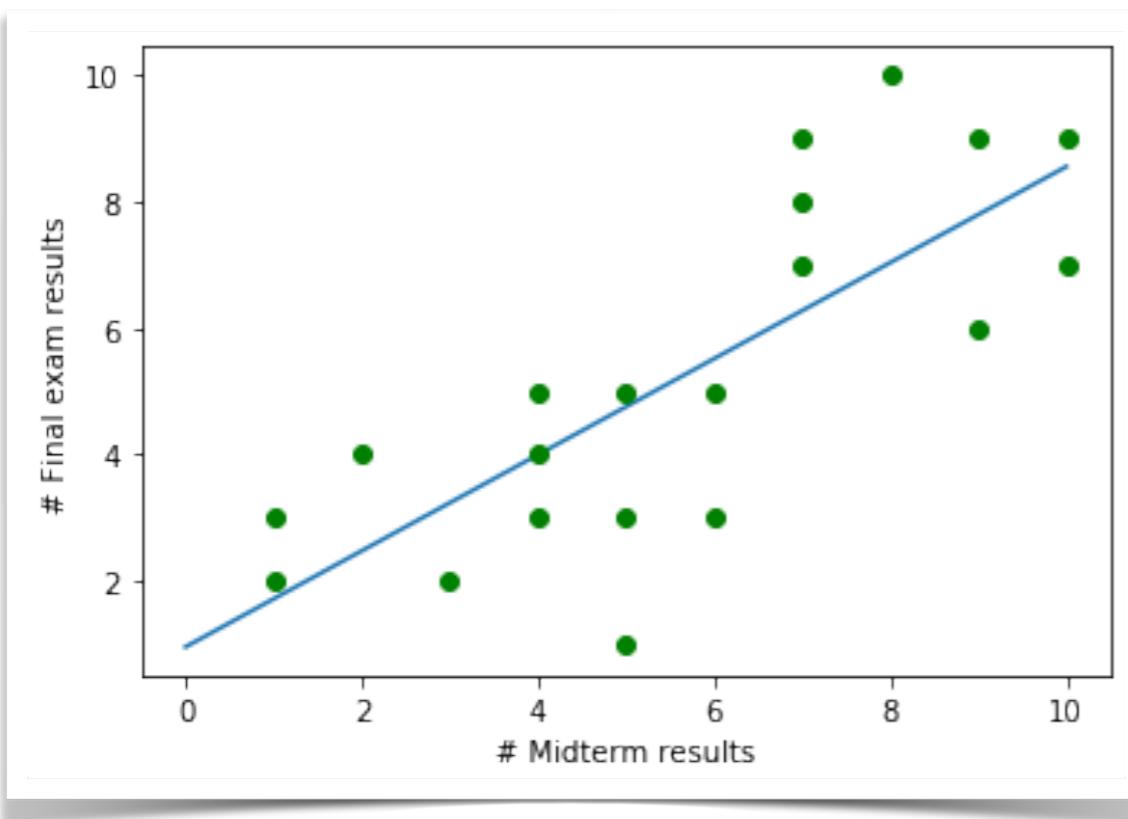
$$\min_{w,b} J(w, b)$$

**Note:** la fonction de coût (et d'écart) dépendent du type de prédiction à réaliser

**Tâche de régression:** erreur quadratique moyenne

**Tâche de classification:** entropie croisée

# Retour au problème de l'étudiant



 **Erreur quadratique moyenne (*mean squared error*)**

Fonction de coût communément utilisée pour des tâches de régression

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)})$$
$$= \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2 \text{ avec } \hat{y}^{(i)} = w \cdot x^{(i)} + b$$

**Note:** le facteur  $1/2$  ne change rien aux écarts, mais va simplifier les futurs calculs

**Notre tâche:** minimiser cette fonction et ainsi obtenir notre équation de droite

# Descente de gradient

## Erreur quadratique moyenne

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2 \text{ avec } \hat{y}^{(i)} = w \cdot x^{(i)} + b$$

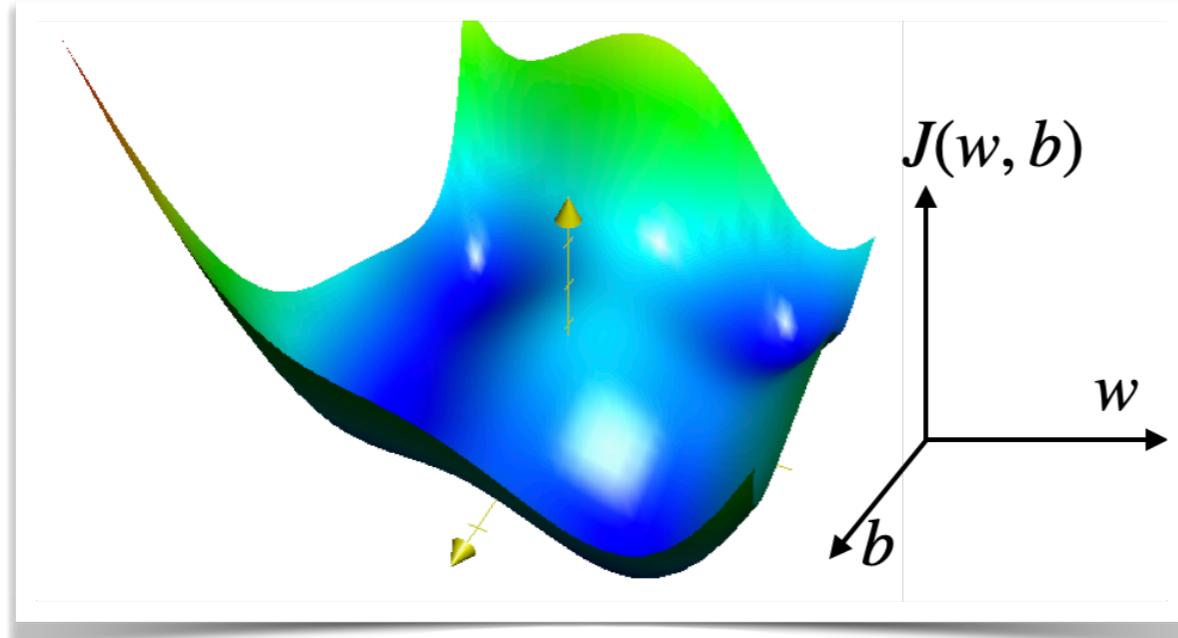


Comment minimiser cette fonction ?

**Descente de gradient:** méthode (parmi d'autres) pour minimiser une fonction

**Hypothèse:** nécessite d'avoir une fonction de coût principalement différentiable

**Fonctionnement:** améliore les paramètres étape par étape en suivant la direction donnée par le gradient



Basé sur trois ingrédients essentiels:

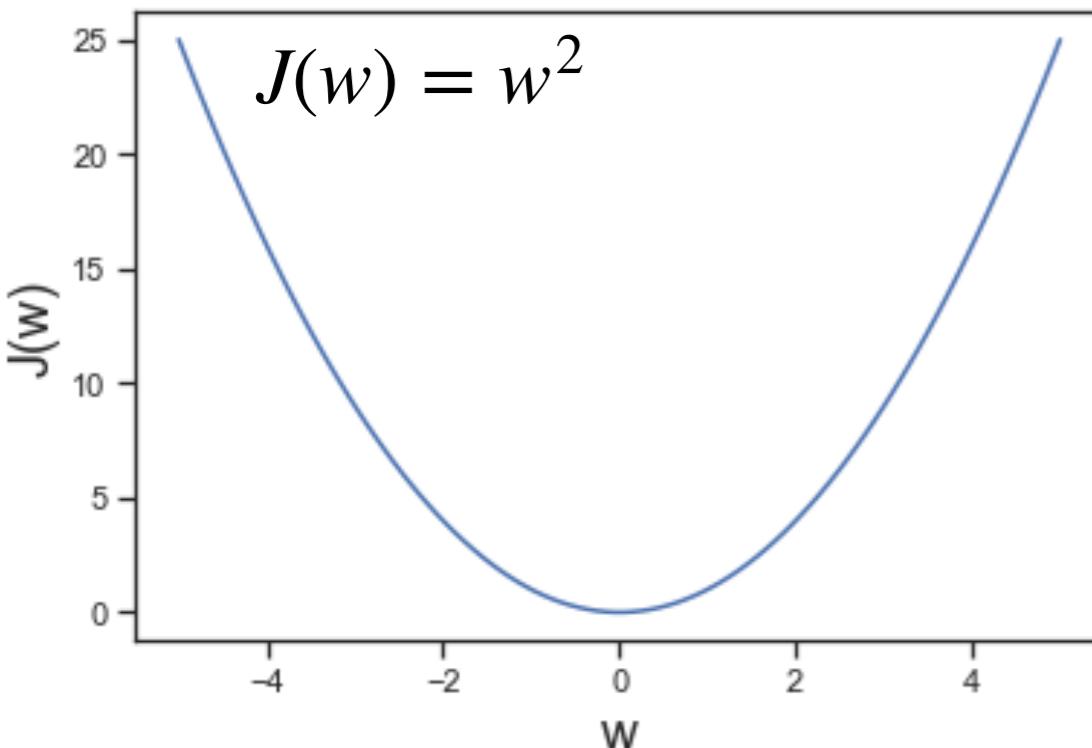
- (1) Le choix d'une position initiale
- (2) La direction du mouvement à prendre
- (3) L'intensité du mouvement à prendre

On itère jusqu'à ce qu'un critère d'arrêt soit atteint

**Analogie:** cette méthode peut être vue comme un *hill climbing* (module 3) dans un espace continu

**Différence:** on va exploiter le fait que la fonction soit différentiable pour déterminer la direction à prendre

# Exemple d'une descente de gradient (fonction à 1 paramètre)



Fonction à minimiser:  $J(w) = w^2, w \in \mathbb{R}$

(1) Initialisation: valeur aléatoire pour  $w$

(2) Direction: obtenue avec la dérivée de  $J(w)$

Il s'agit de la direction de la pente la plus forte

(3) Intensité: obtenue par une valeur positive

Il s'agit du taux d'apprentissage (*learning rate*)

$$\frac{dJ(w)}{dw} = 2w$$

$$\alpha \in ]0,1]$$

Etape d'une descente de gradient: mise à jour des paramètres

Dans notre exemple, il y a trois situations possibles

$$w > 0 \rightarrow \frac{dJ(w)}{dw} > 0$$

Cas 1:  $w$  est positif

Conséquence: la mise à jour va diminuer  $w$

$$w < 0 \rightarrow \frac{dJ(w)}{dw} < 0$$

Cas 2:  $w$  est négatif

Conséquence: la mise à jour va augmenter  $w$

$$w = 0 \rightarrow \frac{dJ(w)}{dw} = 0$$

Cas 3:  $w$  est nul

Conséquence: la descente de gradient a convergé

$$w \leftarrow w - \alpha \frac{dJ(w)}{dw}$$



Appliquons cette idée sur des fonctions ayant plus de paramètres

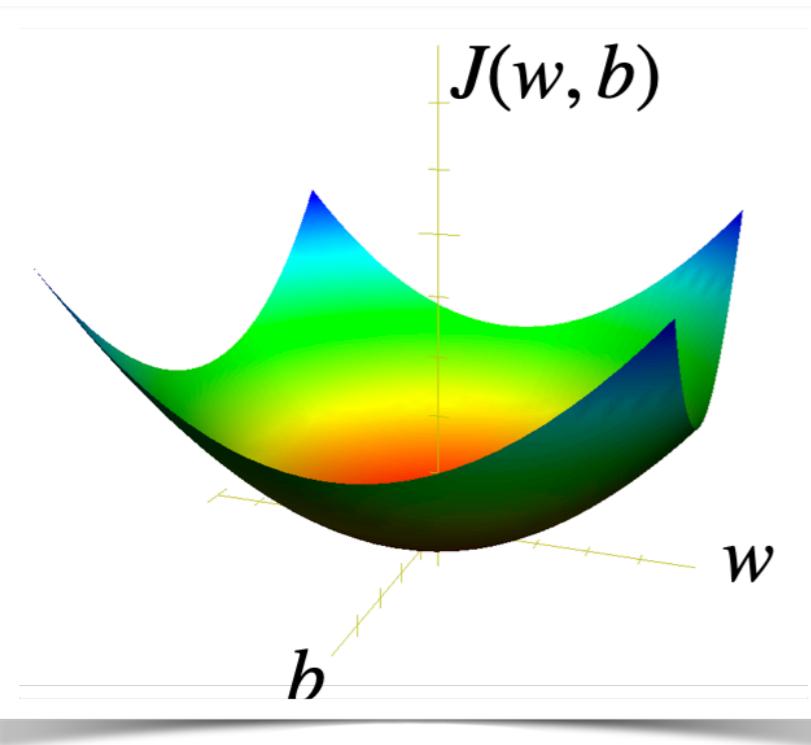
# Descente de gradient dans le cas général



**Une seule variable:** la direction de la plus forte pente est donnée par la **dérivée**

**Plusieurs variables:** la direction de la pente la plus forte est donnée par le **gradient**

**Gradient d'une fonction:** vecteur constitué des dérivées partielles de chaque variable



$$\text{Gradient: } \nabla f(x_1, \dots, x_n) = \left( \frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)$$

**Exemple:** erreur quadratique moyenne

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2 \text{ avec } \hat{y}^{(i)} = w \cdot x^{(i)} + b$$

$$\nabla J(w, b) = \left( \frac{\partial J}{\partial w}, \frac{\partial J}{\partial b} \right)$$

**Etape d'une descente de gradient:** mise à jour de **tous les paramètres suivant sa dérivée partielle**

$$w = w - \alpha \frac{\partial J(w, b)}{\partial w}$$

$$b = b - \alpha \frac{\partial J(w, b)}{\partial b}$$

**Utilisation du gradient:** la fonction de coût doit être différentiable (en grande partie)

# Convexité et gradient de la fonction de coût

## Erreur quadratique moyenne

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2$$

avec  $\hat{y}^{(i)} = w \cdot x^{(i)} + b$



A t-on une convergence vers un minimum global ?

Bonne nouvelle: oui, dans ce cas précis

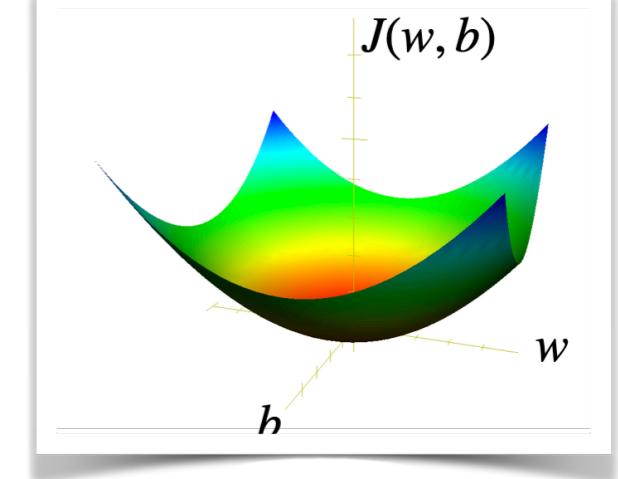
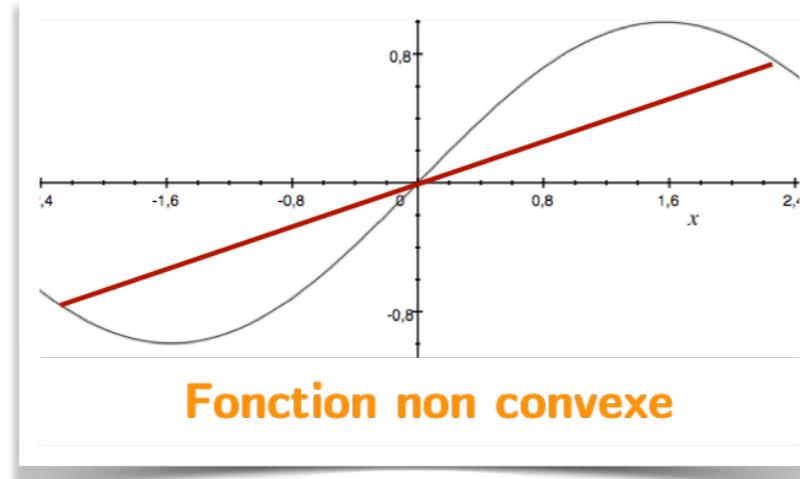
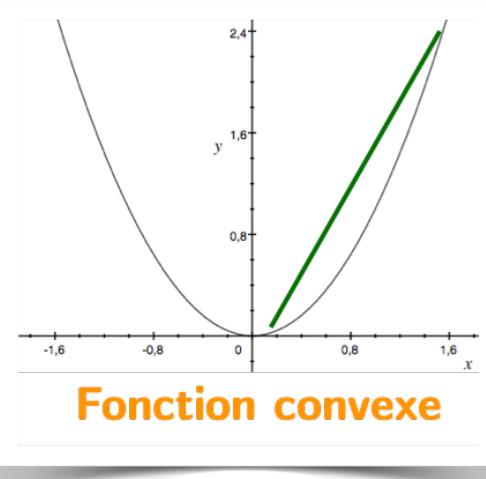
Mauvaise nouvelle: non, dans le cas général

La différence vient de la convexité de la fonction



### Convexité

Une fonction est convexe si, pour deux positions quelconques de la fonction, la droite liant les deux points est toujours au dessus de la fonction



Propriété: une fonction convexe garantit que la descente de gradient converge vers le minimum global

Condition: pour cela, il faut que le taux d'apprentissage ne soit pas trop grand

Régression linéaire avec une erreur quadratique moyenne: la fonction de coût est convexe (parabole)

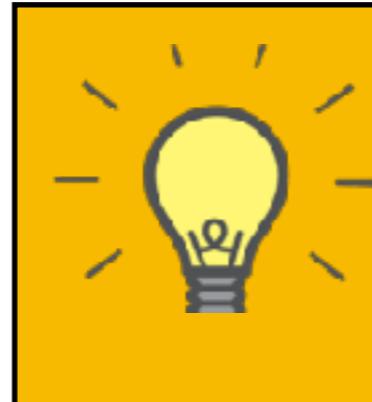
# Gradient de la fonction de coût



Quel est le gradient de notre fonction de coût ?

**Procédure:** calculs basés sur des règles d'analyse

**Fonction à dériver:**  $J(w, b) = \frac{1}{2m} \sum_{i=1}^m (w \cdot x^{(i)} + b - y^{(i)})^2$



**Règles d'analyse**

$$\frac{d(ax + bx)}{dx} = \frac{d(ax)}{dx} + \frac{d(bx)}{dx}$$

$$\frac{d(ax)}{dx} = a \frac{d(x)}{dx}$$

**(1) Gradient:**  $\nabla J(w, b) = \left( \frac{\partial J}{\partial w}, \frac{\partial J}{\partial b} \right)$

**(2) Dérivée partielle selon w:**  $\frac{\partial J(w, b)}{\partial w} = \frac{\partial \left( \frac{1}{2m} \sum_{i=1}^m (w \cdot x^{(i)} + b - y^{(i)})^2 \right)}{\partial w}$

**(3) Application des règles d'analyse:**  $\frac{\partial J(w, b)}{\partial w} = \frac{1}{2m} \left( \frac{\partial \left( (w \cdot x^{(1)} + b - y^{(1)})^2 \right)}{\partial w} + \dots + \frac{\partial \left( (w \cdot x^{(m)} + b - y^{(m)})^2 \right)}{\partial w} \right)$

On sort le terme constant, et on applique la dérivée partielle à chaque élément de la somme

**(4) Dérivée partielle pour une donnée spécifique:** 
$$\begin{aligned} \frac{\partial \left( (w \cdot x^{(i)} + b - y^{(i)})^2 \right)}{\partial w} &= 2x^{(i)}(w \cdot x^{(i)} + b - y^{(i)}) \\ &= 2x^{(i)}(\hat{y}^{(i)} - y^{(i)}) \end{aligned}$$
 Contribution de la donnée  $i$  pour le paramètre  $w$

**(5) Terme final:**  $\frac{\partial J(w, b)}{\partial w} = \frac{1}{2m} \left( 2x^{(1)}(\hat{y}^{(1)} - y^{(1)}) + \dots + 2x^{(m)}(\hat{y}^{(m)} - y^{(m)}) \right) = \frac{1}{m} \sum_{i=1}^m x^{(i)}(\hat{y}^{(i)} - y^{(i)})$

Moyenne de la contribution de chaque donnée

# Descente de gradient: algorithme

`gradientDescent( $D, \alpha$ ) :`

`$w, b = \text{initializeRandomly}()$`

`repeat until convergence :`

`compute  $\hat{y}^{(i)}$   $\forall i \in \{1, \dots, m\}$`

`compute  $J(w, b)$`

`compute  $\frac{\partial J}{\partial w}, \frac{\partial J}{\partial b}$`

`$w = w - \alpha \frac{\partial J}{\partial w}$`

`$b = b - \alpha \frac{\partial J}{\partial b}$`

`return  $w, b$`

**Entrée:** l'ensemble d'entraînement et le taux d'apprentissage

**On donne une valeur initiale aléatoire aux paramètres**

**On itère tant qu'on est pas arrivé à convergence**

**On calcule la prédiction du modèle actuel**

**On évalue la fonction de coût**

**On évalue le gradient**

**Itération de descente de gradient selon  $w$**

**Itération de descente de gradient selon  $b$**

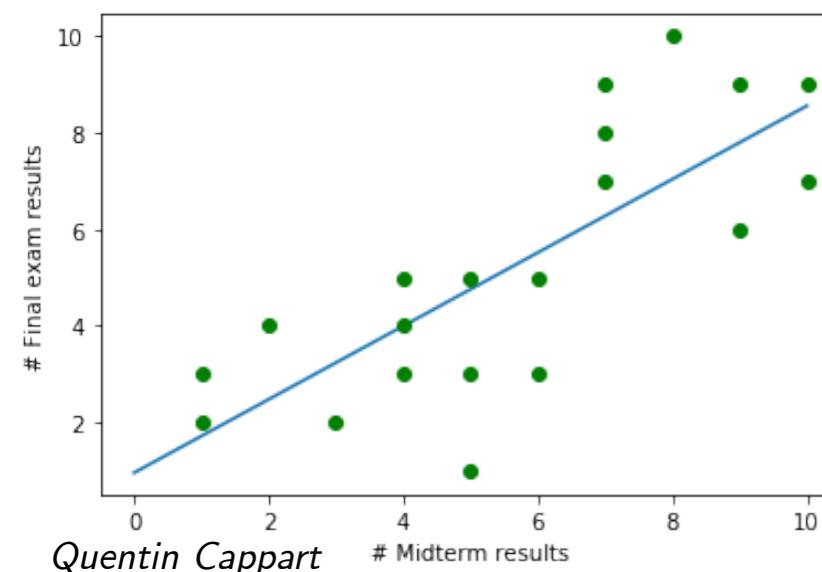
$$\hat{y} = f(x) = w \cdot x + b$$

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2$$

$$\frac{\partial J(w, b)}{\partial w} = \frac{1}{m} \sum_{i=1}^m x^{(i)} (\hat{y}^{(i)} - y^{(i)})$$

$$\frac{\partial J(w, b)}{\partial b} = \frac{1}{m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})$$

Modèle une fois entraîné:  $\hat{y} = 0.76x + 0.96$



Dérivation laissée en exercice

**Note:** il ne s'agit pas de la meilleure approche pour une régression linéaire

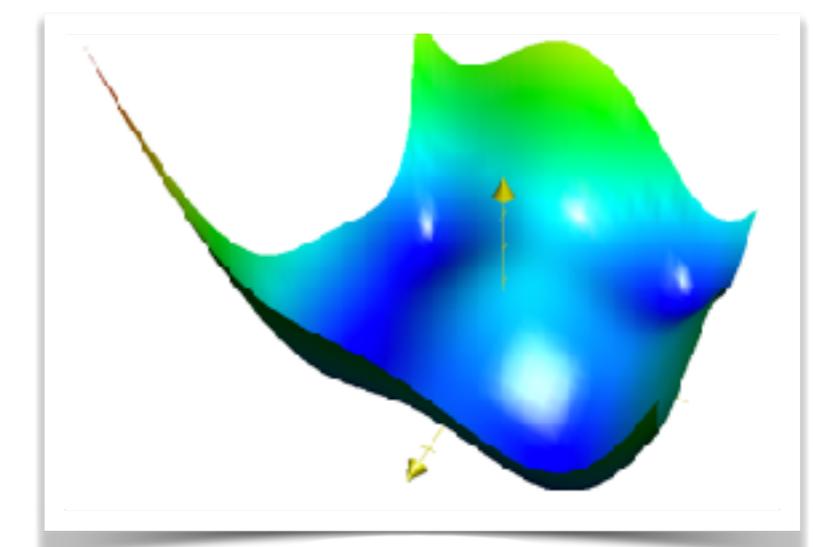
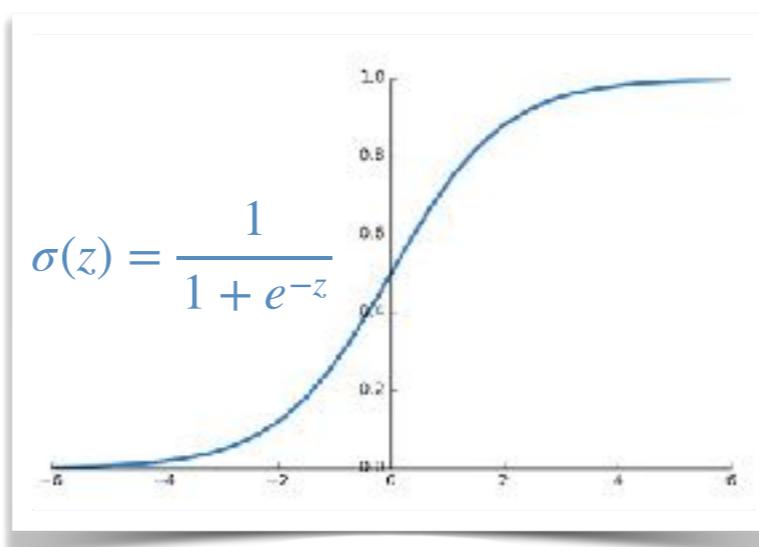
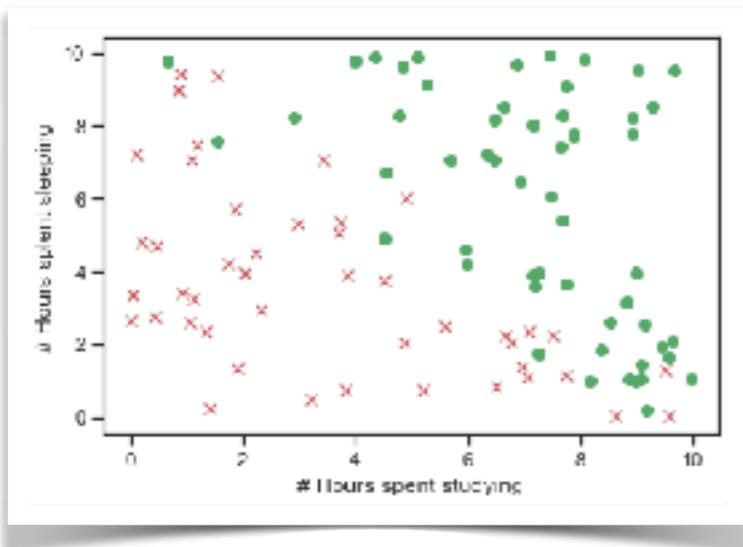
**Avantage:** processus général qu'on va adapter à des cas plus complexes

**Exemples:** régression multiple, logistique, et réseaux de neurones

# Table des matières

## Apprentissage supervisé

- ✓ 1. Motivation et intérêt de l'apprentissage automatique
- ✓ 2. Classification des principaux types d'apprentissage
- ✓ 3. Définition de l'apprentissage supervisé
- ✓ 4. Méthode de la régression linéaire simple
- ✓ 5. Apprentissage par descente de gradient
- 6. Méthode de la régression linéaire multiple
- 7. Méthode de la régression logistique
- 8. Graphe de dépendance, *forward pass*, et *backward pass*



# Régression linéaire multiple

## Problème de l'étudiant

Mise en situation: l'étudiant souhaite maintenant utiliser plusieurs caractéristiques ( $x$ ) pour sa prédiction

Intuition: l'idée est d'avoir une représentation plus fidèle de la réalité



Principe: identique à la régression linéaire

Difficulté: plus difficile à visualiser (espace à  $n$  dimension)

Différence: une donnée est maintenant représentée par un vecteur de caractéristiques

Donnée d'entraînement :  $(x^{(i)}, y^{(i)})$

$x^{(i)}$  : vecteur des caractéristiques de la donnée  $i$

$x_j^{(i)}$  : caractéristique  $j$  de la donnée  $i$

$y^{(i)}$  : label de la donnée  $i$  (inchangé)

$$\hat{y} = (w_1 \ \dots \ w_n) \times \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + b = w^T x + b$$

**Fonction de prédiction linéaire généralisée**  
Fonction de prédiction ayant la forme suivante:

$$\hat{y} = w_1 x_1 + w_2 x_2 + \dots + w_n x_n + b = w^T x + b$$

$x_1, \dots, x_n$  : caractéristiques d'une donnée du problème  
 $b, w_1, \dots, w_n$  : paramètres devant être appris

# Régression linéaire multiple



Combien de paramètres a t-on avec  $n$  caractéristiques ?

Cas général: pour  $n$  caractéristiques, on a  $n+1$  paramètres

$$x \in \mathbb{R}^n$$

$$w \in \mathbb{R}^n$$

$$b \in \mathbb{R}$$

gradientDescent( $D, \alpha$ ) :

$w, b = \text{initializeRandomly}()$

repeat until convergence :

compute  $\hat{y}^{(i)} \forall i \in \{1, \dots, m\}$

compute  $J(w_1, \dots, w_n, b)$

compute  $\frac{\partial J}{\partial w_1}, \dots, \frac{\partial J}{\partial w_n}, \frac{\partial J}{\partial b}$

$w_1 = w_1 - \alpha \frac{\partial J}{\partial w_1}$

...

$w_n = w_n - \alpha \frac{\partial J}{\partial w_n}$

$b = b - \alpha \frac{\partial J}{\partial b}$

return  $w_1, \dots, w_n, b$

$$\hat{y} = f(x) = w^T x + b$$

$x$  et  $w$  sont maintenant des vecteurs

$$J(w_1, \dots, w_n, b) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})^2$$

$$\frac{\partial J(w_1, \dots, w_n, b)}{\partial w_j} = \frac{1}{m} \sum_{i=1}^m x_j^{(i)} (\hat{y}^{(i)} - y^{(i)}) \quad \forall j \in \{1, \dots, n\}$$

le gradient propre à un paramètre considère uniquement la caractéristique qui lui y est liée

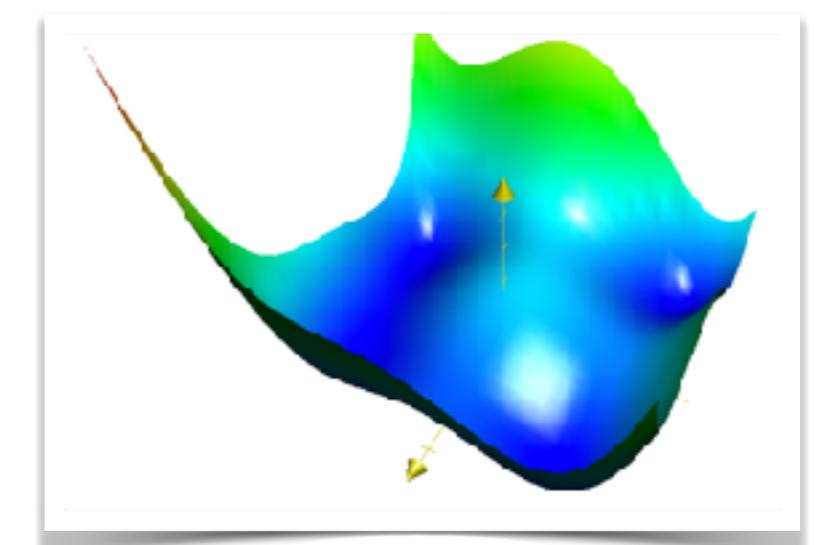
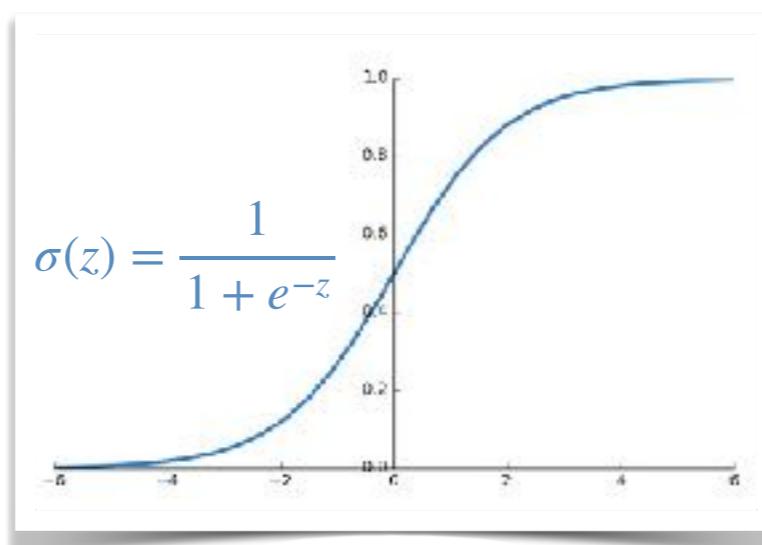
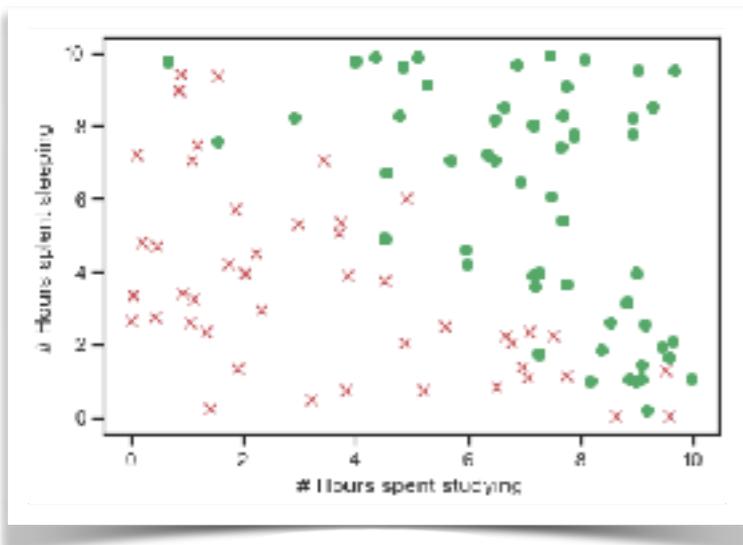
$$\frac{\partial J(w, b)}{\partial b} = \frac{1}{m} \sum_{i=1}^m (\hat{y}^{(i)} - y^{(i)})$$

Le fonctionnement suit exactement le même principe que précédemment

# Table des matières

## Apprentissage supervisé

- ✓ 1. Motivation et intérêt de l'apprentissage automatique
- ✓ 2. Classification des principaux types d'apprentissage
- ✓ 3. Définition de l'apprentissage supervisé
- ✓ 4. Méthode de la régression linéaire simple
- ✓ 5. Apprentissage par descente de gradient
- ✓ 6. Méthode de la régression linéaire multiple
- 7. Méthode de la régression logistique
- 8. Graphe de dépendance, *forward pass*, et *backward pass*



# Mise en situation: prédiction d'une réussite

## Prédiction d'une réussite

**Mise en situation:** l'étudiant souhaite maintenant savoir s'il va réussir ou non son examen

**Caractéristiques:** Il se base sur deux critères, le nombre d'heures d'étude, et de repos



**Son idée:** se baser sur les informations des étudiants précédents

(1) L'étudiant est représenté par deux caractéristiques

$x_1 \in \mathbb{R}$  : le nombre d'heures passées à étudier

$x_2 \in \mathbb{R}$  : le nombre d'heures passées à dormir la veille de l'examen

(2) Le résultat est une valeur binaire

$y \in \{0,1\}$  : l'étudiant a t-il réussi ou non ?

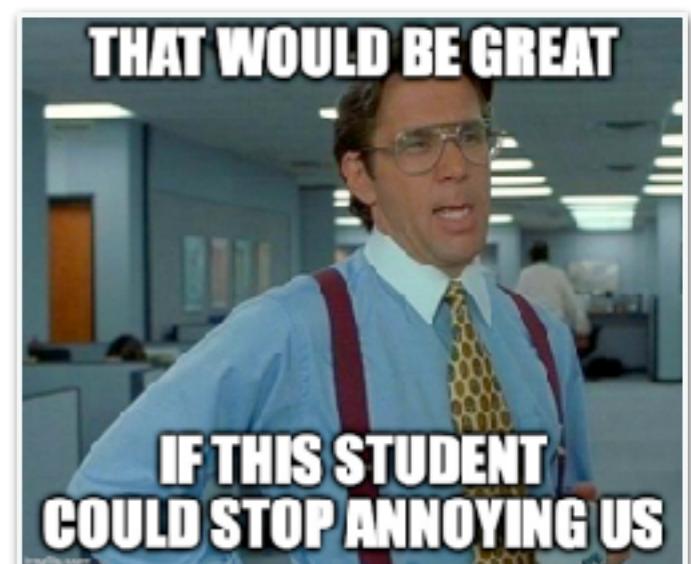
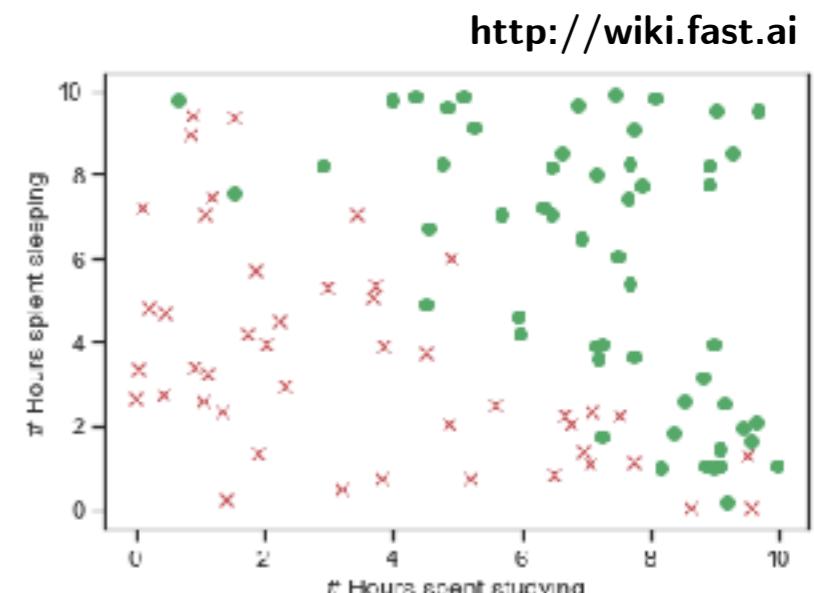
(3) On souhaite construire une fonction donnant une probabilité de succès

$f: \mathbb{R}^2 \rightarrow [0,1]$  : fonction qui retourne une probabilité de succès

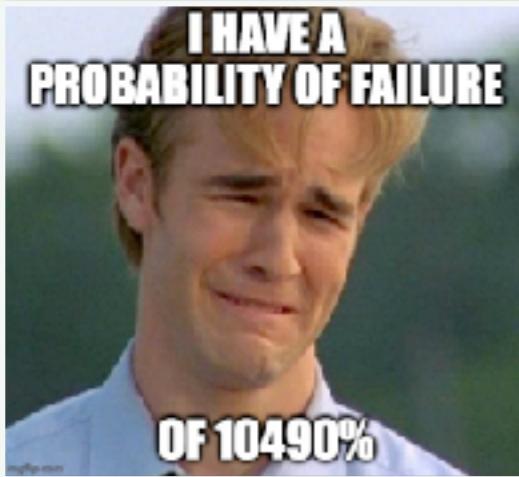
(4) La prédiction peut ensuite se faire suivant un seuil

$f(x) = \hat{y} \geq 0.5 \rightarrow \text{Réussite}$

$f(x) = \hat{y} < 0.5 \rightarrow \text{Echec}$



# Introduction à la régression logistique



Que pensez-vous de l'utilisation d'une régression linéaire ?

**Difficulté:** des valeurs très grandes peuvent être prédites (négatives ou positives)

**Notre situation:** on souhaite obtenir une probabilité de succès (valeur entre 0 et 1)

**Conséquence:** la régression linéaire n'est pas le modèle le plus adapté

**Régression logistique:** étant donné une donnée  $x$ , on veut construire une fonction donnant une probabilité

$\hat{y} = \mathbb{P}(y = 1 | x)$  avec  $\hat{y}$  la probabilité de réussite à l'examen

$\hat{y} \approx 0$  : probable de rater

$\hat{y} \approx 1$  : probable de réussir

$\hat{y} \approx 0.5$  : décision incertaine

**Exigence:** on doit obtenir une valeur entre 0 et 1



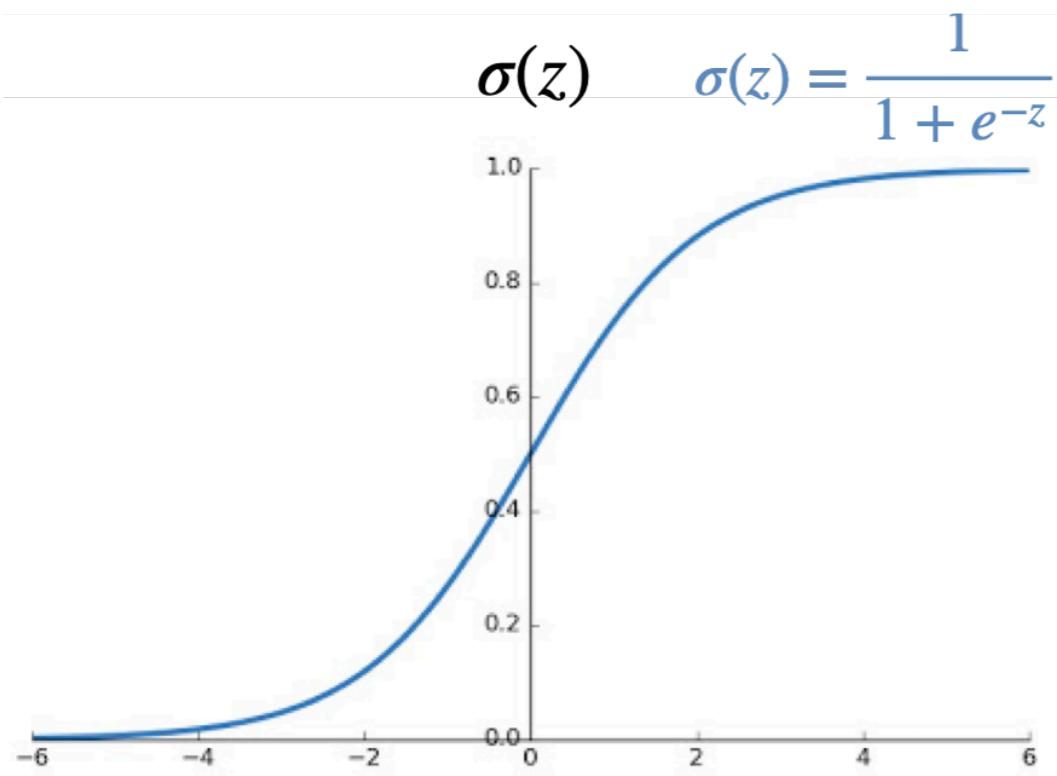
**Idée de conception:** construire une fonction compressant n'importe quelle valeur réelle entre 0 et 1

 **Fonction sigmoïde**

Fonction compressant n'importe quelle valeur réelle entre 0 et 1

$$\sigma(z) : \mathbb{R} \rightarrow [0,1] = \frac{1}{1 + e^{-z}}$$

# Fonction sigmoïde et régression logistique



**Fonction sigmoïde:** compresse une valeur réelle entre 0 et 1

(1) Cas où  $z$  est positivement grand:  $\sigma(z) = \frac{1}{1 + 0} = 1$

**Interprétation:** on est confiant sur la réussite

(2) Cas où  $z$  est négativement grand:  $\sigma(z) = \frac{1}{1 + \infty} = 0$

**Interprétation:** on est confiant sur l'échec

(3) Cas où  $z$  est nul:  $\sigma(z) = \frac{1}{1 + 1} = 0.5$

**Interprétation:** on est incertain sur le choix

**Avantage supplémentaire:** la fonction est différentiable

**Idée:** réaliser une combinaison linéaire (comme avant) mais suivie ensuite d'une fonction sigmoïde

 **Régression logistique**

— Fonction de prédiction ayant la forme suivante:

$$\hat{y} = \sigma(w^T x + b) \text{ avec } \hat{y} \in [0,1]$$

$x$ : vecteur des caractéristiques d'une donnée du problème

$b, w$ : paramètres devant être appris ( $w$  est un vecteur)

**Note:** on peut ensuite définir un seuil de décision pour la classification finale

$f(x) = \hat{y} \geq 0.5 \rightarrow \text{Réussite}$

$f(x) = \hat{y} < 0.5 \rightarrow \text{Echec}$

# Fonction d'écart pour la régression logistique



Comment caractériser la qualité d'une régression logistique ?

**Objectif:** mesurer l'écart entre valeur prédite et la vraie valeur via une fonction d'écart

**Ecart nul:** indique une prédiction parfaite sur nos données

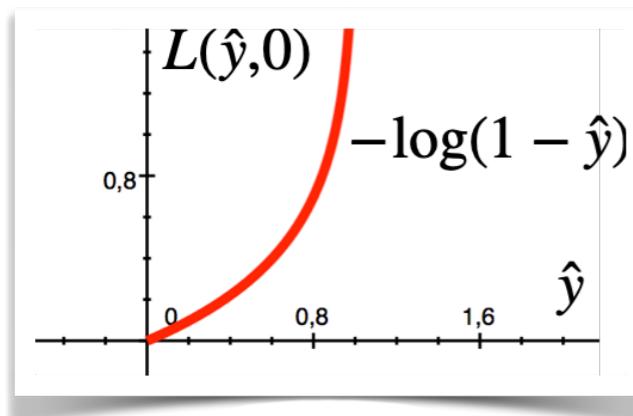


**Erreur de l'entropie croisée binaire (*binary cross entropy loss*)**

Fonction d'écart communément utilisée pour des tâches de classification binaire

$$L(\hat{y}, y) = - (y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \text{ avec } \hat{y} = \sigma(w^T x + b)$$

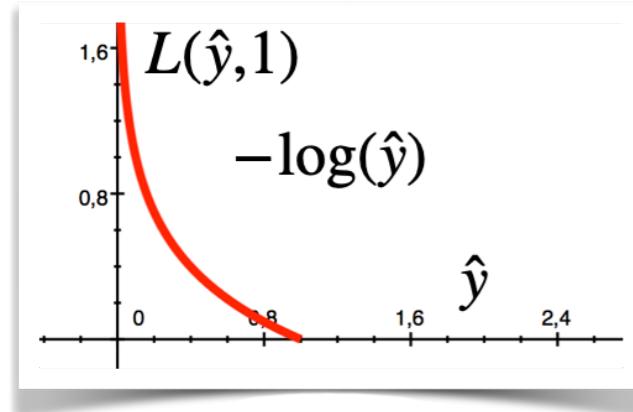
**Intuition:** cette fonction pénalise l'écart entre la prédiction et une vraie valeur de 1 ou de 0



**Cas 1: situation où  $y = 0$  (valeur d'échec - l'étudiant a raté l'examen)**

**Impact sur la fonction:**  $L(\hat{y}, 0) = -\log(1 - \hat{y})$

**Conséquence:** pousse  $\hat{y}$  à être petit ( $\hat{y} \rightarrow 0$ )



**Cas 2: situation où  $y = 1$  (valeur de succès - l'étudiant a réussi l'examen)**

**Impact sur la fonction:**  $L(\hat{y}, 1) = -\log(\hat{y})$

**Conséquence:** pousse  $\hat{y}$  à être grand ( $\hat{y} \rightarrow 1$ )

# Fonction de coût pour la régression logistique

## Fonction de coût d'une régression logistique

**Etape 1:** effectuer la régression logistique sur toutes les données d'entraînement

$$\left\{ \left( x^{(1)}, y^{(1)} \right), \left( x^{(2)}, y^{(2)} \right), \dots, \left( x^{(m)}, y^{(m)} \right) \right\} \rightarrow \hat{y}^{(i)} \approx y^{(i)} \quad \forall i \in \{1, \dots, m\}$$
$$\rightarrow \sigma(w^T x^{(i)} + b) \approx y^{(i)}$$

**Etape 2:** prendre l'écart moyen de toutes ces prédictions

$$J(w, b) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)})$$
$$= -\frac{1}{m} \sum_{i=1}^m \left( y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right)$$

Avec  $\hat{y}^{(i)} = \sigma(w^T x^{(i)} + b)$  et  $\sigma(z) = \frac{1}{1 + e^{-z}}$

**Principe:** exactement pareil que précédemment, mais avec notre nouvelle fonction de perte

**Propriété:** cette fonction est **différentiable** et peut donc être minimisée par une descente de gradient

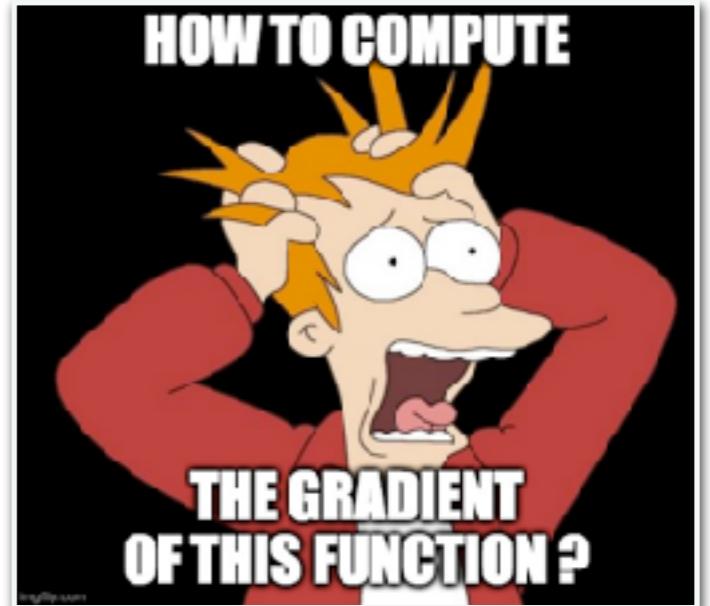
**Propriété:** cette fonction est également **convexe**, on a ainsi une convergence vers un minimum global

# Calcul du gradient

$$\begin{aligned} J(w, b) &= \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) \\ &= -\frac{1}{m} \sum_{i=1}^m \left( y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right) \end{aligned}$$

Avec  $\hat{y}^{(i)} = \sigma(w^T x^{(i)} + b)$  et  $\sigma(z) = \frac{1}{1 + e^{-z}}$

Fonction de coût



**Bonne nouvelle:** malgré l'apparence complexe, cette tâche n'est pas si difficile !

**Astuce:** procéder par une décomposition en trois étapes

**Etape 1:** exprimer la fonction en un graphe de dépendances (*computation graph*)

**Etape 2:** évaluer la fonction avec la valeur actuelle des paramètres (*forward pass*)

**Etape 3:** utiliser la règle du chaînage (*chain rule*) pour calculer les dérivées partielles (*backward pass*)

**Généralité de cette approche:** il s'agit également du fonctionnement d'un réseau de neurone

**Exemple:** on veut faire une descente de gradient sur cette fonction

$$J(w_1, w_2, w_3) = 5(3w_1 + w_2 w_3^2)$$

# Graphe de dépendances et passe en avant (*forward pass*)



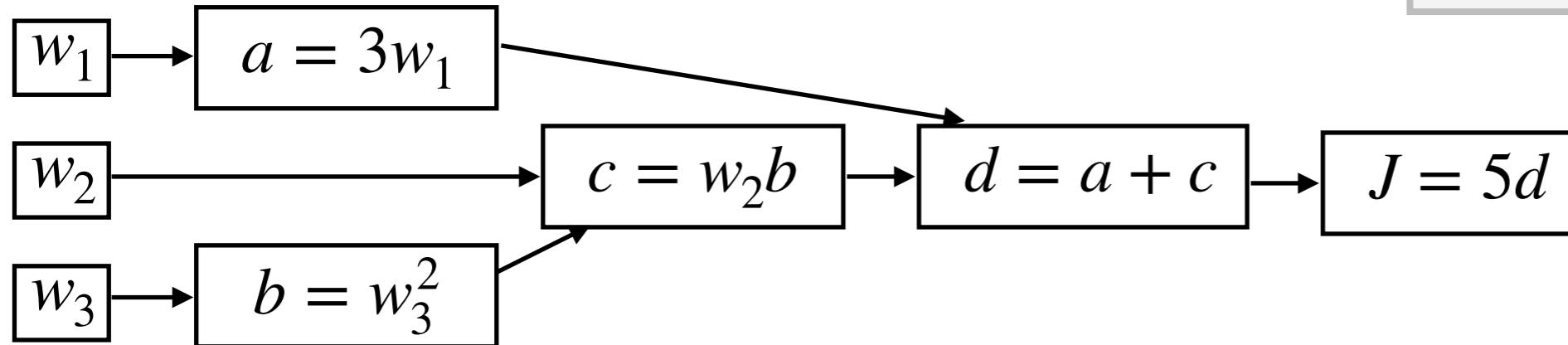
## Graphe de dépendances (*computation graph*)

Représentation d'une expression numérique en une série d'opérations élémentaires

**Intuition:** il s'agit de mettre en évidence toutes les opérations élémentaires ainsi que leurs dépendances

**Exemple:** on veut faire une descente de gradient sur cette fonction

$$J(w_1, w_2, w_3) = 5(3w_1 + w_2 w_3^2)$$

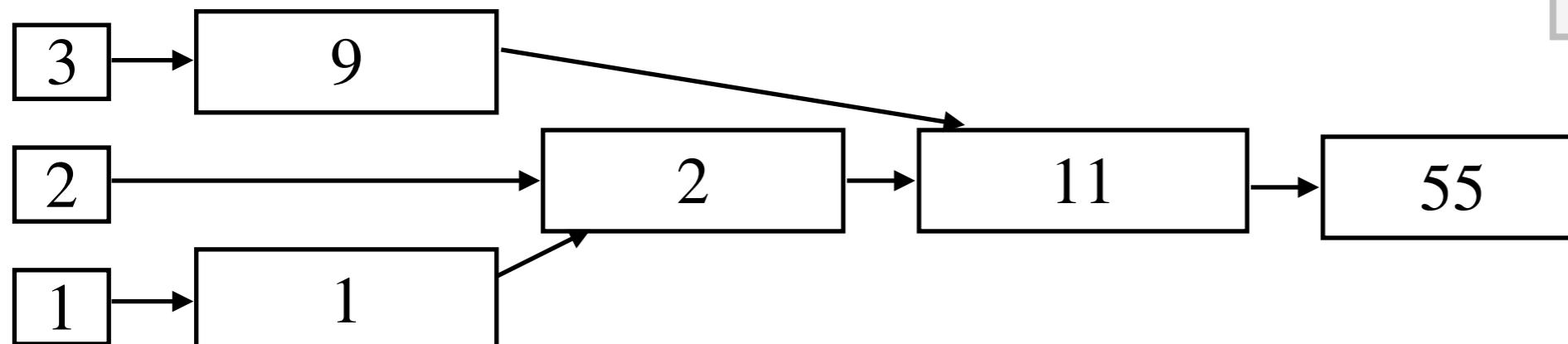


## Passe en avant (*forward pass*)

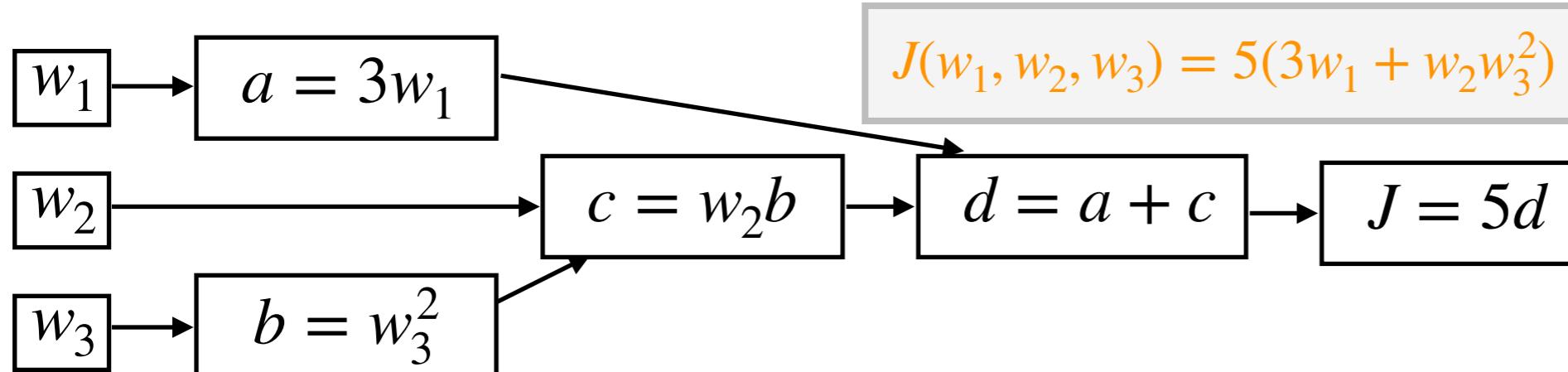
Calcul du coût de la fonction de prédiction, selon la valeur actuelle des paramètres ( $w$  et  $b$ )

**Exemple:** supposons qu'on ait les valeurs suivantes pour les 3 paramètres

$$w_1 = 3, w_2 = 2, w_3 = 1$$



# Passe en arrière (backward pass)



**Passe en arrière (backward pass or backpropagation)**

Calcul de la valeur des dérivées partielles de chaque paramètre pour obtenir le gradient

**Note:** pour ce calcul, on commence à la dernière valeur calculée, et on remonte jusqu'aux paramètres

**Principe:** on utilise la règle du chaînage pour évaluer les dérivées partielles

$$\nabla J(w_1, w_2, w_3) = \left( \frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}, \frac{\partial J}{\partial w_3} \right)$$



**Règle du chaînage**

si  $z$  dépend de  $y$ ,  
et si  $y$  dépend de  $x$ .  $\frac{dz}{dx} = \frac{dz}{dy} \times \frac{dy}{dx}$

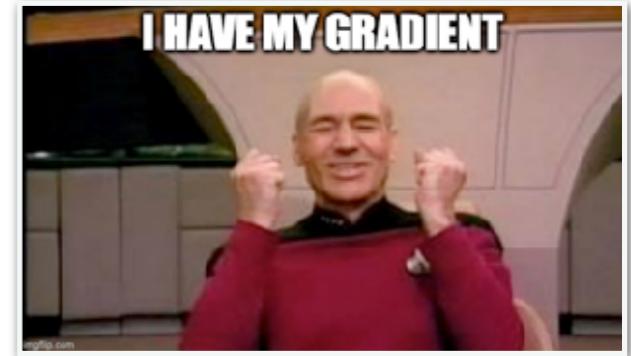
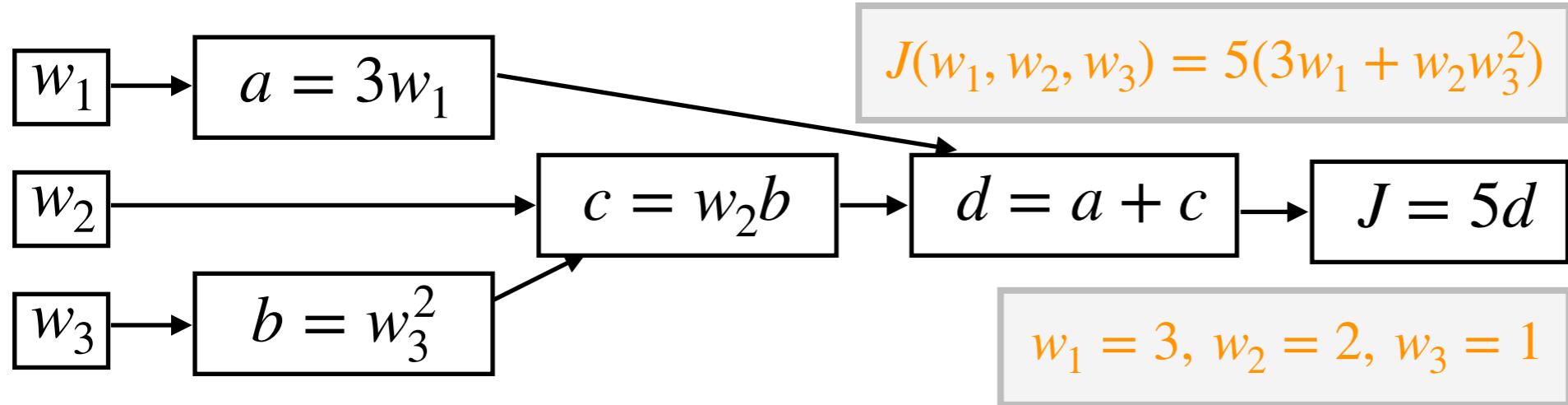
$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial d} \times \frac{\partial d}{\partial a} \times \frac{\partial a}{\partial w_1} = 5 \times 1 \times 3 = 15$$

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial d} \times \frac{\partial d}{\partial c} \times \frac{\partial c}{\partial w_2} = 5 \times 1 \times b = 5b = 5w_3^2 = 5$$

$$\frac{\partial J}{\partial w_3} = \frac{\partial J}{\partial d} \times \frac{\partial d}{\partial c} \times \frac{\partial c}{\partial b} \times \frac{\partial b}{\partial w_3} = 5 \times 1 \times w_2 \times 2w_3 = 10w_2w_3 = 20$$



# Dernière étape: descente de gradient



**Situation actuelle:** on a l'évaluation de notre gradient pour les valeurs actuels de nos paramètres

$$\nabla J(w_1, w_2, w_3) = \left( \frac{\partial J}{\partial w_1}, \frac{\partial J}{\partial w_2}, \frac{\partial J}{\partial w_3} \right) = (15, 5, 20)$$

**Prochaine étape:** utiliser ces valeurs pour effectuer la descente de gradient

**Taux d'apprentissage:**  $\alpha = 0.1$

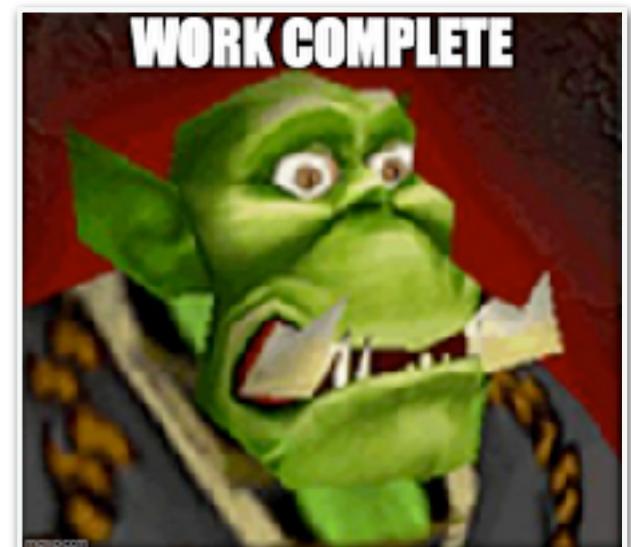
**Valeurs actuelles:**  $w_1 = 3, w_2 = 2, w_3 = 1$

$$w_1 \leftarrow w_1 - \alpha \times 15 = 3 - 0.1 \times 15 = 1.5$$

$$w_2 \leftarrow w_2 - \alpha \times 5 = 2 - 0.1 \times 5 = 1.5$$

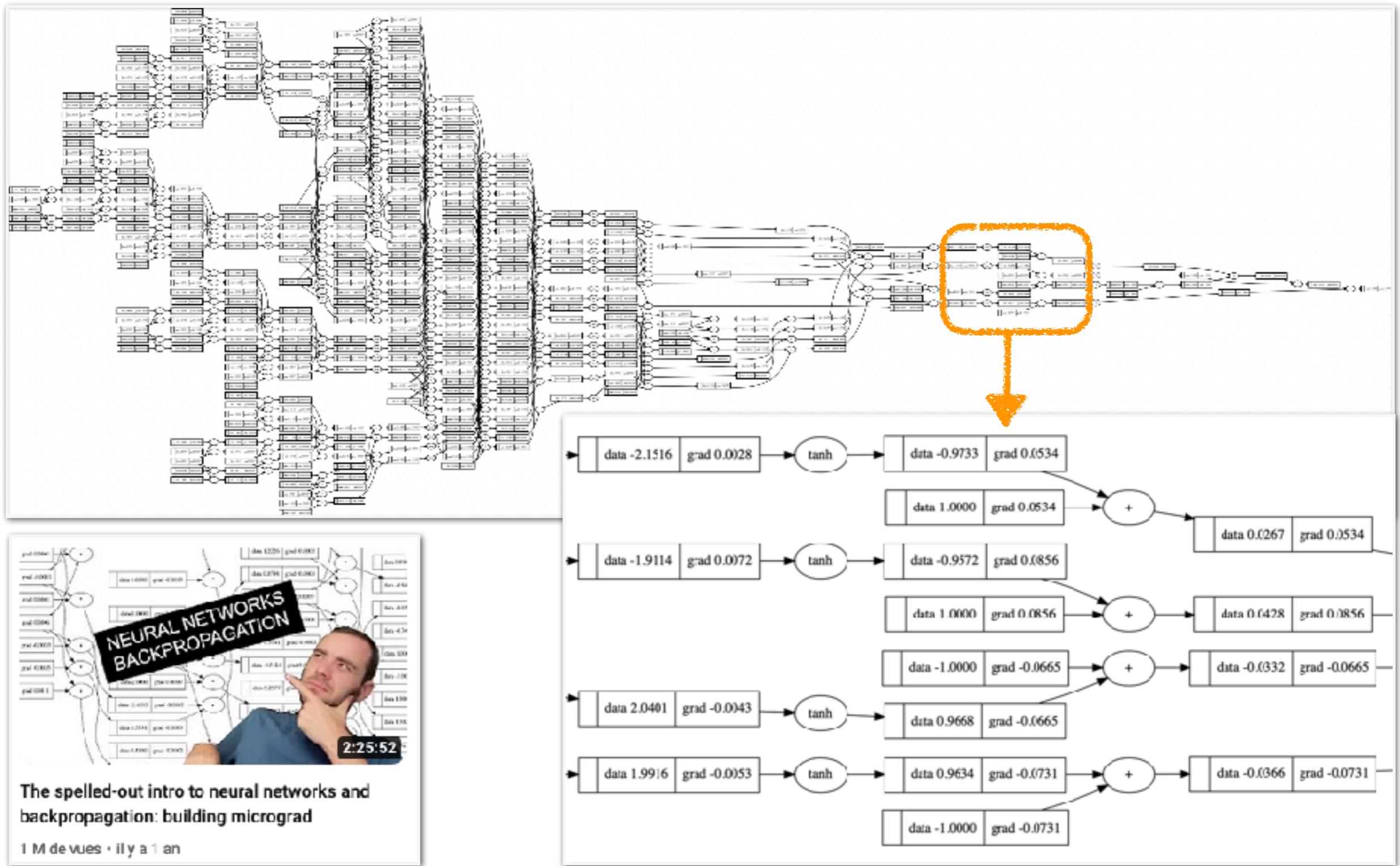
$$w_3 \leftarrow w_3 - \alpha \times 20 = 1 - 0.1 \times 20 = -1$$

**Nouvelles valeurs:**  $w_1 = 1.5, w_2 = 1.5, w_3 = -1$



**Par la suite:** refaire la passe en avant et en arrière avec ces nouvelles valeurs

# Graphe de dépendance en pratique...



Ce principe peut être utilisé pour différentier des fonctions très complexes

# Graphe de dépendances sur notre fonction d'écart



$$L(\hat{y}, y) = - (y \log(\hat{y}) + (1 - y) \log(1 - \hat{y}))$$

Avec  $\hat{y} = \sigma(w^T x + b)$  et  $\sigma(z) = \frac{1}{1 + e^{-z}}$

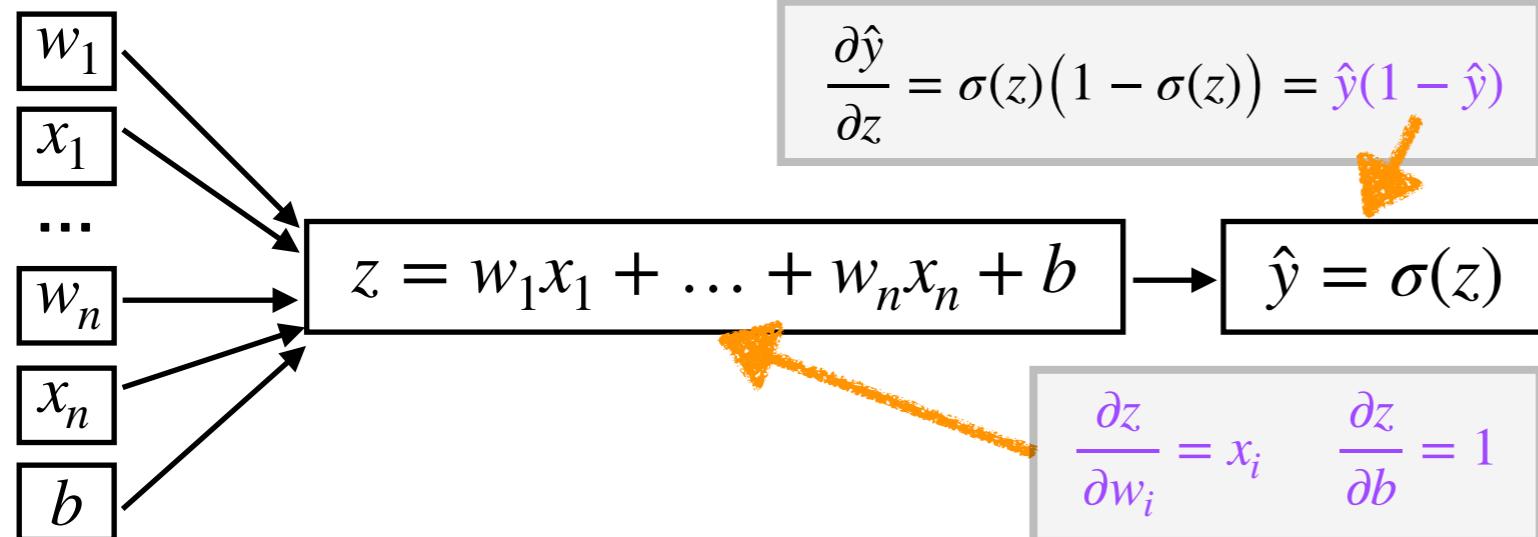


$$\frac{d \log(x)}{dx} = \frac{1}{x}$$

**Objectif:** calculer le gradient

**Principe:** construire le graphe de dépendance

$$\nabla L(\hat{y}, y) = \left( \frac{\partial L}{\partial w_1}, \dots, \frac{\partial L}{\partial w_n}, \frac{\partial L}{\partial b} \right)$$



Dérivation complète:

<https://math.stackexchange.com/questions/78575/derivative-of-sigmoid-function-sigma-x-frac11e-x>

**Prochaine étape:** passe en arrière avec la règle du chainage

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial w_i} = \frac{\hat{y} - y}{\hat{y}(1 - \hat{y})} \times \hat{y}(1 - \hat{y}) \times x_i = x_i(\hat{y} - y)$$

$$\frac{\partial L}{\partial b} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z} \times \frac{\partial z}{\partial b} = \frac{\hat{y} - y}{\hat{y}(1 - \hat{y})} \times \hat{y}(1 - \hat{y}) \times 1 = \hat{y} - y$$

On a ainsi nos dérivées partielles, et donc notre gradient !

$$\begin{aligned} \frac{\partial L(\hat{y}, y)}{\partial \hat{y}} &= -\frac{y}{\hat{y}} + \frac{1-y}{1-\hat{y}} \\ &= \frac{-y(1-\hat{y}) + \hat{y}(1-y)}{\hat{y}(1-\hat{y})} \\ &= \frac{-y + \hat{y}y + \hat{y} - \hat{y}y}{\hat{y}(1-\hat{y})} \\ &= \frac{\hat{y} - y}{\hat{y}(1-\hat{y})} \end{aligned}$$

# Intégration de la fonction de coût



Dernière étape: intégrer la fonction de coût

$$J(w, b) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) = \frac{1}{m} \left( L(\hat{y}^{(1)}, y^{(1)}) + \dots + L(\hat{y}^{(m)}, y^{(m)}) \right)$$

Principe: identique à une régression linéaire, mais avec une autre fonction d'écart

Etat actuel: le gradient de chaque terme de la fonction de perte est connu

$$\frac{\partial L(\hat{y}^{(i)}, y^{(i)})}{\partial w_j} = x_j^{(i)}(\hat{y}^{(i)} - y^{(i)}) \text{ pour toutes les données } i \text{ et paramètres } w_j$$

$$\frac{\partial L(\hat{y}^{(i)}, y^{(i)})}{\partial b} = (\hat{y}^{(i)} - y^{(i)}) \text{ pour toutes les données } i$$

Règles d'analyse

$$\frac{d(ax + bx)}{dx} = \frac{d(ax)}{dx} + \frac{d(bx)}{dx}$$
$$\frac{d(ax)}{dx} = a \frac{d(x)}{dx}$$

Gradient de la fonction de coût: simple application des règles d'analyse

$$\frac{\partial J(w, b)}{\partial w_j} = \frac{1}{m} \left( \frac{\partial L(\hat{y}^{(1)}, y^{(1)})}{\partial w_j} + \dots + \frac{\partial L(\hat{y}^{(m)}, y^{(m)})}{\partial w_j} \right) \quad \forall j \in \{1, \dots, n\}$$

$$\frac{\partial J(w, b)}{\partial b} = \frac{1}{m} \left( \frac{\partial L(\hat{y}^{(1)}, y^{(1)})}{\partial b} + \dots + \frac{\partial L(\hat{y}^{(m)}, y^{(m)})}{\partial b} \right)$$



# Toutes les briques assemblées - problème de l'étudiant

```
1 n_step = 1000
2 alpha = 0.1
3 w1, w2, b = 0, 0, 0
4
5 for j in range(n_step):
6     dw1, dw2, db, J = 0, 0, 0, 0
7     # Considering all the training set
8     for i in range(m):
9
10        # Forward pass
11        z_i = w1 * x1[i] + w2 * x2[i] + b
12        a_i = sigmoid(z_i)
13        J = J + loss(a_i,y[i])
14
15        # Backward pass
16        dz_i = a_i - y[i]
17        dw1 = dw1 + x1[i] * dz_i
18        dw2 = dw2 + x2[i] * dz_i
19        db = db + dz_i
20
21    J = J / m
22    dw1 = dw1 / m
23    dw2 = dw2 / m
24    db = db / m
25    # Gradient descent step
26    w1 = w1 - alpha * dw1
27    w2 = w2 - alpha * dw2
28    b = b - alpha * db
```

## Passe en avant (*forward pass*)

$$J(w, b) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) \quad \text{avec } \hat{y}^{(i)} = \sigma(w^T x^{(i)} + b)$$

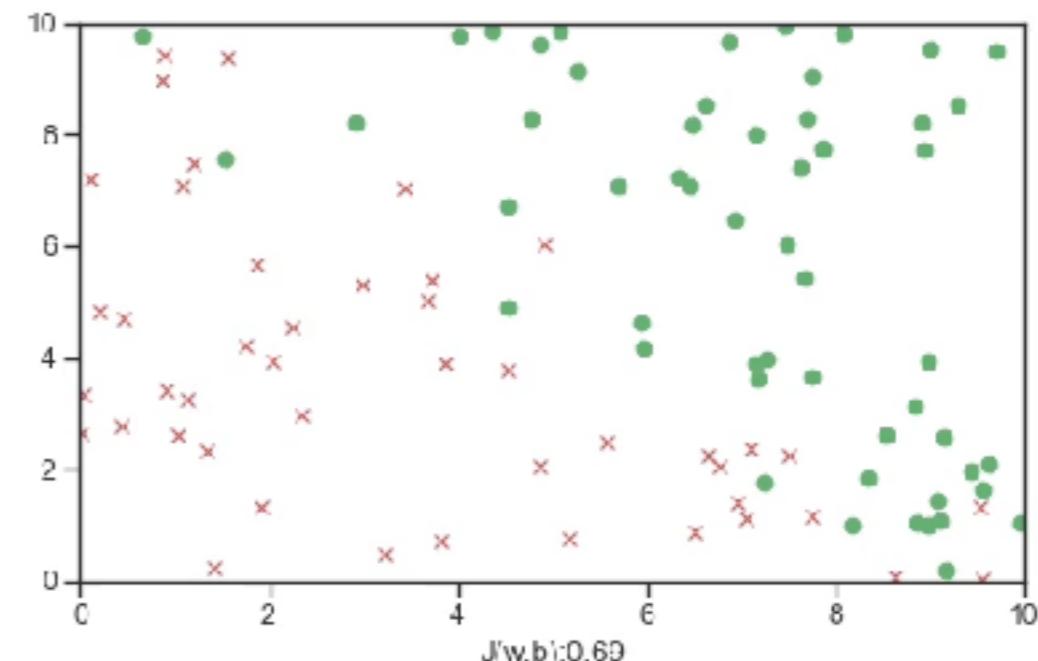
## Passe en arrière (*backward pass*)

$$\frac{\partial J(w, b)}{\partial b} = \frac{1}{m} \left( \frac{\partial L(\hat{y}^{(1)}, y^{(1)})}{\partial b} + \dots + \frac{\partial L(\hat{y}^{(m)}, y^{(m)})}{\partial b} \right)$$

$$\frac{\partial J(w, b)}{\partial w_1} = \frac{1}{m} \left( \frac{\partial L(\hat{y}^{(1)}, y^{(1)})}{\partial w_1} + \dots + \frac{\partial L(\hat{y}^{(m)}, y^{(m)})}{\partial w_1} \right)$$

$$\frac{\partial J(w, b)}{\partial w_2} = \frac{1}{m} \left( \frac{\partial L(\hat{y}^{(1)}, y^{(1)})}{\partial w_2} + \dots + \frac{\partial L(\hat{y}^{(m)}, y^{(m)})}{\partial w_2} \right)$$

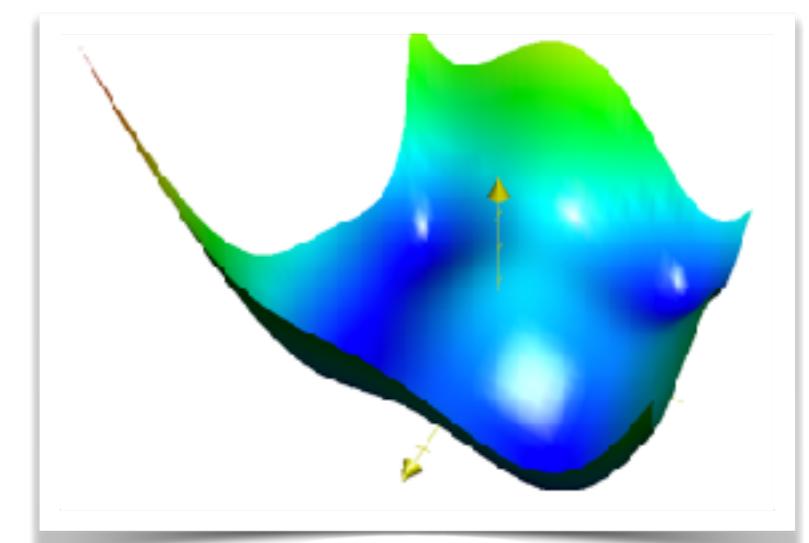
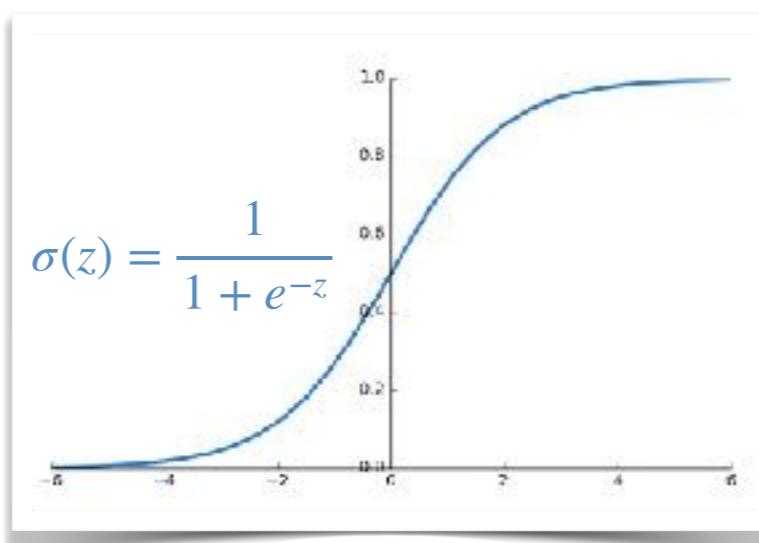
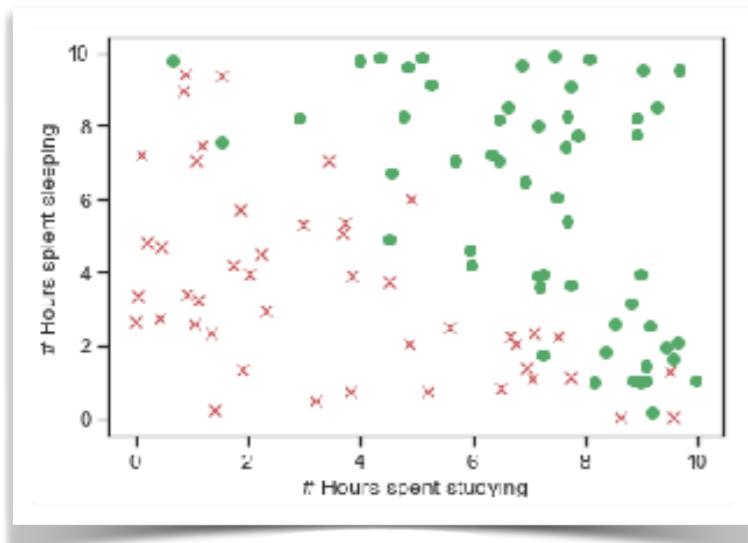
avec  $\frac{\partial L(\hat{y}^{(i)}, y^{(i)})}{\partial w_i} = x_i^{(i)} (\hat{y}^{(i)} - y^{(i)})$  et  $\frac{\partial L(\hat{y}^{(i)}, y^{(i)})}{\partial b} = (\hat{y}^{(i)} - y^{(i)})$



# Table des matières

## Apprentissage supervisé

- ✓ 1. Motivation et intérêt de l'apprentissage automatique
- ✓ 2. Classification des principaux types d'apprentissage
- ✓ 3. Définition de l'apprentissage supervisé
- ✓ 4. Méthode de la régression linéaire simple
- ✓ 5. Apprentissage par descente de gradient
- ✓ 6. Méthode de la régression linéaire multiple
- ✓ 7. Méthode de la régression logistique
- ✓ 8. Graphe de dépendance, *forward pass*, et *backward pass*



# Synthèse des notions vues

## Apprentissage automatique



*"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E"*

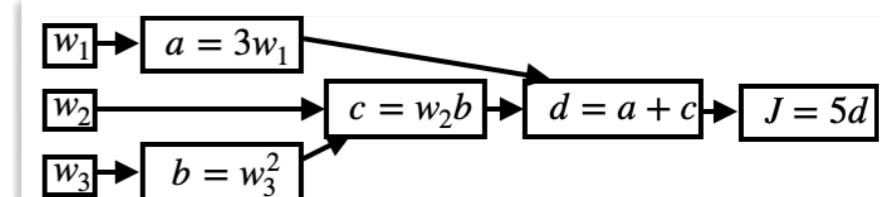
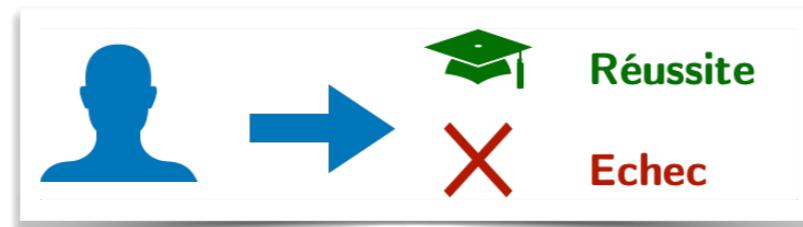
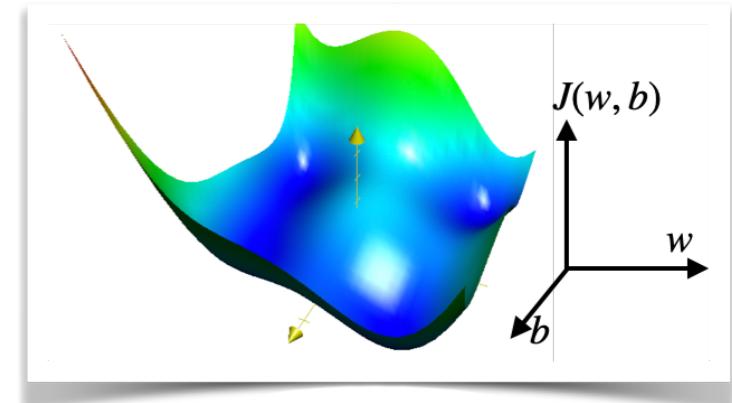
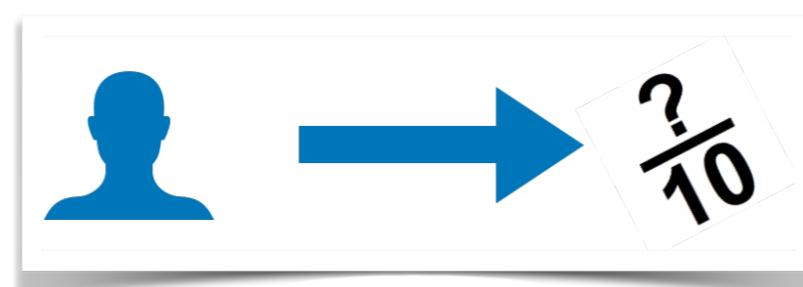
<http://www.cs.cmu.edu/afs/cs.cmu.edu/user/mitchell/ftp/mlbook.html>

- (1) L'apprentissage est dédié pour une tâche précise (T)
- (2) L'apprentissage a besoin d'expériences passées (E - données)
- (3) Besoin d'avoir un moyen d'évaluer les performances d'un modèle (P)

## Apprentissage supervisé: minimiser d'un coût entre une prédiction et les données connues

Plusieurs modèles possibles:

- (1) Régression linéaire simple
- (2) Régression linéaire multiple
- (3) Régression logistique
- (4) Et d'autres !

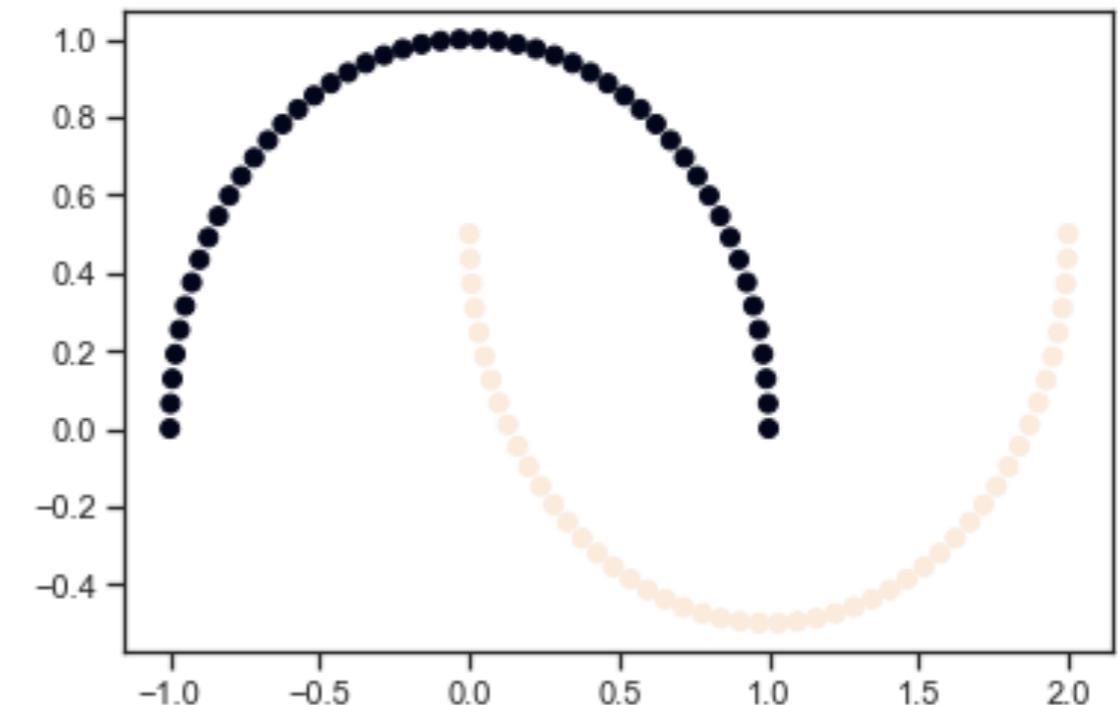
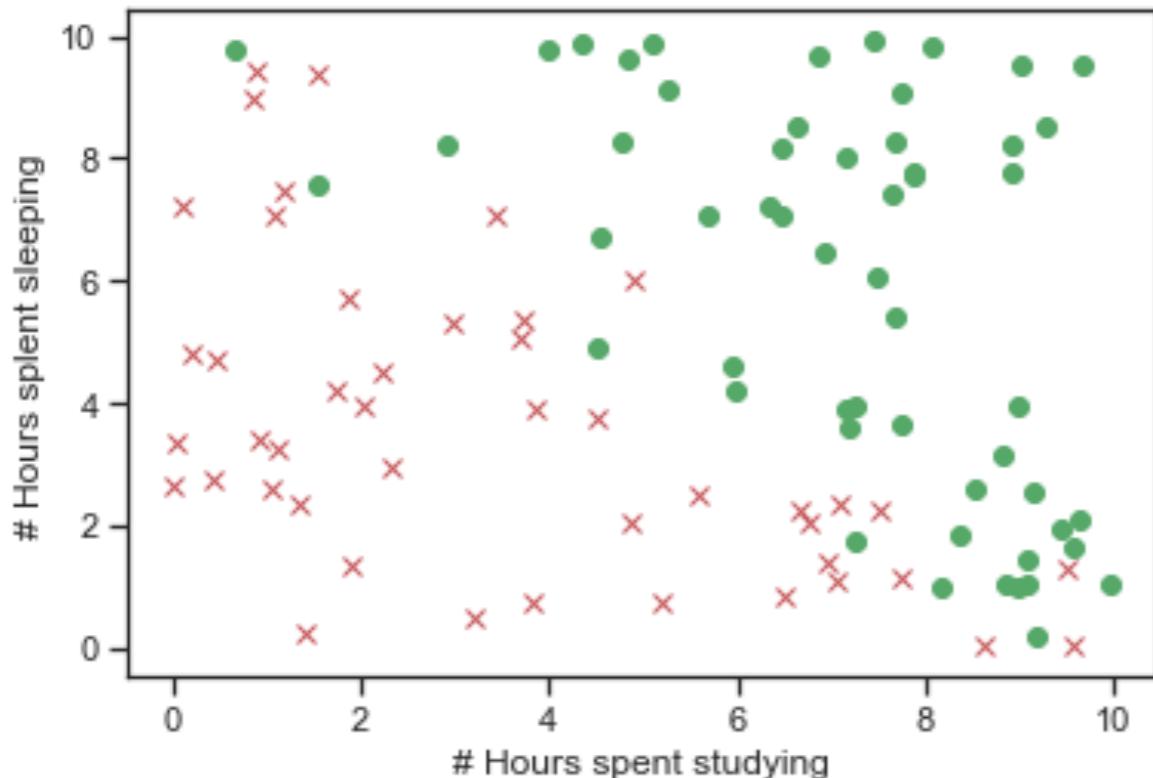


Algorithme d'apprentissage: par descente de gradient via un graphe de dépendance

# Limitation de nos méthodes de régression



Quelles sont les limitations de nos méthodes de régression actuelles ?



**Limitation:** notre prédictions sont une combinaison linéaire des variables (éventuelle sigmoidisée)

**Hypothèse sous-jacente:** les données sont linéairement séparables (ou presque)

**En pratique:** ce n'est pas souvent le cas !

**Figure de gauche:** séparation linéaire acceptable

**Figure de droite:** données non linéairement séparables

**Conclusion:** il nous faut une fonction de prédition plus expressive !



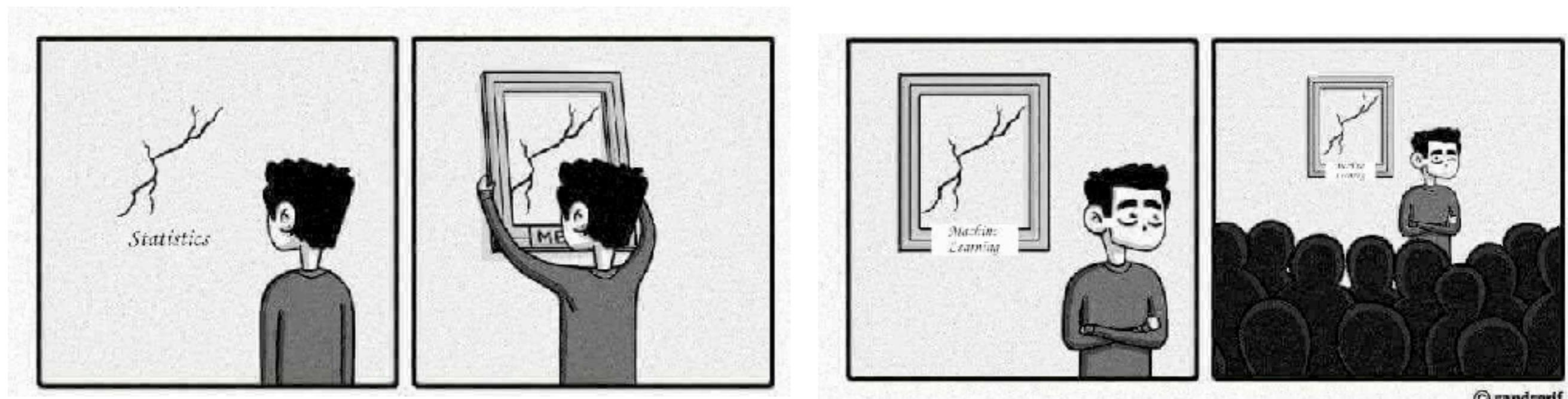
# Exemples de questions d'examen

## Théorie

1. Donner et expliquer une fonction d'écart ou de coût pour une régression donnée
2. Décrire la fonction sigmoïde, ses intérêts, et donner un cas d'utilisation
3. Expliquer les différences entre features et paramètres
4. Expliquer le principe de la descente de gradient

## Pratique

1. Effectuer une étape de descente de gradient pour une fonction simple tout en détaillant les différentes étapes (graphe de dépendance, passe en avant, et passe en arrière)



© sandserif



DALLE: A student calculating his chance to succeed at the exam, oil pastel drawing