Une introduction à l'apprentissage automatique

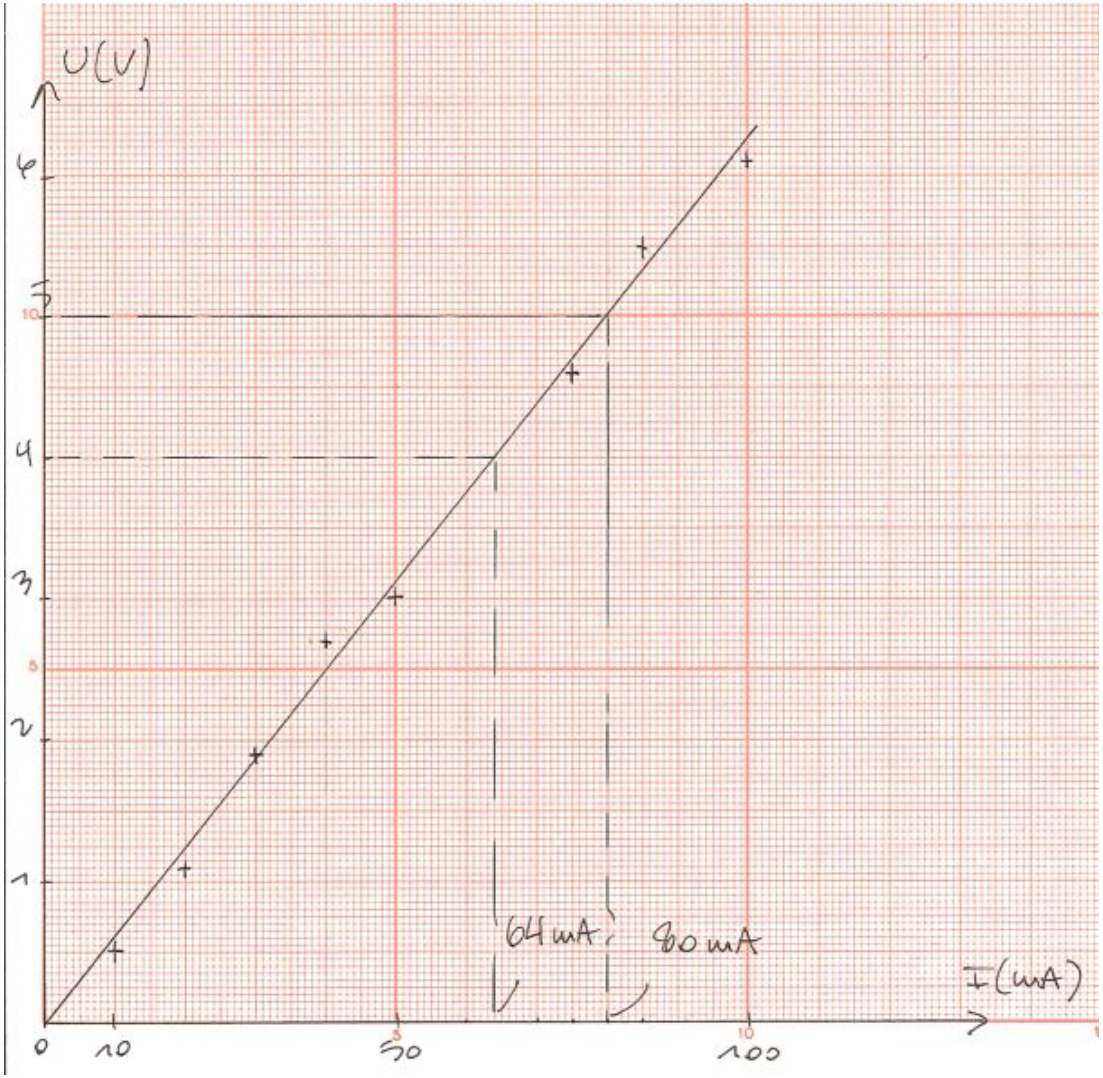
Eloïse BERTHIER

vendredi 8 mars 2019

Quels outils pour le *machine learning* ?

Machine Learning : l'étude scientifique des **algorithmes** et des modèles **statistiques** que les ordinateurs utilisent pour accomplir une tâche sans instruction explicite, mais plutôt en s'appuyant sur des motifs et de l'inférence.

Un exemple simple : la régression linéaire



On sait que la relation entre intensité et tension est linéaire :

$$U = RI$$

On a collecté des données $(I_j, U_j)_{j \in \{1,\dots,n\}}$.

On cherche à estimer la résistance R inconnue.

Trois mots sur les statistiques

• On suppose la relation entre intensité et tension linéaire : $U = RI$.

$\rightarrow$ C'est **un modèle statistique**.

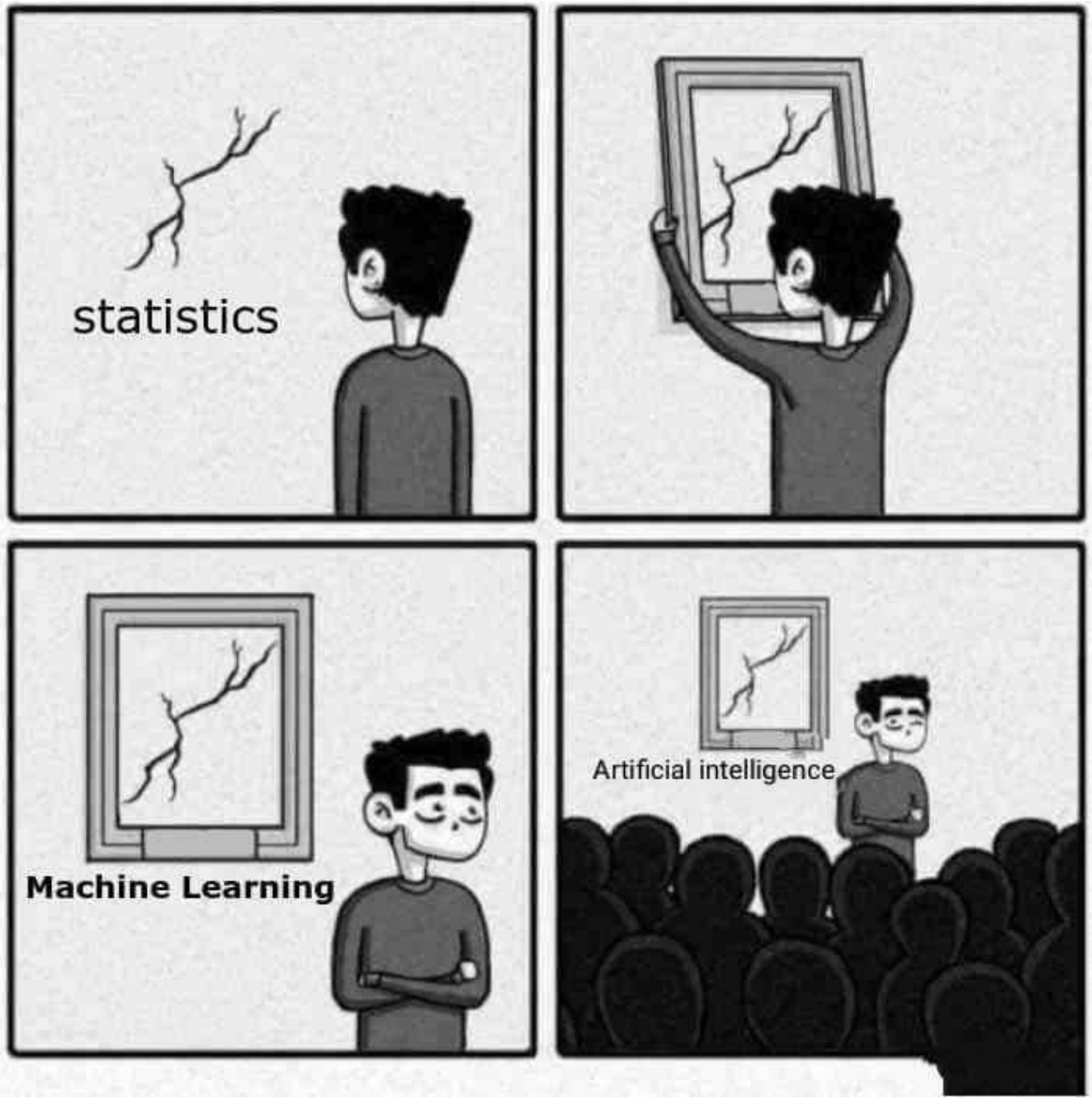
• On peut calculer explicitement le paramètre $\hat{R}$ estimé à partir des données. On peut même parfois obtenir une mesure d'incertitude.

$\rightarrow$ C'est **l'apprentissage statistique**.

• Une fois $\hat{R}$ calculé, on peut l'utiliser pour prédire la tension à une nouvelle intensité I_{new} :

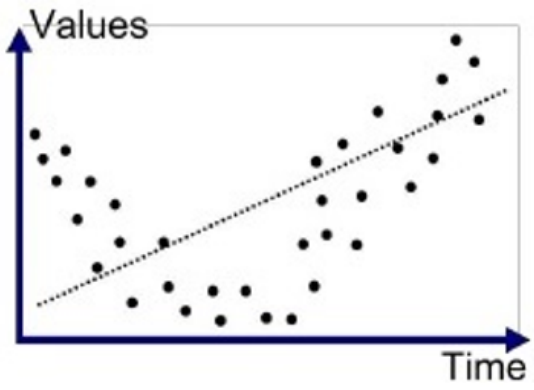
$$U_{new} = \hat{R} I_{new}$$

$\rightarrow$ C'est **l'inférence statistique**.

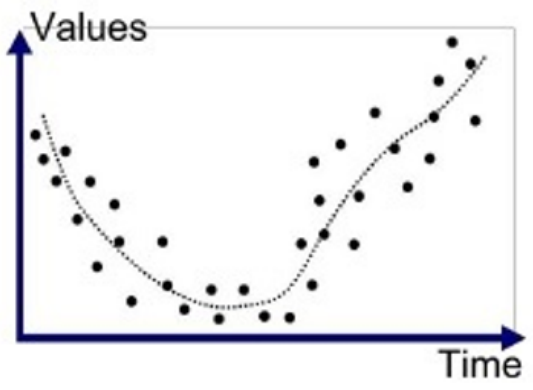

statistics
Machine Learning
Artificial intelligence

Comment construire un modèle statistique ?

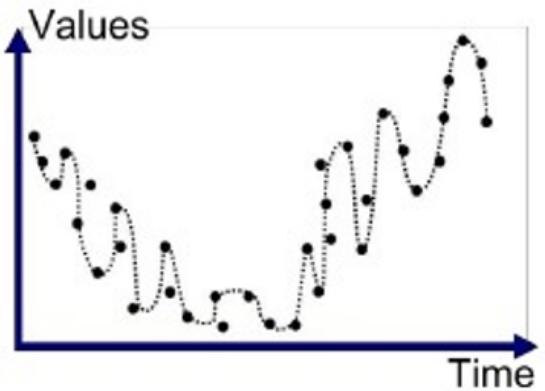
Choisir le bon niveau de complexité :



Underfitted



Good Fit/Robust



Overfitted

Comment apprendre les paramètres d'un modèle ?

Régression linéaire en dimension d :

- Modèle : $y = \langle w, x \rangle + \varepsilon$, où $x \in \mathbb{R}^d, y \in \mathbb{R}$ et $w \in \mathbb{R}^d$.
- Données d'apprentissage : $(x_i, y_i)_{i=1,\dots,n} \in (\mathbb{R}^d \times \mathbb{R})^n$.
- Minimisation de l'erreur d'apprentissage :

$$\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (\langle w, x_i \rangle - y_i)^2$$

- Solution : $\hat{w} = (X^\top X)^{-1} X^\top y$

où $X = \begin{pmatrix} x_1^1 & \dots & x_1^d \\ \vdots & \ddots & \vdots \\ x_n^1 & \dots & x_n^d \end{pmatrix}$ et $y = \begin{pmatrix} y_1 \\ \vdots \\ y_d \end{pmatrix}$

$\hookrightarrow$ **Algèbre linéaire**

L'apprentissage supervisé : cas général

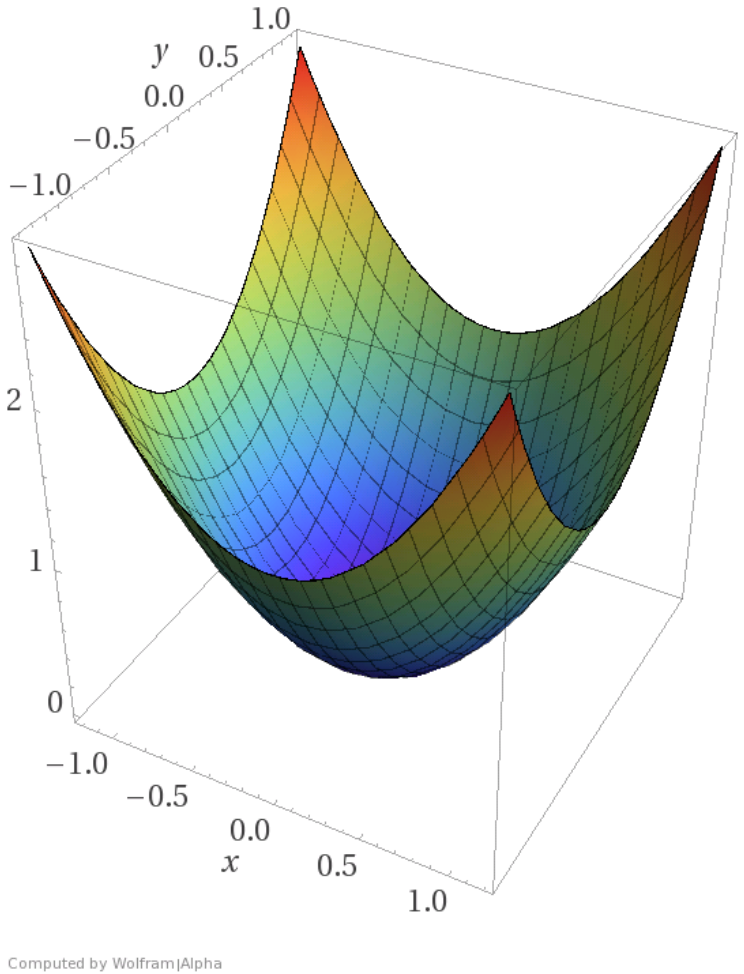
Données d'apprentissage : $(X_i, y_i)_{i=1,\dots,n} \in (\mathcal{X} \times \mathbb{R})^n$.
Modèle : $y = f(x)$, pour un certain $f \in \mathcal{F}$.
Problème à résoudre pour l'apprentissage :

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) + \lambda \Omega(f)$$

Optimisation — Erreur — Régularisation

Trouver un modèle qui fait peu d'erreurs, et le plus simple possible.

L'optimisation



But : minimiser une fonction convexe.

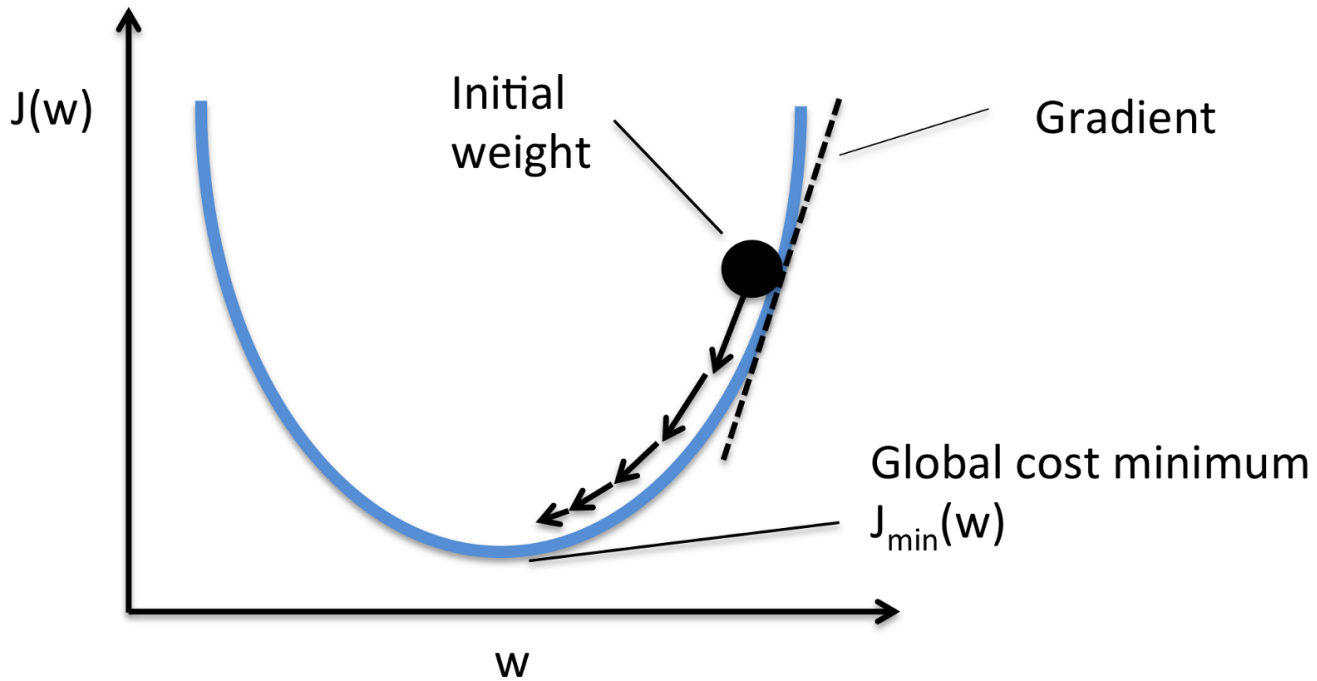
Trouver u tel que $\forall v, f(u) \leq f(v)$.

Fonction convexe :

$\forall u, v, \lambda \in (0,1), f((1-\lambda)u + \lambda v) \leq (1-\lambda)f(u) + \lambda f(v)$

Un algorithme d'optimisation...

L'**algorithme** le plus simple : la descente de gradient.



Répéter jusqu'à convergence : $w \leftarrow w - \eta J'(w)$

... Des algorithmes d'optimisation

- Gradient descent
- Stochastic gradient descent
- Coordinate gradient descent
- Accelerated gradient descent
- Averaged gradient descent
- Subgradient descent
- Proximal gradient descent
- Conjugate gradient descent
- Conditional gradient descent
- Newton method
- Quasi Newton methods
- Alternative direction method of multipliers
- Douglas-Rachford
- ...

+ versions distribuées sur plusieurs machines

En pratique

L'implémentation se fait majoritairement en **Python**, où la plupart des outils sont en *open source* et faciles d'utilisation.

En quelques lignes de code

```python
import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model
from sklearn.metrics import mean_squared_error, r2_score

# Load the diabetes dataset
diabetes = datasets.load_diabetes()


# Use only one feature
diabetes_X = diabetes.data[:, np.newaxis, 2]

# Split the data into training/testing sets
diabetes_X_train = diabetes_X[:-20]
diabetes_X_test = diabetes_X[-20:]

# Split the targets into training/testing sets
diabetes_y_train = diabetes.target[:-20]
diabetes_y_test = diabetes.target[-20:]

# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training sets
regr.fit(diabetes_X_train, diabetes_y_train)

# Make predictions using the testing set
diabetes_y_pred = regr.predict(diabetes_X_test)
```

Régression linéaire

```python
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import SGD

# Generate dummy data
import numpy as np
x_train = np.random.random((1000, 20))
y_train = keras.utils.to_categorical(np.random.randint(10, size=(1000, 1)), num_classes=10)
x_test = np.random.random((100, 20))
y_test = keras.utils.to_categorical(np.random.randint(10, size=(100, 1)), num_classes=10)

model = Sequential()
# Dense(64) is a fully-connected layer with 64 hidden units.
# in the first layer, you must specify the expected input data shape:
# here, 20-dimensional vectors.
model.add(Dense(64, activation='relu', input_dim=20))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(10, activation='softmax'))

sgd = SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy',
              optimizer=sgd,
              metrics=['accuracy'])

model.fit(x_train, y_train,
          epochs=20,
          batch_size=128)
score = model.evaluate(x_test, y_test, batch_size=128)
```

Réseau de neurones

En conclusion

Quels outils pour le *machine learning* ?

- des statistiques ;
- de l'algèbre linéaire ;
- de l'optimisation ;
- de l'algorithmique ;
- du Python ;
- et beaucoup d'anglais !

Data Scientist:

The Sexiest Job of the 21st Century

Meet the people who can coax treasure out of messy, unstructured data.

by Thomas H. Davenport and D.J. Patil

When Jonathan Goldman arrived for work in June 2006 at LinkedIn, the business networking site, the place still felt like a start-up. The company had just under 8 million accounts, and the number was growing quickly as existing members invited their friends and colleagues to join. But users weren't seeking out connections with the people who were already on the site at the rate executives had expected. Something was apparently missing in the social experience. As one LinkedIn manager put it, "It was like arriving at a conference reception and realizing you don't know anyone. So you just stand in the corner sipping your drink—and you probably leave early."