

# Démystifier les réseaux de neurones

Cédric Beaulac

Simon Fraser University & University of Victoria

5 Mai 2021

# Introduction

- ▶ Un réseau de neurones est une fonction de prédiction utilisée en intelligence artificielle.
- ▶ Une solution aux problèmes d'apprentissage supervisé.
- ▶ On peut donc faire de la classification et prédiction en apprenant l'interaction entre certaines variables.

# Introduction

- ▶ Explication simple du problème d'apprentissage supervisé.
- ▶ Discussion sur les combinaisons linéaires.
- ▶ Comment les réseaux de neurones résolvent ce problème ?

# Plan de la présentation

Introduction

Preliminaires

Apprentissage supervisé

Statistique: régression linéaire

Un exemple

La régression linéaire

Optimisation

Réseaux de neurones

Optimisation

Applications

Conclusion

# Préliminaire

# C'est quoi le problème en statistique et intelligence artificielle ?

- ▶ On veut mieux comprendre la relation entre diverses variables en utilisant des données préalablement collectées.
- ▶ Par exemple: la relation entre un vaccin et l'immunité qu'il procure.

# Apprentissage supervisé: un problème d'actualité

- ▶ On veut apprendre une fonction  $f$  qui prend en entrée des prédicteurs  $x$  (pour prédire) et qui retourne une réponse  $y$  (ce qu'on veut prédire).
- ▶ Bien que simple en apparence c'est encore le plus gros problème en statistique.
- ▶ Représente des problèmes scientifiques, mais aussi d'entreprise, de jeux, de technologies, de santé, etc...

# Apprentissage supervisé: un problème d'actualité

- ▶ Ce que font les *téléphones intelligents*:
- ▶ Prédire quels restaurants tu vas aimer (réponse) en fonction des restaurants que tu as appréciés (prédicteurs).
- ▶ Déterminer quel vaccin est le plus efficace, quelles seront les conséquences sur l'environnement d'une mesure, le contenu d'une image, etc...



# Statistique: régression linéaire

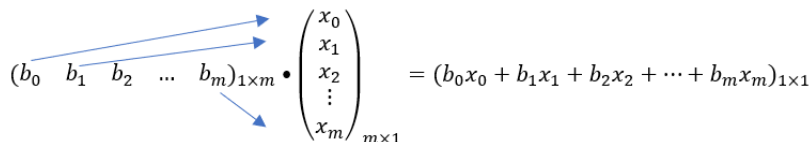
# Statistique: un exemple plate

- ▶ Exemple plate: prédire le prix d'une maison.
- ▶ Prix de la maison est la réponse.
- ▶ Comme prédicteurs nous avons: la superficie et le nombre de salles de bain.
- ▶ La fonction simple  $f$  en statistique est la combinaison linéaire.

# Statistique: un exemple plate

- ▶  $\vec{b}$  un vecteur de coefficients (dernière définition je le promets... je pense).
- ▶  $\vec{b}_{1 \times m} \bullet \vec{x}_{m \times 1} = b_0x_0 + b_1x_1 + \dots b_mx_m$  est le produit matriciel entre  $\vec{b}$  et  $\vec{x}$ , une combinaison linéaire des prédicteurs  $\vec{x}$  avec des coefficients  $\vec{b}$ .

# Statistique: produit matriciel


$$(b_0 \ b_1 \ b_2 \ \dots \ b_m)_{1 \times m} \bullet \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}_{m \times 1} = (b_0 x_0 + b_1 x_1 + b_2 x_2 + \dots + b_m x_m)_{1 \times 1}$$

**Figure:** Le produit matriciel (scalaire) tel que vous avez appris est une manière compacte et efficace pour calculer et représenter une combinaison linéaire.

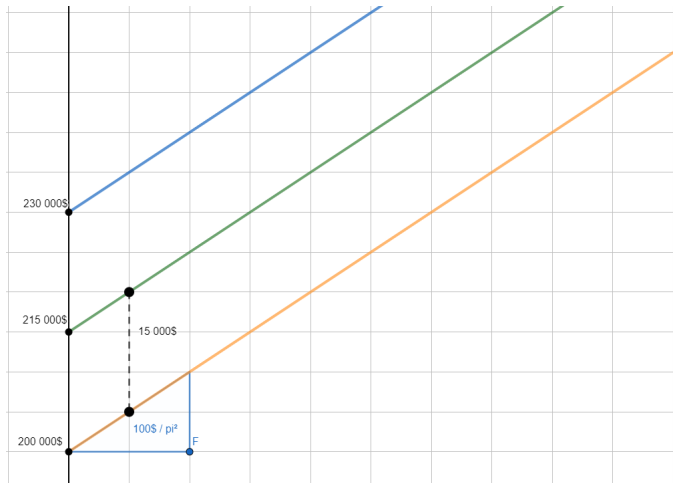
# Statistique: un exemple plate

- ▶ Par exemple, on peut dire que:

$$y = 200000 + 100 \cdot \text{nb de pied carré} + 15000 \cdot \text{nb de salle de bain} \quad (1)$$

- ▶ Donc:  $\vec{b} = [200000, 100, 15000]$  et  
 $\vec{x} = [1, \text{nb de pied carré}, \text{nb de salle de bain}]$ .

## Statistique: un exemple plate



# Statistique: un exemple plate

- ▶ Facile a *programmer*.
- ▶ La combinaison linéaire est facile à interpréter.
- ▶ Il est aussi facile de prendre la dérivée en fonction des coefficients (à voir plus tard).

# La régression linéaire

- ▶ Le problème statistique est de déterminer les coefficients  $\vec{b}$  qui ont le plus de sens.
- ▶ On veut déterminer les coefficients qui mènent à la plus petite erreur de prédiction.
- ▶ Cette technique est la *régression linéaire*.



# Régression linéaire: Optimisation

- ▶ Supposons que notre prédiction est  $\vec{b} \bullet \vec{x}$ : pour une maison avec  $x_1$  pied carré et  $x_2$  salle de bain, on prédit que  $\hat{y} = b_0 + b_1x_1 + b_2x_2$ .
- ▶ Pour établir les valeurs de  $\vec{b} = [b_0, b_1, b_2]$  on utilise des observations et on essaie de réduire la distance entre la réalité et la prédiction.
- ▶ On veut donc minimiser  $(\vec{b} \bullet \vec{x} - y)^2$ , l'erreur quadratique de prédiction.

## Régression linéaire: Optimisation

- ▶ Supposons que nous avons le prix pour  $n$  maisons, on veut choisir le vecteur  $\vec{b}$  qui minimise  $\frac{\sum_{i=1}^n (\vec{b} \bullet \vec{x}_i - y_i)^2}{n}$ .
- ▶ **Connexion à votre cours de calcul différentiel wow:**  
Quand la dérivé égale 0 on a trouvé un minimum (ou un maximum)
- ▶ On calcule donc la dérivée de  $\frac{\sum_{i=1}^n (\vec{b} \bullet \vec{x}_i - y_i)^2}{n}$  en fonction de  $\vec{b}$  et on choisit  $\vec{b}$  tel que cette dérivée égal 0.
- ▶ bingo

## Régression linéaire: Optimisation

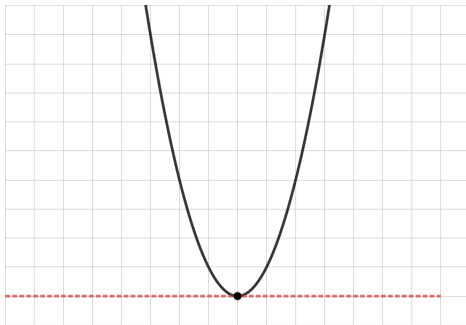


Figure: La pente de la droite rouge est de 0.

# Régression linéaire: Une solution simple pour l'apprentissage supervisé.

- ▶ Pour résumé, on veut prédire une variable  $y$  avec des prédicteurs  $x$  (disons de dimension  $m$ ).
- ▶ La régression linéaire propose de prendre une combinaison linéaire des  $m$  prédicteurs.
- ▶ À l'aide de données, on choisit les coefficients de la combinaison linéaire qui minimisent l'erreur de prédiction quadratique.
- ▶ Pour faire cela, une simple dérivée suffit.

# Réseaux de neurones

# Réseau de neurones: introduction

- ▶ La combinaison linéaire c'est bien... mais parfois c'est un peu trop simple.
- ▶ Exemples : génétiques, images, phénomène environnemental, etc...
- ▶ Le réseau de neurones est une fonction plus complexe, mais qui utilise tout de même la combinaison linéaire pour permettre l'optimisation.

# Réseaux de neurones: introduction

- ▶ Le principe est d'optimiser plusieurs combinaisons linéaires en parallèle en simultané.
- ▶ En répétant ce processus de manière séquentielle, on permet une fonction  $f$  complexe.

## Réseaux de neurones et algèbre linéaire

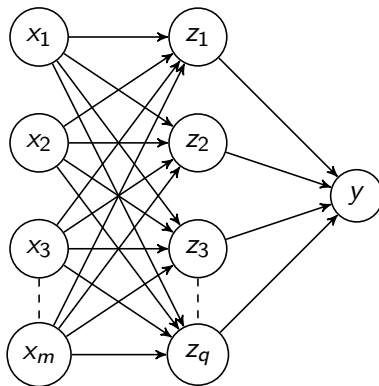


Figure: Représentation graphique d'un réseau de neurones. Le premier étage représente nos prédicteurs  $x$ . Le deuxième étage  $z$  est un *étage caché* de neurones. Chaque arrête représente un coefficient. Chaque élément  $z_i$  est une différente combinaison linéaire.



# Réseaux de neurones et algèbre linéaire

- ▶ La figure précédente est une représentation standard d'un réseau de neurones.
- ▶ C'est sexy, impressionnant et intimidant pour rien; tape-à-l'oeil.
- ▶ Il est plus simple mathématiquement de la représenter le système à l'aide de produits matriciels.

# Réseaux de neurones et algèbre linéaire

- La meilleure manière de représenter et de calculer ces combinaisons linéaires est d'utiliser le produit matriciel.

$$\mathbf{B}_1 \bullet \vec{x} = \vec{z}$$

où  $\mathbf{B}_1$  est une matrice  $q \times m$ ,  $\vec{x}$  un vecteur  $m \times 1$  et donc  $\vec{z}$  est de taille  $q \times 1$ .

- $\mathbf{B}_1$  est une matrice de coefficients que nous devons fixer.

# Réseaux de neurones et algèbre linéaire

$$\begin{pmatrix} b_{00} & b_{01} & b_{02} & \dots & b_{0m} \\ b_{10} & b_{11} & b_{12} & \dots & b_{1m} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ b_{q0} & b_{q1} & b_{q2} & \dots & b_{qm} \end{pmatrix}_{q \times m} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}_{m \times 1} = \begin{pmatrix} b_{00}x_0 + b_{01}x_1 + b_{02}x_2 + \dots + b_{0m}x_m \\ b_{10}x_0 + b_{11}x_1 + b_{12}x_2 + \dots + b_{1m}x_m \\ b_{20}x_0 + b_{21}x_1 + b_{22}x_2 + \dots + b_{2m}x_m \\ \vdots \\ b_{q0}x_0 + b_{q1}x_1 + b_{q2}x_2 + \dots + b_{qm}x_m \end{pmatrix}_{q \times 1} = \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ \vdots \\ z_q \end{pmatrix}_{q \times 1}$$

**Figure:** Le produit matriciel tel que vous avez appris est une manière compacte et efficace pour calculer et représenter plusieurs combinaisons linéaires à la fois.

# Réseaux de neurones et algèbre linéaire

- ▶ Un réseau de neurones peut avoir plusieurs étages cachés.
- ▶ Finalement, nous complétons avec une dernière combinaison linéaire (de  $z$ ) pour obtenir la réponse :

$$\mathbf{B}_2 \bullet \mathbf{B}_1 \bullet \vec{x} = y$$

où  $\mathbf{B}_2$  est une matrice de taille  $1 \times q$

# Réseaux de neurones et algèbre linéaire

- Un fin savant en algèbre linéaire pourrait remarquer un problème:

$$\mathbf{B}_2 \bullet \mathbf{B}_1 = \vec{b}_{1 \times m}$$

où  $\vec{b}$  est un vecteur  $1 \times m$ , une simple combinaison linéaire!

# Réseaux de neurones: définition complète

- Le truc pour complexifier ces simples combinaisons linéaires est d'appliquer une fonction non linéaire ( $\sigma$ ) sur les éléments du vecteur caché  $\vec{z}$ :

$$\mathbf{B}_2 \bullet \sigma(\mathbf{B}_1 \bullet \vec{x}) = \mathbf{B}_2 \bullet \sigma(\vec{z}) = y$$

par exemple  $\sigma(j) = \frac{1}{1+e^{-j}}$

# Réseaux de neurones: définition complète

- ▶ Pour résumé, un réseau de neurones est une fonction qui applique séquentiellement des produits matricielles et des fonctions non linéaires aux variables en entrée.
- ▶ Permet donc de générer des fonctions  $f$  extrêmement complexes et flexibles.
- ▶ Nous reste plus qu'à déterminer les matrices de coefficients idéales!

## Réseaux de neurones: Optimisation

- ▶ L'idée reste la même.
- ▶ Avec un jeu de données, on choisit les matrices de coefficients **B** qui minimisent une erreur de prédiction.
- ▶ Cependant ici, on ne peut pas simplement calculer la dérivée et mettre cette dérivée égal à 0.



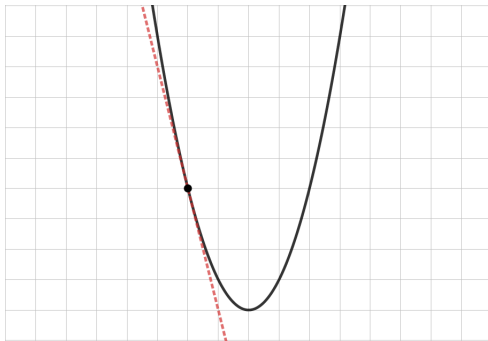
# Réseaux de neurones: Optimisation

- ▶ C'est le moment où la théorie devient compliqué, donc allons-y en surface.
- ▶ On utilise l'algorithme du gradient.
  1. Fix des valeurs initiales aléatoires pour **B**.
  2. On calcule la dérivée partielle en fonction de chaque élément des matrices **B**.
  3. On modifie les valeurs de **B** dans la direction inverse du gradient pour se diriger vers un point où la dérivée est de 0.

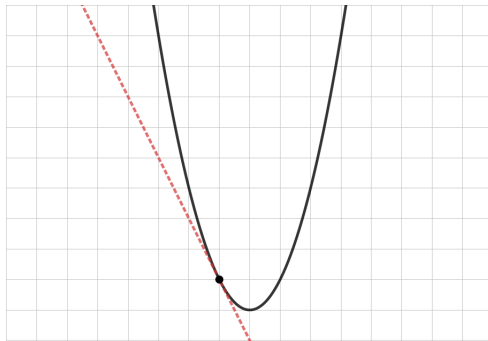
# Algorithme du gradient



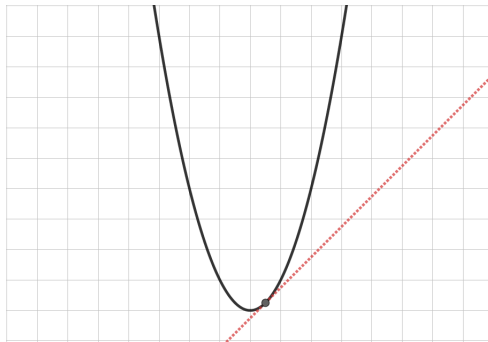
# Algorithme du gradient



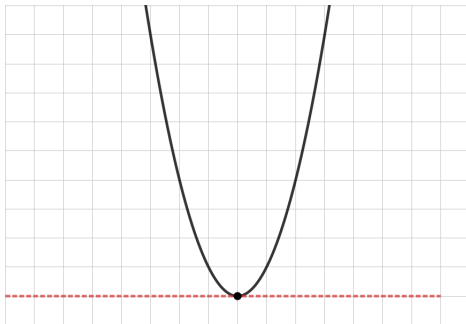
# Algorithme du gradient



# Algorithme du gradient



# Algorithme du gradient



# Réseaux de neurones: Optimisation

- ▶ On a donc besoin de pouvoir calculer le gradient!
- ▶ Nos combinaisons linéaires permettent de le faire facilement.
- ▶ Suffit de choisir une fonction  $\sigma$  différentiable
- ▶ et d'utiliser les règles de dérivée en chaine (que vous avez appris en calcul différentiel ? Peut-être ?).

## Une application: l'analyse d'images

- ▶ En analyse d'image, les réseaux de neurones sont centraux.
- ▶ Les prédicteurs sont les pixels et la réponse est le contenu de l'image par exemple.
- ▶ Utilisé dans les voitures intelligentes, en reconnaissance faciale, lecture de texte écrit à main.



# Une application: l'analyse d'images

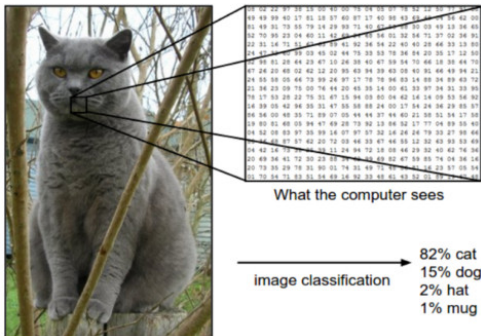
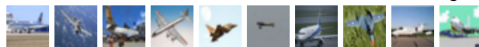


Figure: Analyse d'images

## Une application: l'analyse d'images

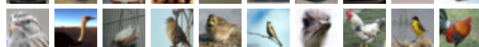
**airplane**



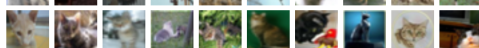
**automobile**



**bird**



**cat**



**deer**



**dog**



Figure: Données CIFAR-10

## Une application: la génération d'image

- ▶ On parle ici d'un de mes projets de travail.
- ▶ On a une base de données de chiffre écrit à main.
- ▶ On apprenant à prédire le chiffre et l'auteur, on peut générer des images de chiffres qui imitent les vrais chiffres.

## Results : controlled image generation

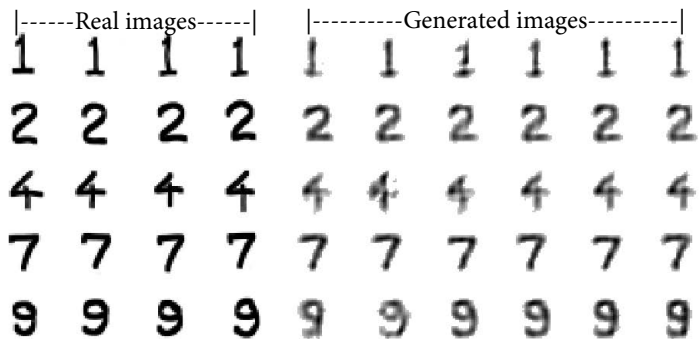


Figure: Example with ID12.

## Results : controlled image generation

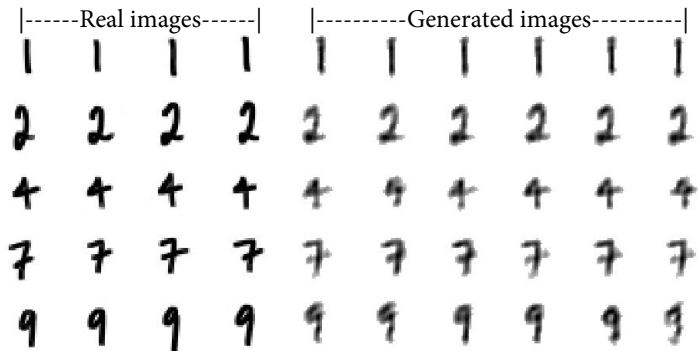


Figure: Example with ID14.

## Results : controlled image generation

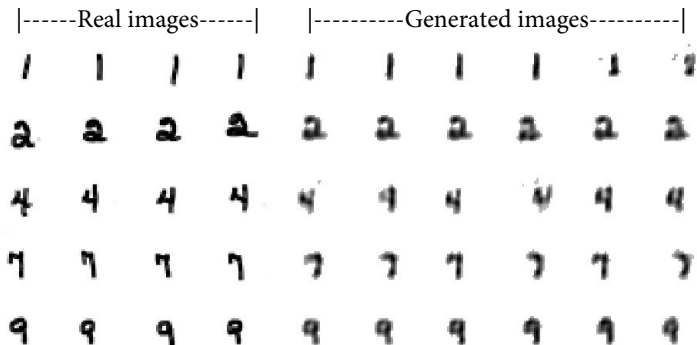


Figure: Example with ID29.

## Results : controlled image generation

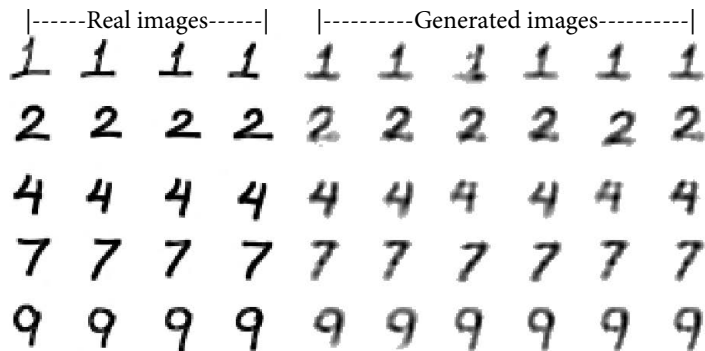


Figure: Example with ID70.

# Results : controlled image generation



## Plus d'applications: l'analyse d'image

- ▶ Voiture intelligente (*self-driving*).
- ▶ Reconnaissance faciale. (Pour identification, mais aussi tous les filtres et *fake green screen*)
- ▶ Imagerie médicale: gros succès et gros impact.

## Filtres

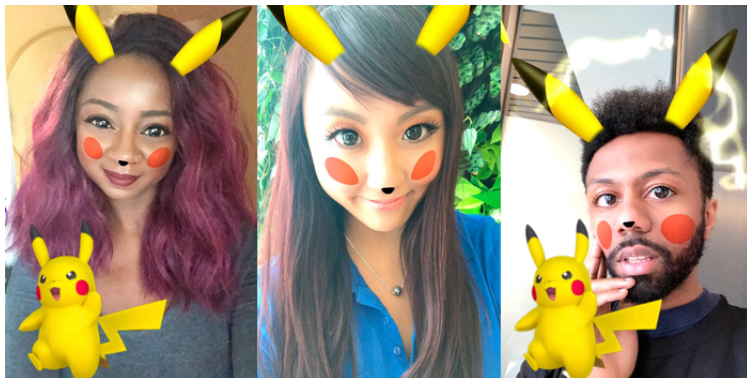


Figure: Les filtres doivent identifier où sont vos yeux, joues, etc...

## Imagerie médicale

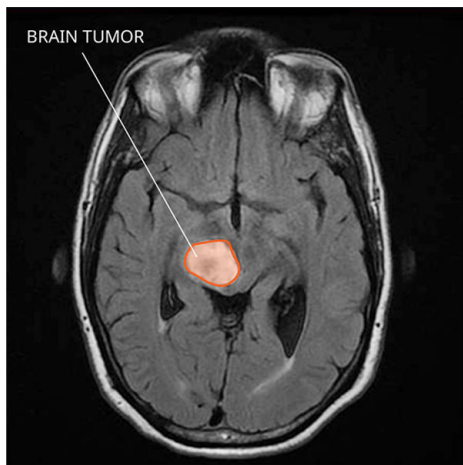


Figure: Identifier si une tumeur est bénigne ou maligne.

## Plus d'applications: les systèmes de recommandations

- ▶ Recommandation de contenu: Uber eats, tiktok, youtube, spotify, Netflix, etc..
- ▶ Réseaux sociaux: facebook, tinder, twitter, etc..

## Plus d'applications: Interpretations de langage

- ▶ Interprétation de texte écrit à la main
- ▶ Reconnaissance vocale

# Conclusion

# Discussion

- ▶ L'algèbre linéaire c'est fort.
- ▶ Plus sérieusement, de la simple algèbre linéaire et la dérivée sont la fondation des modèles les plus avancés d'intelligence artificielle.

## De grands problèmes

- ▶ Comment décide-t-on du nombre de neurones et d'étages ?
- ▶ La surface d'optimisation est chaotique: pour une nouvelle initialisation et algorithme du gradient, on obtient de nouveaux résultats.
- ▶ Le data set est très important: peut mener à des problèmes d'éthique et peut faire perdurer certaines iniquités.



# Des questions ?