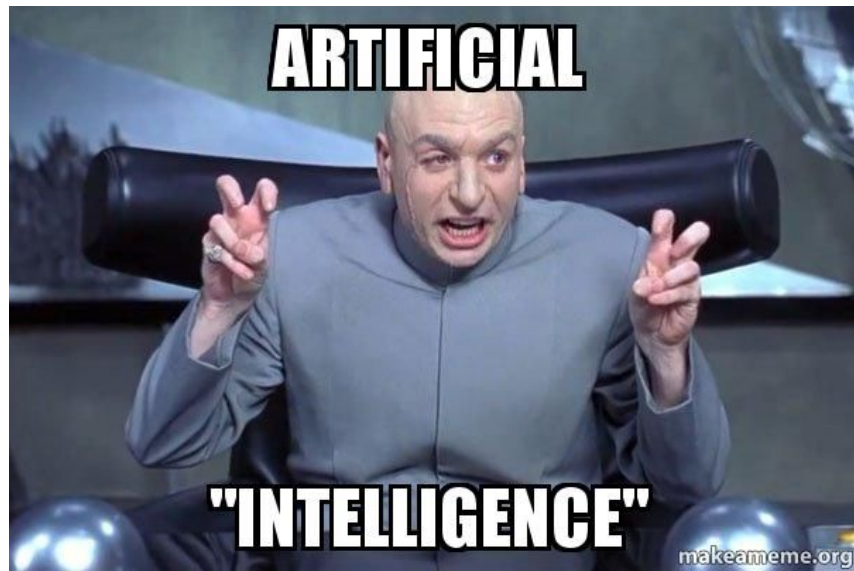


Intro

Intelligence Artificielle

Julien - SummerIA 2017



- Intelligence artificielle
 - Machine Learning
 - Deep Learning

Intelligence artificielle

- machine capable de simuler de l'intelligence
 - tâches complexes, propre à l'Homme



IA forte vs faible

IA faible :

- Approche “ingénieur”
- UNE tâche précise
- Autonomie, assistance, simuler
- Machine non sensible
- Ex : Siri
- **Partout maintenant**

IA forte (IA générale)

- Vision “Science-fiction”
- S’adapte à une nouvelle tâche
- Reasonner, conscience
- Machine sensible
- N’existe pas !?



“Good old-fashioned” artificial intelligence

Un cas très simple d’IA :

- **“Estimer le prix d’une maison à partir de ses informations”**
- Information (manquante?) : superficie int/ext? Nb de pièces ? ...

Interviewer un expert (un agent immobilier)

- comprendre son processus d’estimation
- le traduire en programme (“si ça, ... alors ça, ..., sinon ça ...”)
- évaluer ce programme sur quelques exemples

Intelligence artificielle par Machine Learning

Machine Learning : algo/programme/machine auto-apprenante
(ou apprentissage automatique)

- On entraîne une machine/algo à faire une tâche précise
- Entraînement à partir de données d'apprentissage
- Modification des paramètres (variables) de l'algo pendant l'apprentissage
- Objectif : **généralisation** (ne pas se tromper sur de nouvelles données)

Cas de l'estimation du prix d'une maison:

- choisir un algorithme de ML (en fct° du type de problème)
- lui montrer des exemples (informations + prix)
- on évalue ses performances sur de nouvelles données

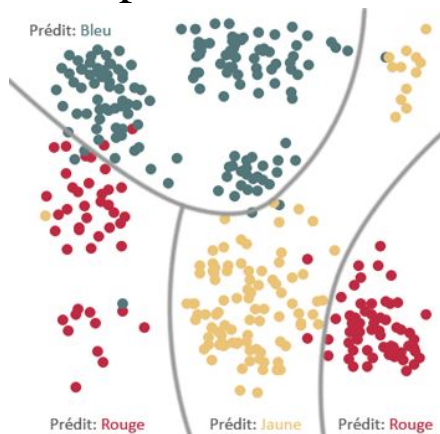
Apprentissage supervisé vs non-supervisé

Apprentissage supervisé :

- On a la “bonne réponse” (label)

Ex :

- classification d'images
- prédire le prix d'une maison



Apprentissage non-supervisé:

- Il n'y a pas de “bonne réponse”

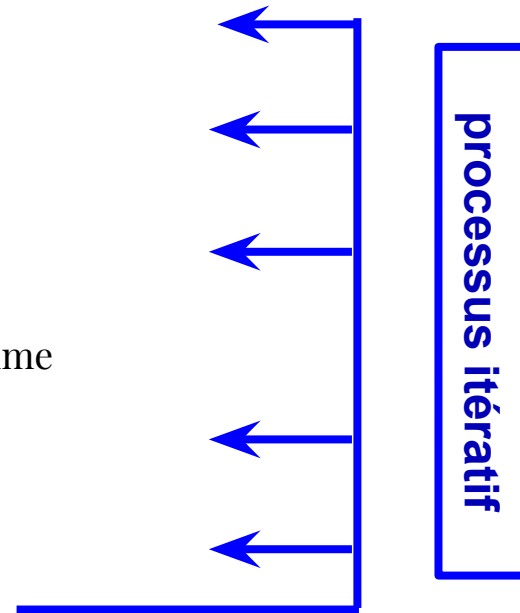
Ex :

- segmentation de clients



Les étapes d'un projet de Machine Learning

- Récupérer les données et les comprendre
 - Quelle tâche ? Quel métier ? ...
- Analyser les données
 - Données manquantes ? données aberrantes ? ...
- Préparer les données : “feature engineering”
 - extraire l'information pertinente
 - créer des variables compréhensibles pour l'algorithme
- Choisir un modèle/algorithme
 - en fonction de la tâche et des données
- Lancer l'entraînement
- Evaluer son modèle et ?



Feature Engineering

Un exemple : dummification / one-hot encoding

sample	...	car_type	...	more_50000
id_001	x	familly_car	x	0
id_002	x	sport_car	x	1
id_003	x	offroad_car	x	1
id_004	x	None	x	0

En Machine Learning :

Bon résultat = **bon feature engineering** + bon algo

Variable catégorielle : “car_type”

- 5 catégories
- None -> (1,0,0,0,0)
- familly_car -> (0,1,0,0,0)
- sport_car -> (0,0,1,0,0)
- offroad_car -> (0,0,0,1,0)
- minivan_car -> (0,0,0,0,1)

Importance de données - Big Data

Plus de données = meilleur modèle !

Business model : revente de données

- application/service gratuit » rentable avec la revente de données
- ex : Waze (appli GPS)

Importance de l'IA/ML au quotidien

Beaucoup de plateformes :






- cours en ligne (MOOC)
- compétitions
- forums
- blogs ...

ActiveAllEntered

kaggle.com

All Categories

Search competitions

	Passenger Screening Algorithm Challenge Improve the accuracy of the Department of Homeland Security's threat recognition algorithms <i>Featured</i> · 4 months to go	\$1,500,000 198 teams
	Zillow Prize: Zillow's Home Value Prediction (Zestimate) Can you improve the algorithm that changed the world of real estate? <i>Featured</i> · 5 months to go	\$1,200,000 2,280 teams
	Carvana Image Masking Challenge Automatically identify the boundaries of the car in an image <i>Featured</i> · a month to go	\$25,000 389 teams
	Web Traffic Time Series Forecasting Forecast future traffic to Wikipedia pages <i>Research</i> · 21 days to go	\$25,000 695 teams
	Personalized Medicine: Redefining Cancer Treatment Predict the effect of Genetic Variants to enable Personalized Medicine <i>Research</i> · a month to go	\$15,000 845 teams

Petits rappels

Data science : science/art du traitement de données

Métier : “Data Scientist”

Éléments pour devenir un bon data “scientist” :

- mathématiques
 - algèbre, analyse (fonction), probabilité et statistique,
- informatique
 - optimisation et complexité
 - Python et des librairies spécifiques (scikit-learn, **keras**, tensorflow, ...)

Théorie

ML : Apprentissage supervisé par descente du gradient

Application à la classification d'images

Appr. supervisé par descente du gradient

- **Une façon simple** de mettre à jour les paramètres d'un algo
- Notions
 - optimisation (minimisation d'une fonction)
 - fonctions : logarithme, exponentielle
 - probabilités
 - algèbre (matrices et vecteurs)

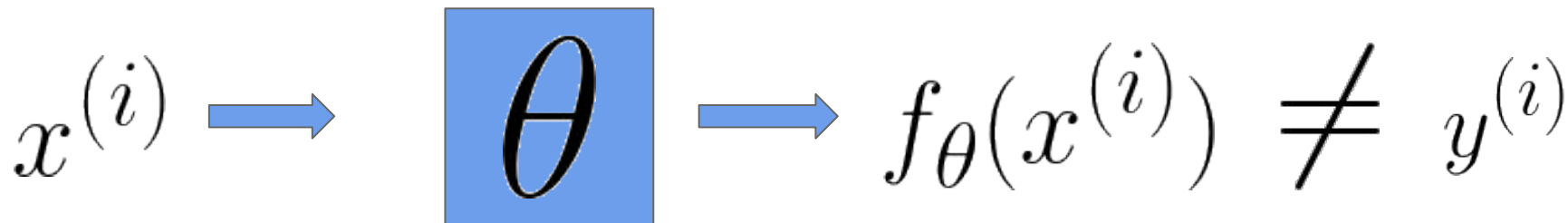
Il existe beaucoup d'algorithmes qui s'entraînent différemment !

Notations de Machine Learning

paire (donnée, label) : $(x^{(i)}, y^{(i)})$

paramètres entraînables : θ

fonction de prédiction : $f_{\theta}(x^{(i)})$



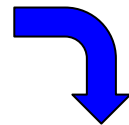
Cas de la classification d'images

$$x^{(i)} \rightarrow \text{image of a yellow bird} \rightarrow \mathbb{R}^{w \times h}$$

$$y^{(i)} \rightarrow \text{bird} \rightarrow 2 \rightarrow \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \dots \end{pmatrix}$$

$$f_{\theta}(x^{(i)}) \rightarrow \begin{pmatrix} .05 \\ .02 \\ .91 \\ .00 \\ \dots \end{pmatrix}$$

dataset "CIFAR10"



airplane



automobile



bird



cat



deer



dog



frog



horse



ship



truck



Cas de la classification d'images

$$x^{(i)} \rightarrow \text{image} \rightarrow \mathbb{R}^{w \times h}$$

image :
- vecteur
- ou tableau (matrice)
- nombres réels

$$y^{(i)} \rightarrow \text{bird} \rightarrow 2 \rightarrow \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \dots \end{pmatrix}$$

$$f_{\theta}(x^{(i)}) \rightarrow \begin{pmatrix} .05 \\ .02 \\ .91 \\ .00 \\ \dots \end{pmatrix}$$

bird

cat

deer

dog

frog

horse

ship

truck



Cas de la classification d'images

$$x^{(i)} \rightarrow \text{image of a yellow bird} \rightarrow \mathbb{R}^{w \times h}$$

$$y^{(i)} \rightarrow \text{bird} \rightarrow 2 \rightarrow \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix}$$

$$f_{\theta}(x^{(i)}) \rightarrow \begin{pmatrix} .05 \\ .02 \\ .91 \\ .00 \\ \vdots \end{pmatrix}$$

airplane



automobile



bird



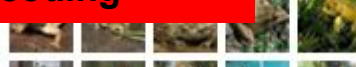
cat



label :

- un nombre ou identifiant (2)
- un vecteur de 0 et 1
- 'one-hot encoding'

dog



horse



ship



truck



Cas de la classification d'images

$$x^{(i)} \rightarrow \text{image of a yellow bird} \rightarrow \mathbb{R}^{w \times h}$$

$$y^{(i)} \rightarrow \text{bird} \rightarrow 2 \rightarrow \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$f_{\theta}(x^{(i)}) \rightarrow \begin{pmatrix} .05 \\ .02 \\ .91 \\ .00 \\ \dots \end{pmatrix}$$

prédiction :

- vecteur d'appartenance à chacune des classes
- valeurs en 0 et 1
- somme = 1

airplane

automobile

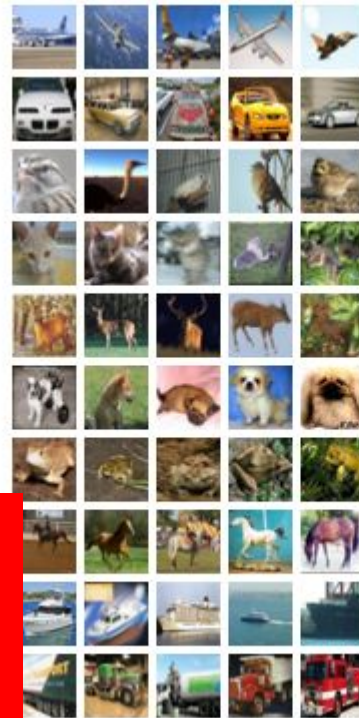
bird

cat

deer

dog

frog



Cas de la classification d'images : fonction d'erreur

$$f_{\theta}(x^{(i)}) \rightarrow \begin{cases} \mathbb{P}(Y = truck \mid x^{(i)}, \theta) & 0.12 \\ \mathbb{P}(Y = bird \mid x^{(i)}, \theta) & 0.81 \\ \circ & \cdot \\ \circ & \cdot \\ \circ & \cdot \\ \mathbb{P}(Y = plane \mid x^{(i)}, \theta) & 0.05 \end{cases} \neq \begin{cases} 0.0 \\ 1.0 \\ \cdot \\ \cdot \\ \cdot \\ 0.0 \end{cases}$$

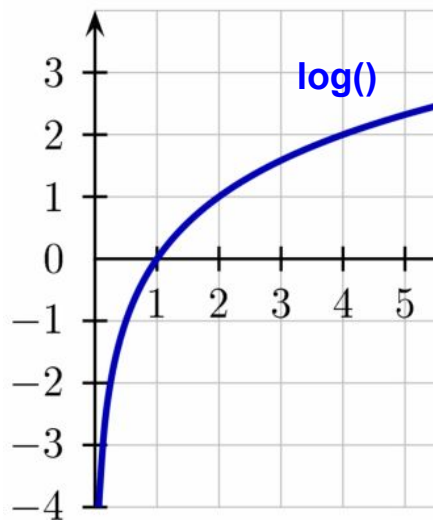
$y^{(i)}$
 \downarrow
 $bird$

$$E = -\log (P(Y = y^{(i)} \mid x^{(i)}, \theta))$$

Cas de la classification d'images : fonction d'erreur

Erreur : Entropie croisée (cross entropy)

- mesure l'erreur de prédiction
- “distance” entre la vérité et la prédiction



$$E = -\log(0.81) \\ = 0.1165$$

0.12
0.81
.
.
.
.
0.05

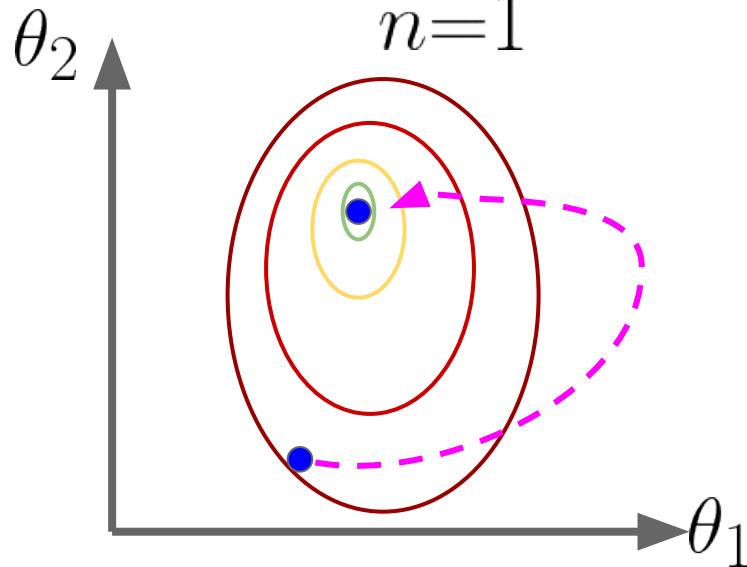
\neq

$y^{(i)}$
↓
0.0
1.0
.
.
.
.
0.0
bird

$$E = -\log (P(Y = y^{(i)} \mid x^{(i)}, \theta))$$

Minimiser la fonction d'erreur

$$E(X, Y, \theta) = \frac{1}{N} \sum_{n=1}^N E(x^{(i)}, y^{(i)}, \theta)$$

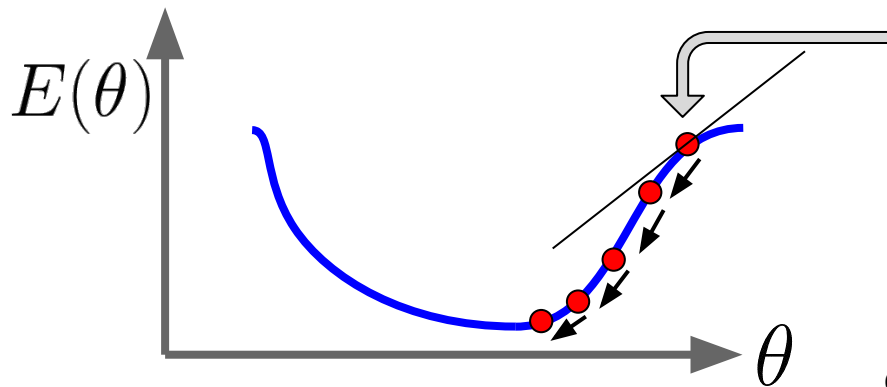


cas d'une fonction 2D (2 paramètres)

Minimisation par descente du gradient

$$\theta_{t+1} = \theta_t - \lambda \frac{\partial E(X, Y, \theta)}{\partial \theta}$$

- méthode itérative (à répéter un certain nombre de fois)
- mise à jour des paramètres de façon à minimizer la fonction
- λ : pas d'apprentissage (hyper-paramètre)
- ne fonctionne que si la fonction d'erreur est dérivable



- partir d'une valeur initiale (aléatoire)
- calculer la dérivée en ce point
- mettre à jour la valeur du paramètre

cas d'une fonction 1D (1 paramètre)

Problème des 'gros' datasets

Si (X, Y) trop grand : trop difficile à calculer (temps, mémoire, ...)

Solutions :

- Erreur sur 1 exemple : descente du gradient stochastique (SGD)

$$\theta_{t+1} = \theta_t - \lambda \frac{\partial E(X^{(i)}, Y^{(i)}, \theta)}{\partial \theta}$$

Problème des 'gros' datasets

Si trop grand : trop difficile à calculer (temps, mémoire, ...)

Solutions :

- Erreur sur 1 exemple : descente du gradient stochastique (SGD)

$$\theta_{t+1} = \theta_t - \lambda \frac{\partial E(X^{(i)}, Y^{(i)}, \theta)}{\partial \theta}$$

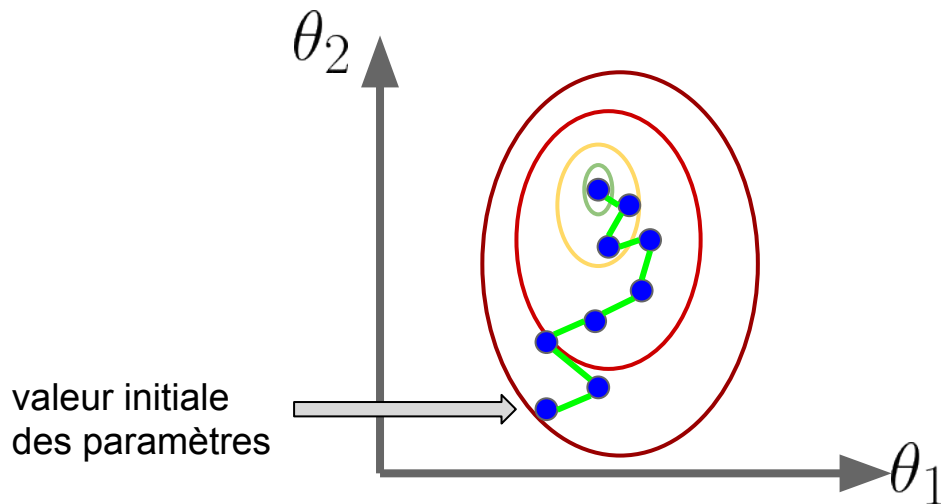
- dérivées instables

- Erreur sur un 'batch' d'exemple : SGD par mini-batch

$$\theta_{t+1} = \theta_t - \lambda \frac{\partial E(X^{[batch]}, Y^{[batch]}, \theta)}{\partial \theta}$$

Pseudo-code pour “SGD par mini-batch”

```
for epoch in range(nb_epoch):  
    for batch in range(nb_batch):  
        train_model(model, X = X[batch*batch_size : (batch+1)*batch_size],  
                      y = y[batch*batch_size : (batch+1)*batch_size],  
                      learning_rate = lr)
```

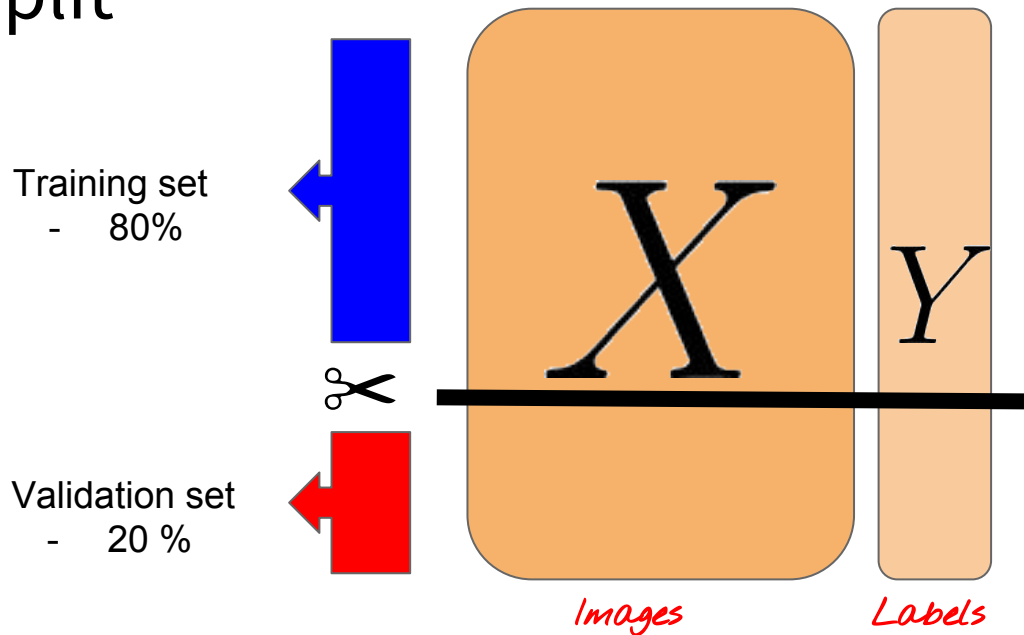


Hyper-paramètres :

- pas d'apprentissage
- taille des batches
- nombre d'epoch (itérations)

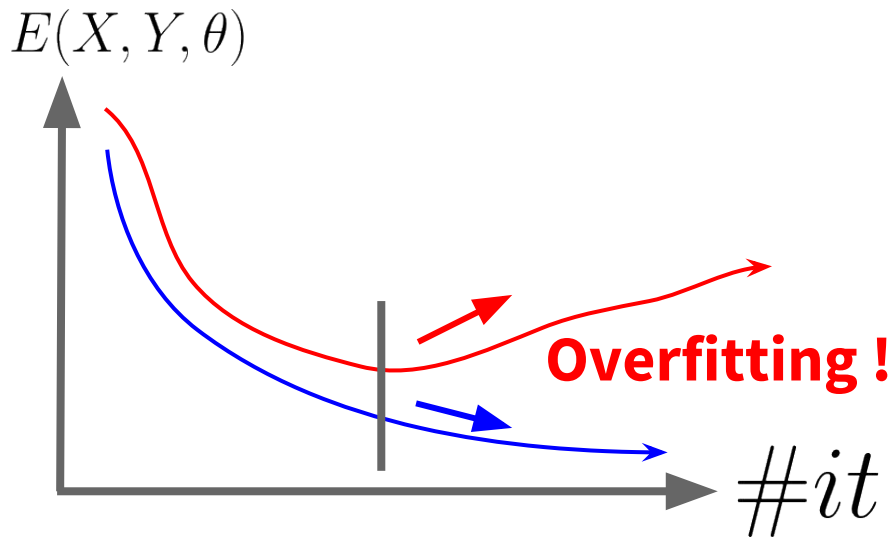
Courbes d'apprentissage et Overfitting

Train/Test split



Courbes d'apprentissage et Overfitting

- Découpage aléatoire et équilibré
- Permet de suivre l'entraînement
- Permet de détecter l'overfitting (ou sur-apprentissage)



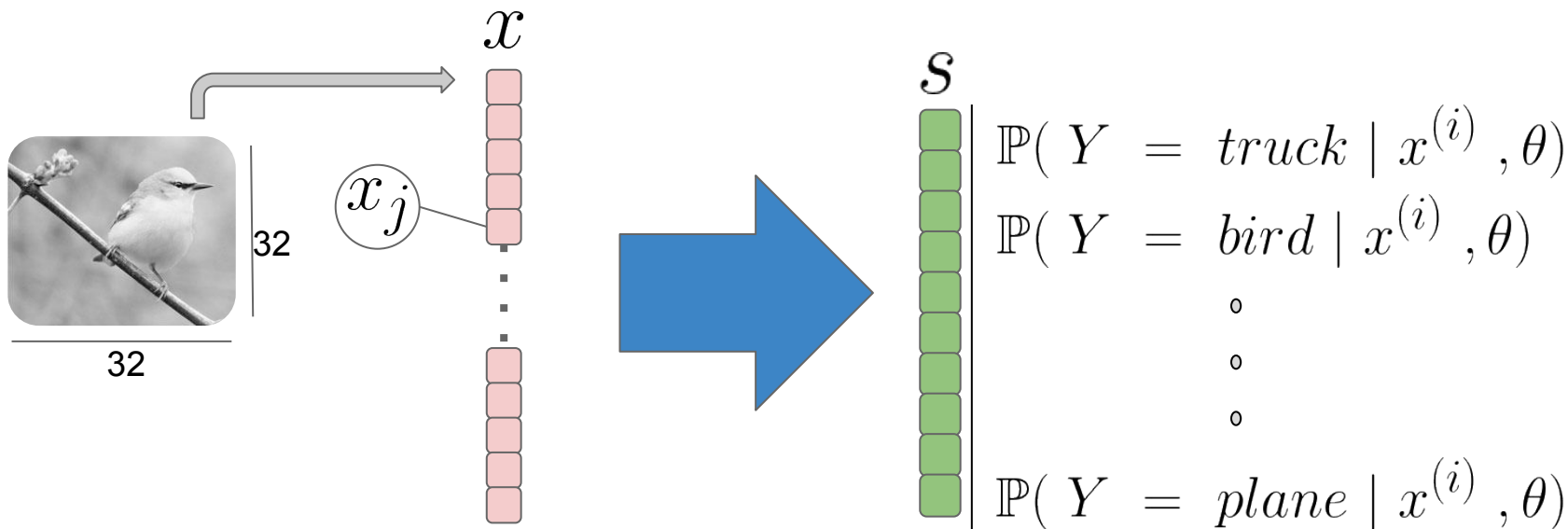
Premier modèle : Régression logistique

Ce que l'on veut ? - Changement de représentation

- passer d'une image, représentée sous la forme d'un vecteur
- à un vecteur de probabilités
- via la fonction de transfert



Passer d'un vecteur à un vecteur : **multiplication matricielle**



Multiplication matrice x vecteur

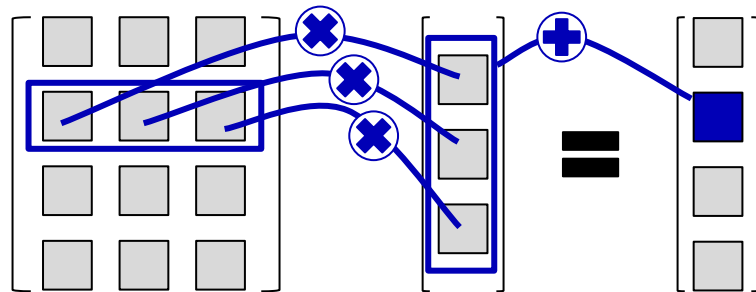
$$x \xrightarrow{W} y$$

$$\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \\ w_{3,1} & w_{3,2} & w_{3,3} \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix}$$

$$W \cdot x = y$$

$$x \in \mathbb{R}^3$$
$$y \in \mathbb{R}^4$$
$$W \in \mathbb{R}^{4,3}$$

$$y_j = \sum_{i=1}^3 w_{j,i} \cdot x_i$$



Régression Logistique (1/2)

$$s = \text{softmax}(Wx + b)$$



32

32

x_j

x

(1024,)

W
(10, 1024)

Matrice de poids

Wx

(10,)

+

b

(10,)

Biais (vecteur)

s

(10,)

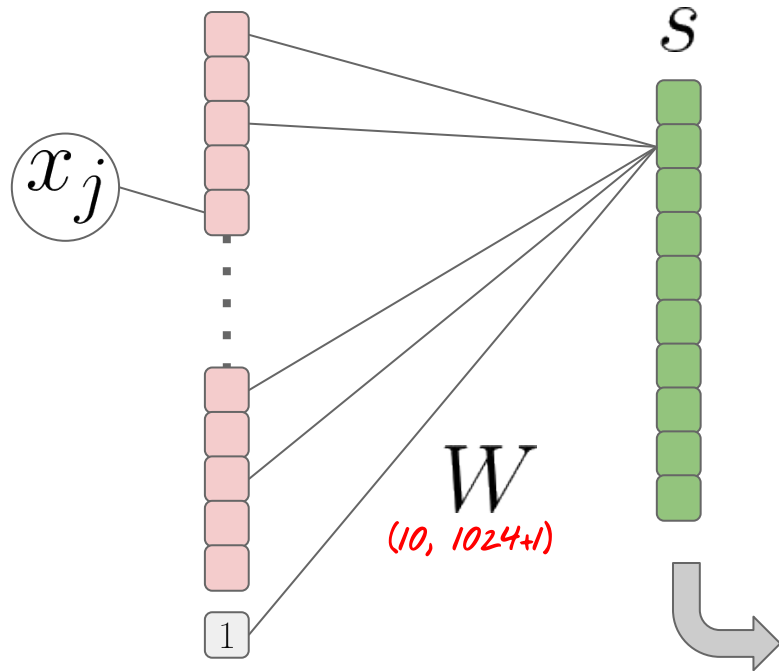
$$\text{softmax}(h_i) = \frac{e^{h_i}}{\sum_{j=1}^{10} e^{h_j}}$$

$$h = Wx + b$$

$$\theta = \{W, b\}$$

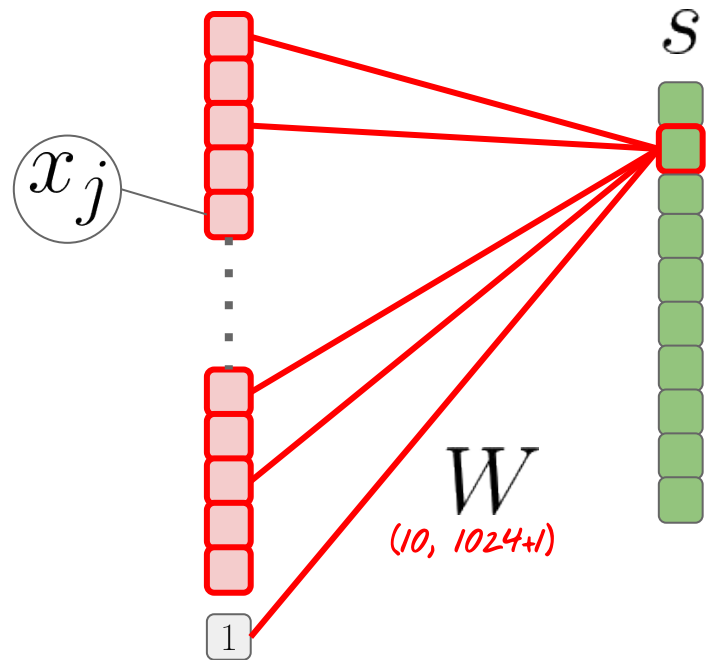
Régression Logistique (2/2)

- Vision simplifiée
- Fait apparaître la **notion de neurones**
- **10,250 paramètres entraîner**



$$s_i = \text{softmax}\left(\sum_{j=1}^{1024+1} w_{i,j} x_j\right)$$

Régression Logistique (2/2)



- Vision simplifiée
- Fait apparaître la **notion de neurones**



- Réseau de neurones
- 1 couche d'entrée (pixels)
- 1 couche de sortie (probabilités)
- **pas de couches entre**

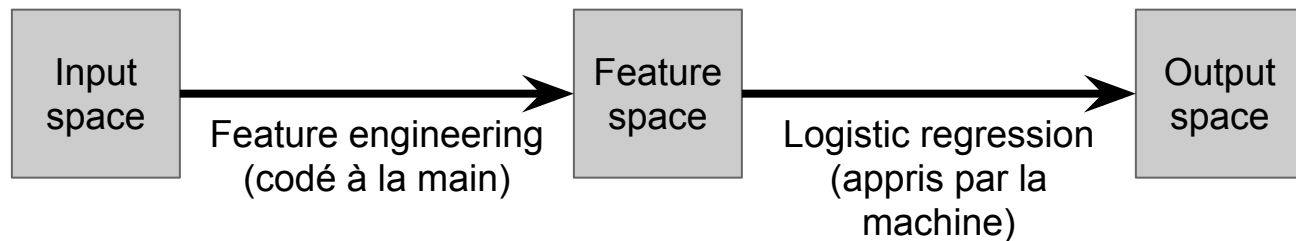
$$s_i = \text{softmax}\left(\sum_{j=1}^{1024+1} w_{i,j} x_j\right)$$

Deep Learning

Machine Learning Pipeline

Traditional Machine Learning pipeline :

- Faire du “feature engineering” pour extraire l’information pertinente
- Utiliser un algo de Machine Learning (régression logistique, SVMs, arbres de décision, ...)



Pour les images : HOG features, SIFT methods, Histograms, LBP features,

Problème des images ?

Image RGB : 32x32 pixels (x3 car RGB) \Rightarrow 3072 pixels

- si codage uint8 : 256 valeurs possible par pixel !

$\Rightarrow 3072^{256}$ images possibles ! (1000 x plus que le nb d'atomes dans l'univers)

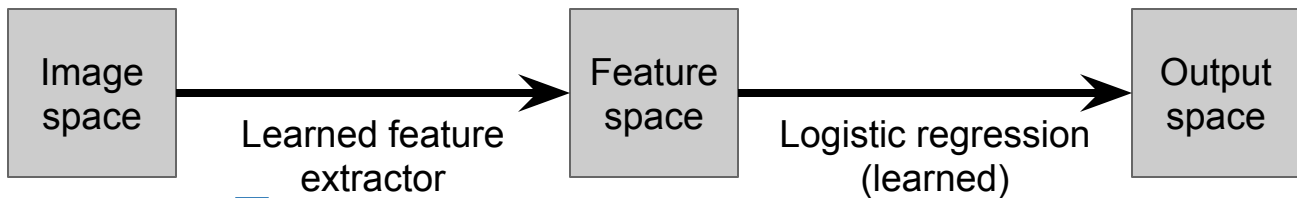
Comment extraire l'information pertinente des images ?

- ex : classification chien vs chat ?



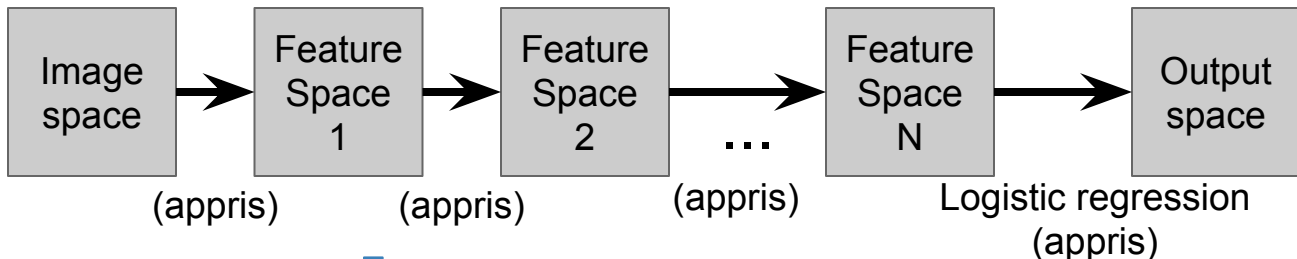
Apprendre à extraire l'information pertinente ?

Representation Learning



- Apprendre une nouvelle représentation des données
- Ex : PCA (Principal Component Analysis)

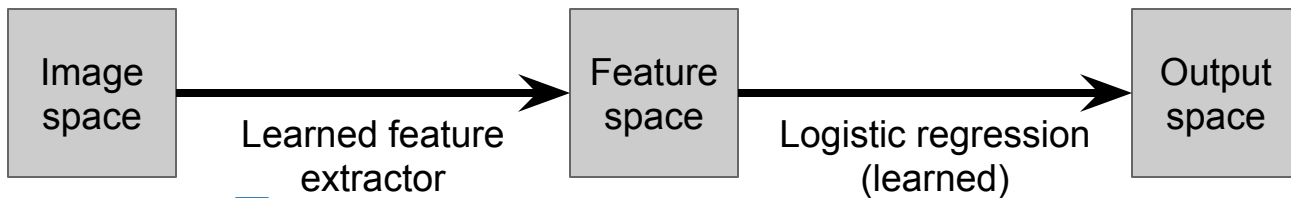
Deep Representation Learning



- Apprendre une hiérarchie de représentations
- Se fait avec des réseaux de neurones artificiels

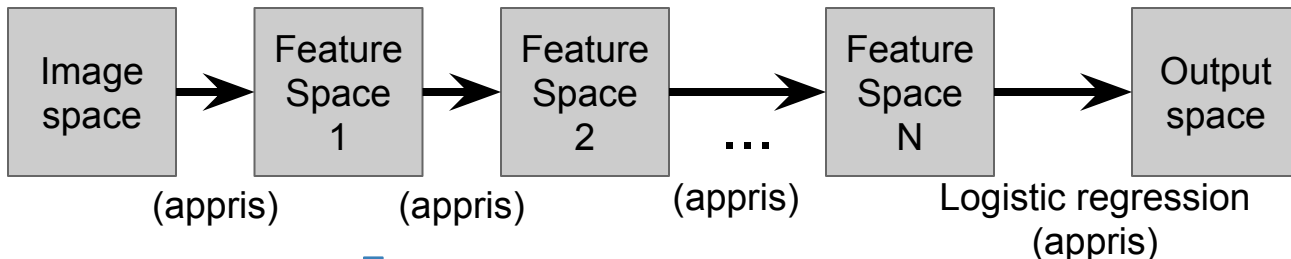
Apprendre à extraire l'information pertinente ?

Representation Learning



- Apprendre une nouvelle représentation des données
- Ex : PCA (Principal Component Analysis)

Deep Representation Learning



- Apprendre une hiérarchie de représentations
- Se fait avec des réseaux de neurones artificiels

Deep Learning (apprentissage profond)

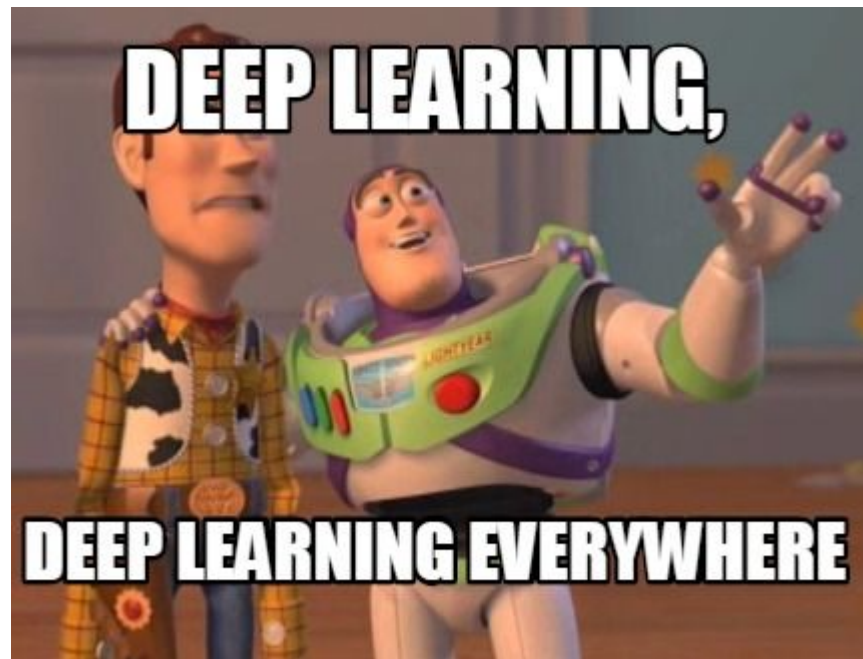
- des modèles qui apprennent à réaliser une tâche directement à partir des données brutes (ou très peu préparées)
- utilisent des réseaux de neurones artificiels (cf atelier)
- des millions de paramètres à entraîner !
- demandent beaucoup d'exemples d'apprentissage
- et beaucoup de temps (limite psychologique des “3 semaines”)
- utiliser des cartes graphiques (type Nvidia GTX ...)

Découvrir le Deep Learning par le pratique:

- Atelier “Reconnaissance Image - Deep Learning”
- Défi “Reconnaissance Image”
 - application aux voitures autonomes

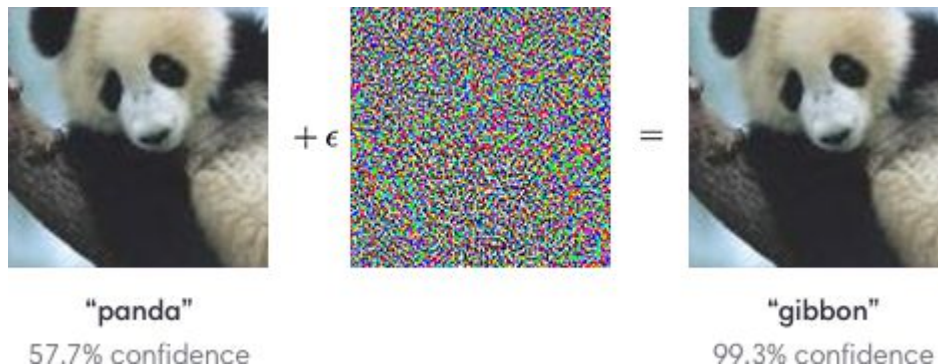
Deep Learning partout

- Voitures autonomes
- Imagerie médicale
- Jeu de Go (AlphaGo)
- Traduction automatique
- Recommandations (Deezer, Netflix, ...)
- Jeux vidéos (StarCraft, ...)
- Création de jeux vidéos
- un peu partout maintenant ...



Critiques du Deep Learning

- Peu de résultats théoriques !
 - très empirique : par essais/erreurs \gg expertise 'Deep Learning'
- Pas de “convergence assurée”
- Facilement “hackable”



- Très récents : premiers résultats importants en 2012
 - Benchmark 'ImageNet', depuis que du Deep Learning dessus
- Phénomène de “mode”
 - Vendre de l'IA absolument et utiliser le mot-clef “Deep Learning”
 - La recherche va très vite : \sim 30 papiers par jours à ce sujet et plus de 5 conférences majeures

Atelier “Reconnaissance Image”

Prendre en main une librairie Python

Prendre en main la **librairie 'Keras'** :

- créer et entraîner des modèles en quelques lignes
- pas besoin de tout coder à la main

Application : classification automatique d'images

Modèles :

- régression logistique
- réseau de neurones profond
- réseau de neurones convolutif