

# **Apprentissage Automatique**

## **MI 203**

Introduction

**22/01/2020**

S. Herbin, A. Chan Hon Tong

# Organisation du cours

Date	Intervenant cours	Intervenant TD supplémentaires	Contenu
22/01/2020	S. HERBIN	A. LECHAT N. DIAZ RODRIGUEZ	Introduction
29/01/2020	S. HERBIN	R. CAYE DAUDT N. DIAZ RODRIGUEZ	Arbres
5/2/2020	A. CHAN HON TONG	G. VAUDAUX RUTH N. DIAZ RODRIGUEZ G. LENCZNER	Réseaux de neurones
12/2/2020	A. CHAN HON TONG	G. VAUDAUX RUTH N. DIAZ RODRIGUEZ G. LENCZNER	Deep Learning
26/2/2020	A. CHAN HON TONG	R. CAYE DAUDT N. DIAZ RODRIGUEZ G. LENCZNER	Non supervisé
4/3/2020	S. HERBIN	A. LECHAT N. DIAZ RODRIGUEZ	SVM
11/3/2020	S. HERBIN	N. DIAZ RODRIGUEZ G. LENCZNER	Examen écrit (1h) + TD noté (2h)

# Equipe pédagogique



**Stéphane Herbin**  
**stephane.herbin@onera.fr**



**Adrien Chan Hon Tong**  
**adrien.chan\_hon\_tong@onera.fr**



**Natalia Diaz Rodriguez**



**Alexis Lechat**



**Rodrigo Caye Daudt**



**Guillaume Vaudaux Ruth**

**Adrien Bennetot**

**Gaston Lenczner**

# ONERA

- DTIS: Département Traitement de l'Information et Systèmes à Palaiseau
- Equipe IVA (Image, Vision, Apprentissage)
- Activités de Machine Learning: télédétection, vidéo, drones
- Stages et thèses: (<http://w3.onera.fr/stages/stages-dtis>)

# Organisation

- Cours magistral: 1h à 1h30
- TD applicatif: 1h30 à 2h
- Examen: 1h (le 11/3/2020)
- TD noté: 2h (le 11/3/2020)
- Note : Exam (50%), TD (50%)

# Aujourd'hui

- Organisation du cours
- Introduction générale:
  - Exemples, définitions, problématiques, approches, pratiques
- Deux approches élémentaires à connaître:
  - Modélisation bayésienne
  - k plus proches voisins (kNN)

# Apprentissage Automatique

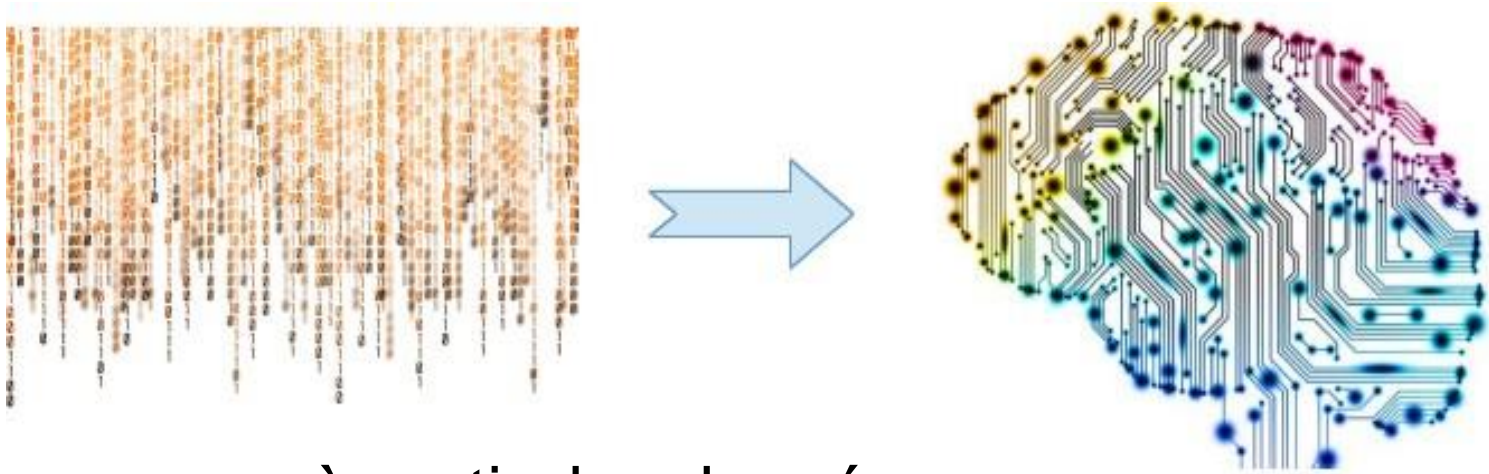
- Pourquoi ce cours?
  - Des données partout (image, son, texte, traces...)
  - Thème « à la mode »: intelligence artificielle, deep learning, big data...
  - Intérêt technologique: c'est efficace (sous certaines hypothèses)
  - Intérêt scientifique: permet une modélisation empirique
- Finalité: **donner du sens** aux signaux, images et données
  - Inférence/Prédiction → valeur numérique
  - Classification → valeur discrète, label
  - Description → texte
- Objectifs: savoir mettre en œuvre une approche d'apprentissage
  - Cours → méthodes
  - TD → pratique

# Vocabulaire

- Reconnaissance des formes (« pattern recognition »)
- Apprentissage (« Machine Learning »)
- Intelligence artificielle
- Data Science
- Statistique
- « Data Driven » vs. « Model Based »



# Apprentissage Automatique

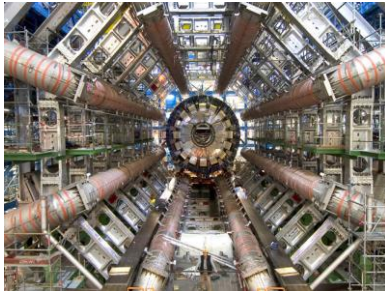


Programmer à partir des données :

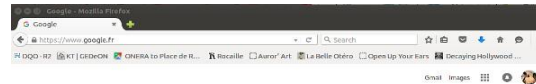
- *Comment extraire l'information des données pour nous aider à prendre de meilleures décisions ?*
- *Comment prendre automatiquement des décisions en fonction des données ?*
- *Comment adapter un système à un environnement changeant ?*

# Données = carburant du ML

CERN /  
Large Hadron Collider  
~70 Po/an



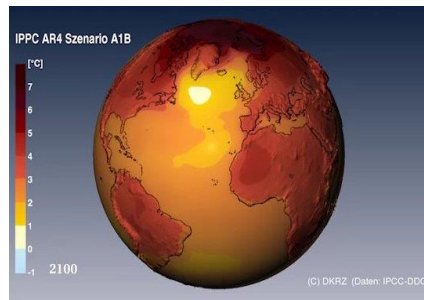
Google :  
24 PetaOctets/jour



Copernicus :  
> 1Po/an



DKRZ (Climat)  
500 Po



Google



Square Kilometer Array  
1376 Po/an (en 2024)



BIG DATA

# Apprentissage automatique : applications

Anti-Spam (*Classifieur Bayésien*)



1997 : DeepBlue bat Kasparov

2017: Alpha GO bat Ke Jie



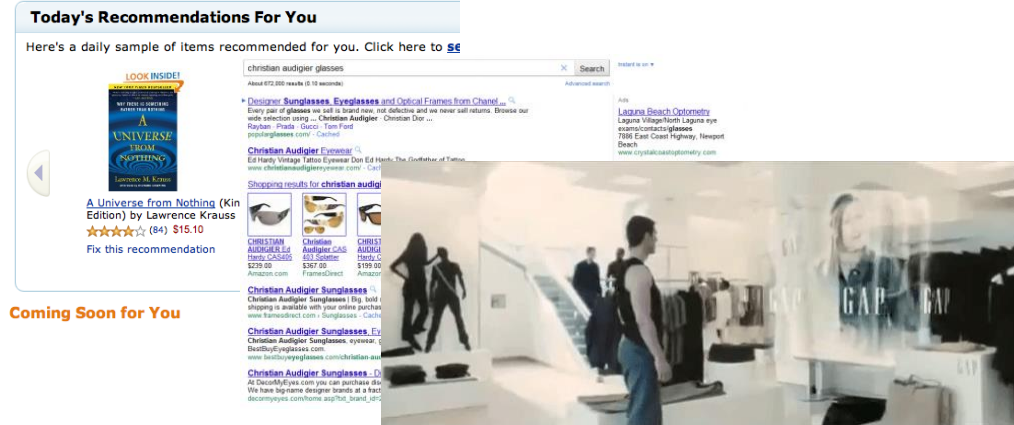
Tri postal automatique (*détection de chiffres manuscrits par réseaux de neurones*)



# Apprentissage automatique : applications

Recommandation ciblée  
(régression logistique)

Michel, Welcome to Your Amazon.com (if you're not Michel.Trottier)



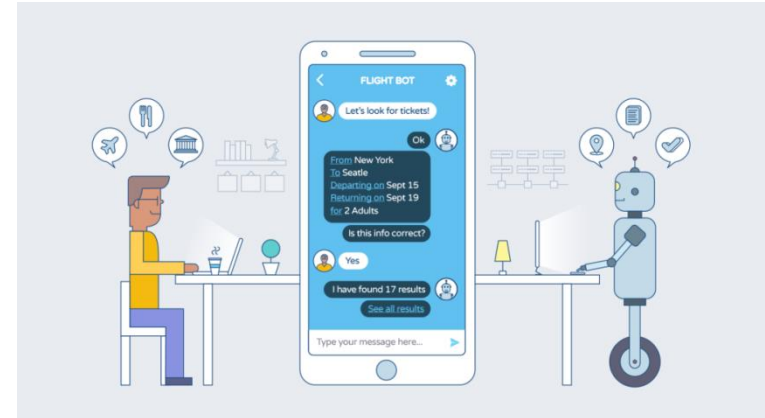
Appareil photo avec détection  
de visages (boosting)



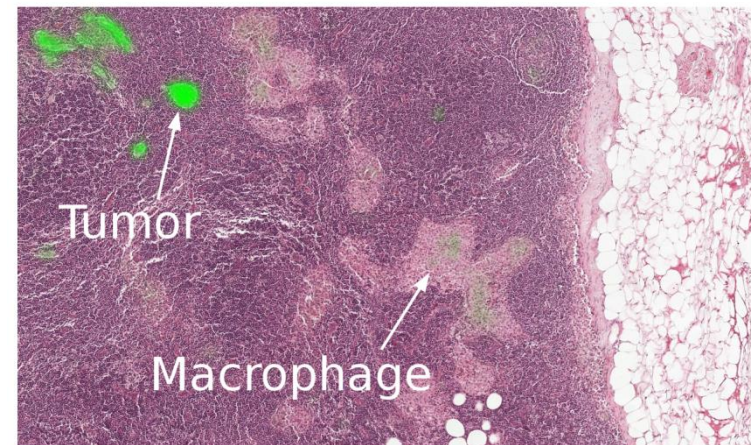


# Apprentissage automatique : applications

Chat Bots  
(*Réseaux de neurones*)



Détection de tumeurs  
(*Réseaux de neurones*)



# Pourquoi l'apprentissage automatique ?

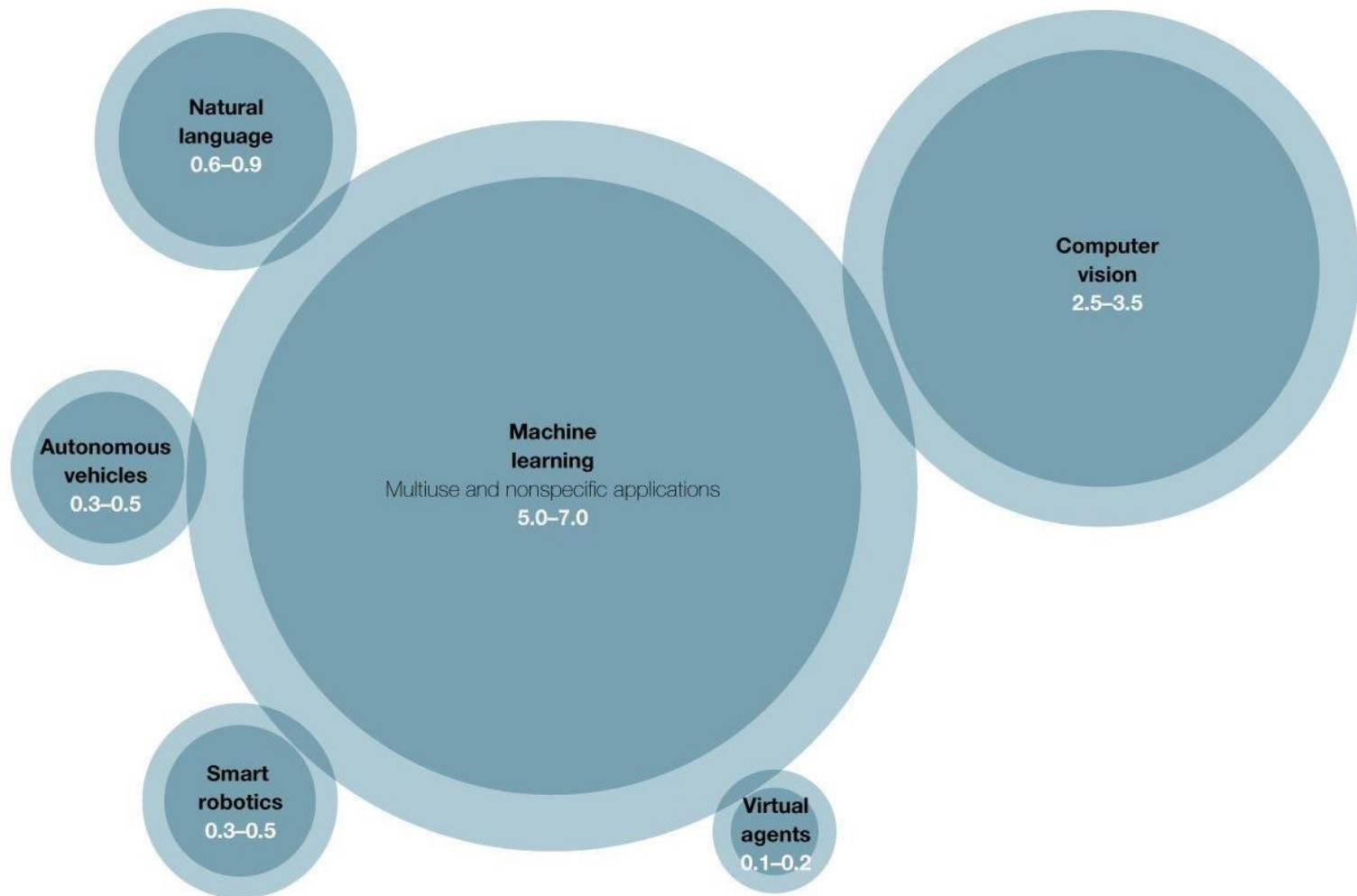
- Raison épistémologique
  - On ne sait pas modéliser les problèmes complexes ... mais on dispose d'exemples en grand nombre décrivant la variété des situations
- Raison scientifique
  - L'apprentissage est une faculté essentielle du vivant
- Raison économique
  - La récolte de données est plus facile que le développement d'expertise

# Machine Learning

- Un domaine scientifique hybride:
  - Statistique
  - Intelligence artificielle
  - « Computer science »
  - Traitement du signal
- Utilisant des techniques généralistes:
  - Optimisation numérique
  - Hardware
  - Gestion de base de données

## External investment in AI-focused companies by technology category, 2016<sup>1</sup>

\$ billion



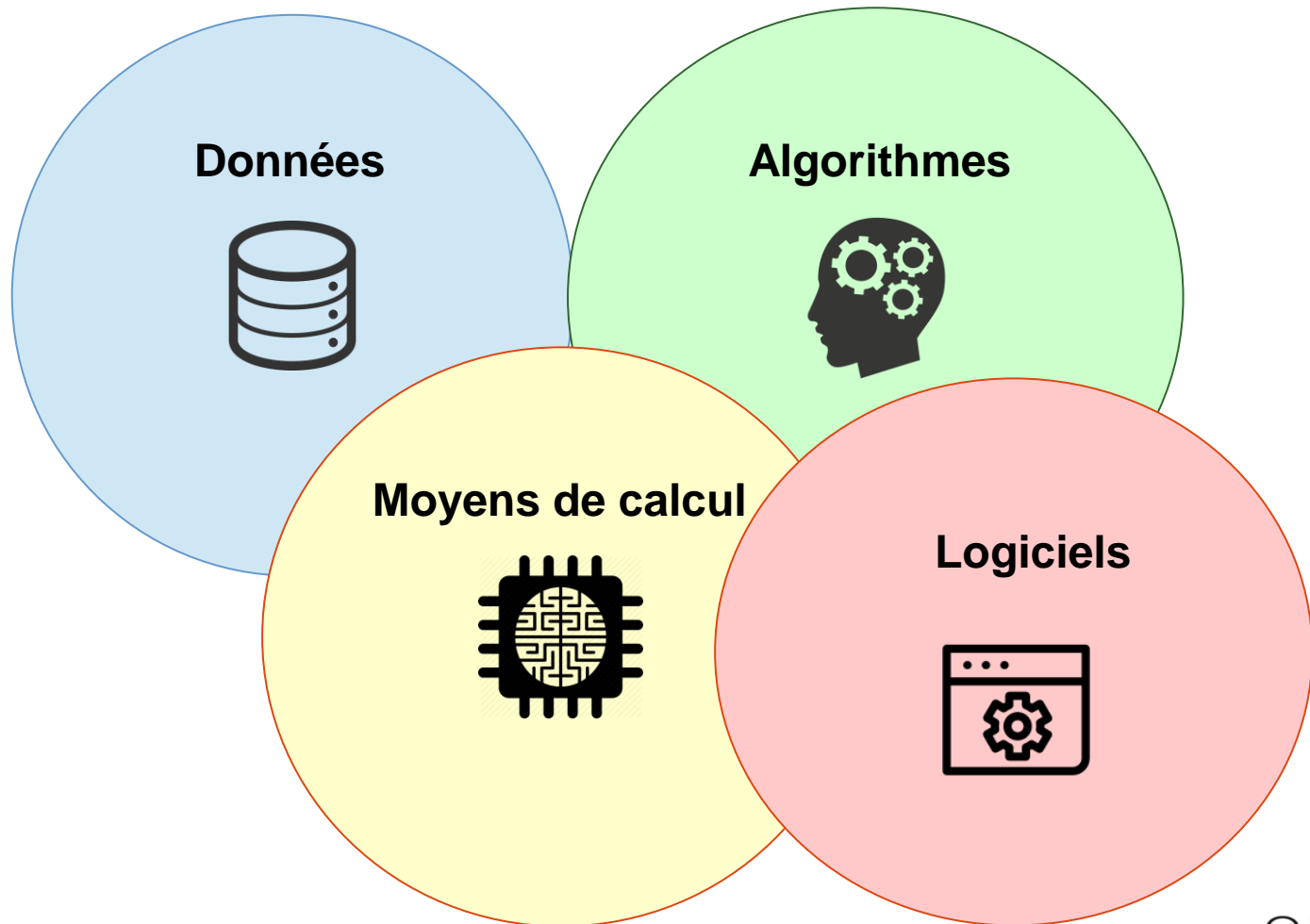
<sup>1</sup> Estimates consist of annual VC investment in AI-focused companies, PE investment in AI-related companies, and M&A by corporations. Includes only disclosed data available in databases, and assumes that all registered deals were completed within the year of transaction.

McKinsey&Company | Source: Capital IQ; Pitchbook; Dealogic; McKinsey Global Institute analysis



# « Deep Learning » : le mot clé inévitable

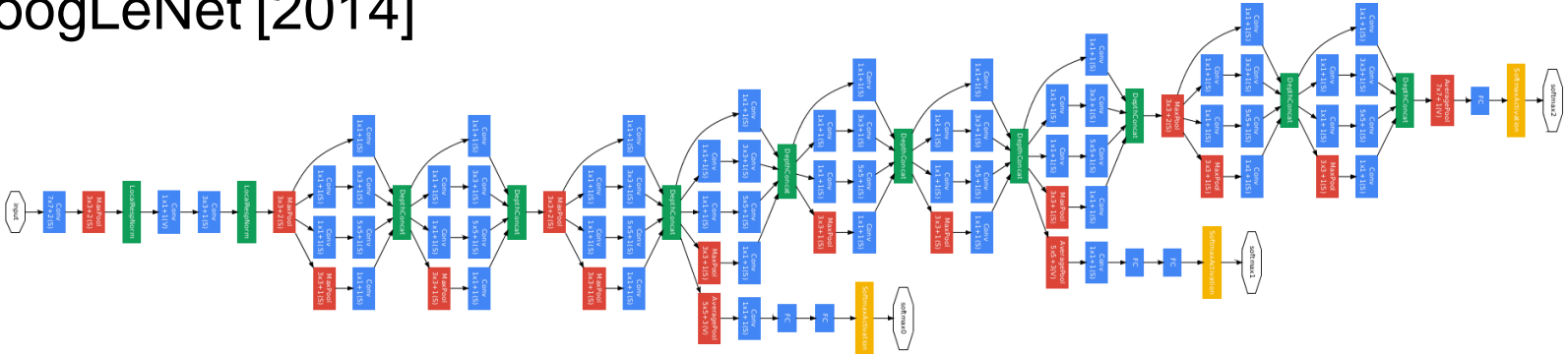
*Une rupture scientifique et technologique en apprentissage*



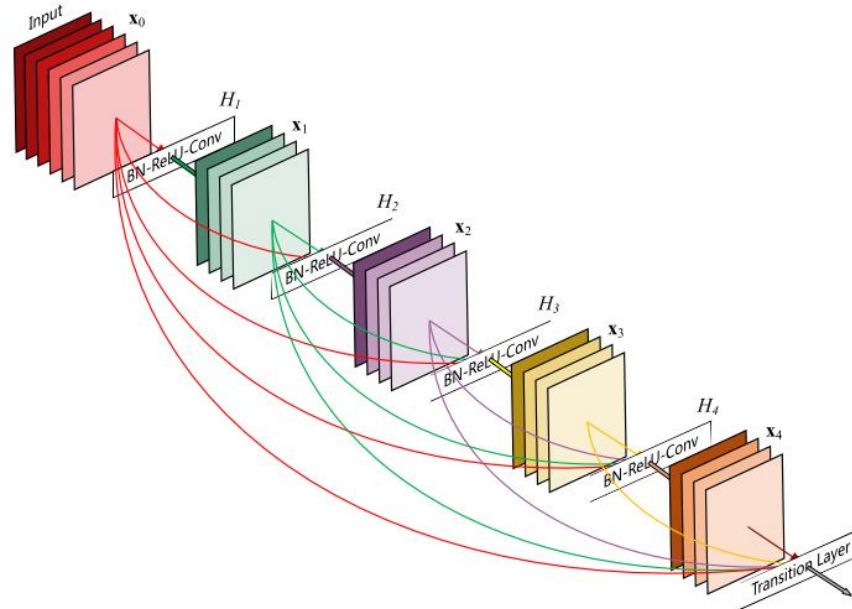
# « Deep Learning »

Modèles de réseaux de neurones hiérarchiques :

GoogLeNet [2014]



DenseNet [2016]



# « Deep Learning »

L'invasion [2010s] : classification Imagenet

2012 Teams	%error	2013 Teams	%error	2014 Teams	%error
Supervision (Toronto)	15.3	Clarifai (NYU spinoff)	11.7	GoogLeNet	6.6
ISI (Tokyo)	26.1	NUS (singapore)	12.9	VGG (Oxford)	7.3
VGG (Oxford)	26.9	Zeiler-Fergus (NYU)	13.5	MSRA	8.0
XRCE/INRIA	27.0	A. Howard	13.5	A. Howard	8.1
UvA (Amsterdam)	29.6	OverFeat (NYU)	14.1	DeeperVision	9.5
INRIA/LEAR	33.4	UvA (Amsterdam)	14.2	NUS-BST	9.7
		Adobe	15.2	TTIC-ECP	10.2
		VGG (Oxford)	15.2	XYZ	11.2
		VGG (Oxford)	23.0	UvA	12.1

ConvNet / non-ConvNet

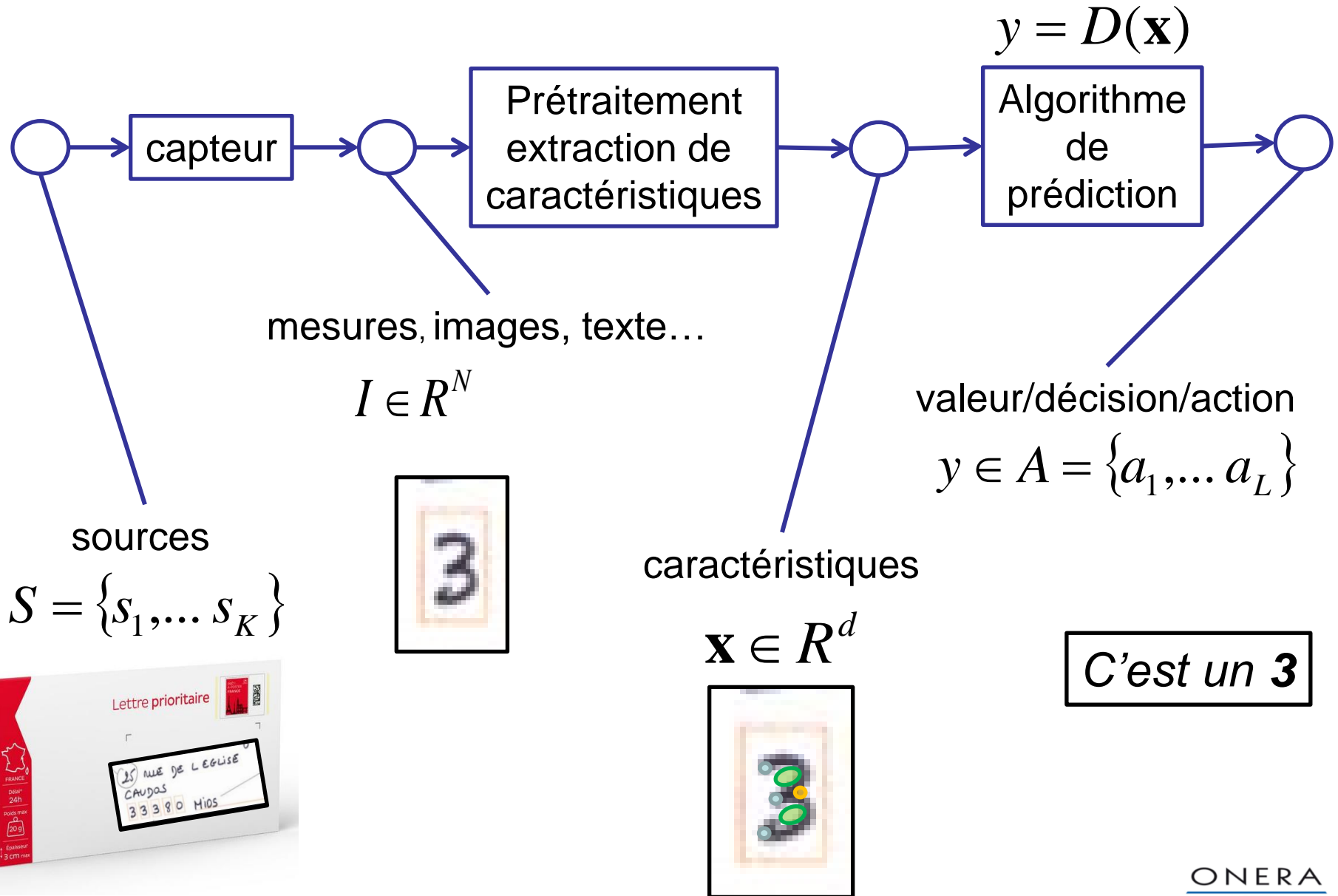
Figure extraite de LeCun @ CVPR'2015

→ *Performants pour une multitude de tâches (autres que vision), avec des jeux de données variés.*

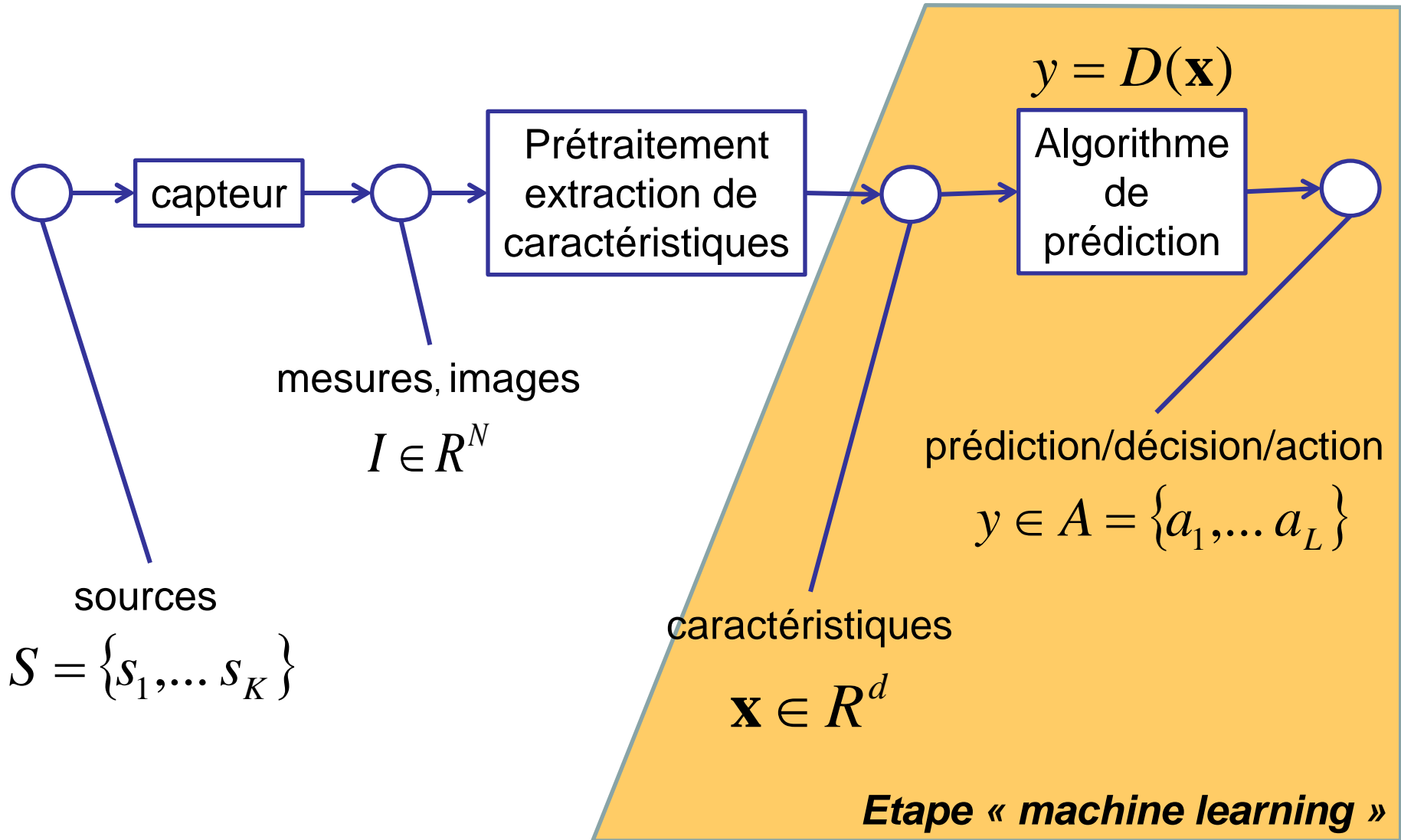
# MACHINE LEARNING

## Démarche générale

# Chaîne de prédiction générique



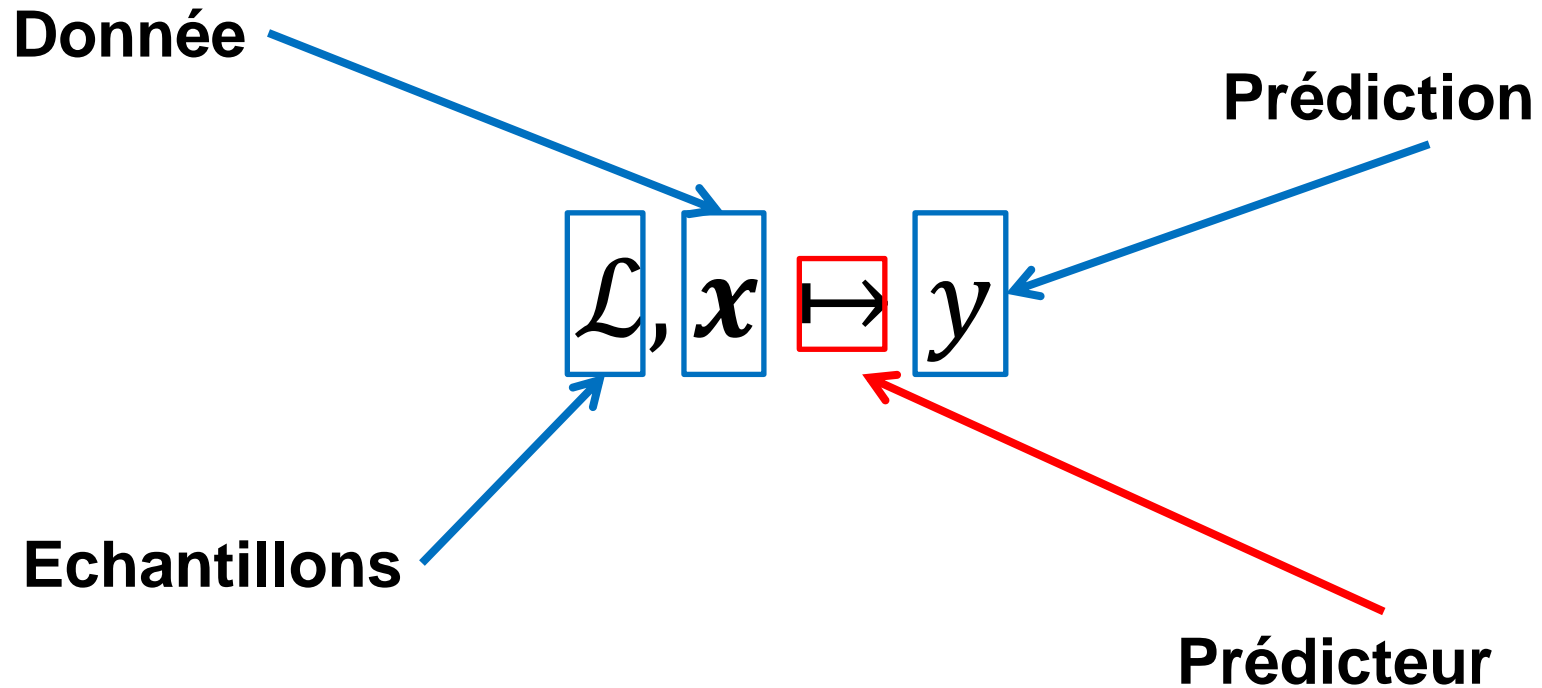
# Prédire / inférer



# Machine Learning

- Donnée ( $x$ )
  - Mesures, texte, image, enregistrement, vidéo ou caractéristiques extraites de ...
- Prédiction ( $y$ )
  - Décision, choix, action, réponse, préférence, groupe, valeur...
- Echantillons ( $\mathcal{L}$ )
  - Exemples de données et de prédictions (bonnes ou mauvaises)
  - Base d'apprentissage

# Machine Learning



- Hypothèse forte: les échantillons contiennent toute l'information exploitable
- Prédicteur = « interpolateur » à partir des données  $\mathcal{L}$



# Programmer à partir de données

- Comment définir le prédicteur?
- Modèle paramétrique:

$$f: \mathbf{W}, \mathbf{x} \mapsto y$$

- Apprendre = trouver à partir de l'ensemble d'échantillons  $\mathcal{L}$  les bons paramètres  $\mathbf{W}$

$$\mathcal{L} \mapsto \mathbf{W}$$


- « Bon » = capable de reproduire le comportement défini par  $\mathcal{L}$  sur de nouvelles données = **généralisation**

# Deux phases

Test

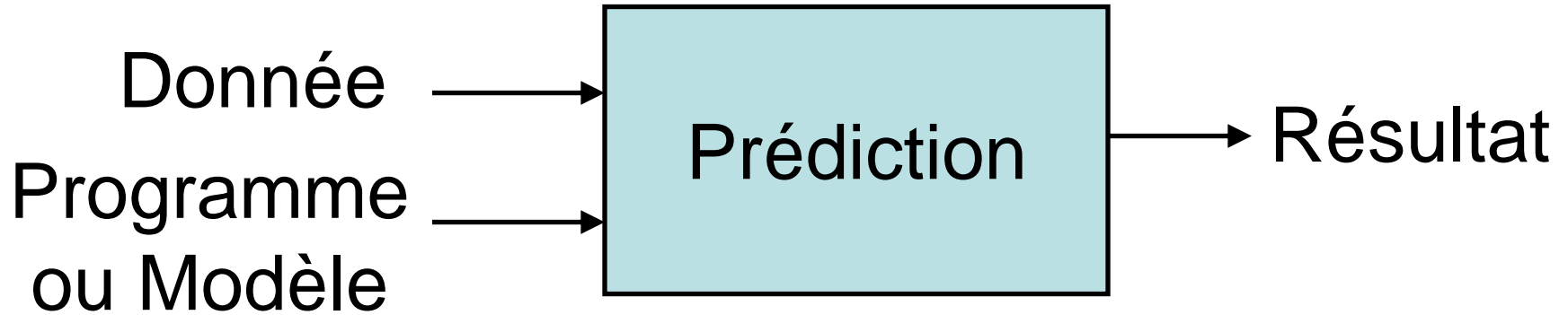
$$f: W, x \mapsto y$$

Apprentissage

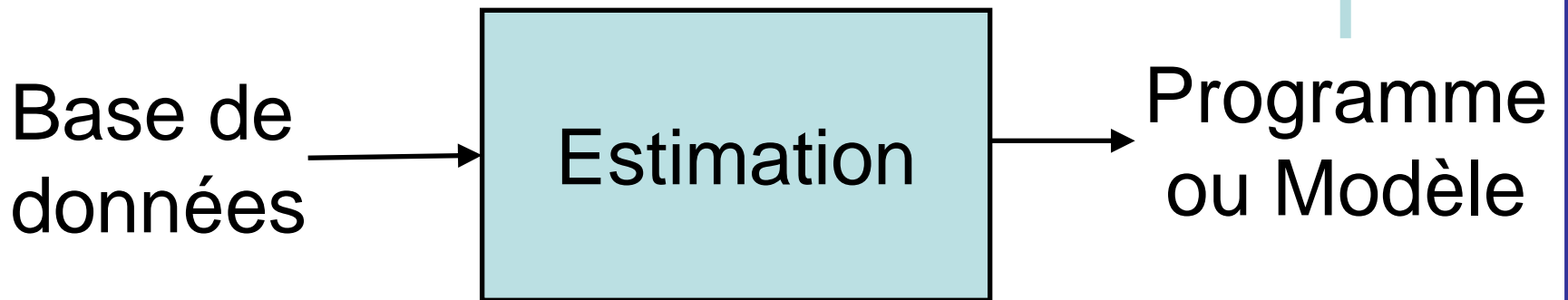
$$\mathcal{L} \mapsto W$$


# Deux phases

## Test



## Apprentissage



# Exemple: Reconnaissance de chiffres manuscrits



- Comment définir les éléments ?

$$\mathcal{L}, \mathbf{W}, \mathbf{x}, y$$

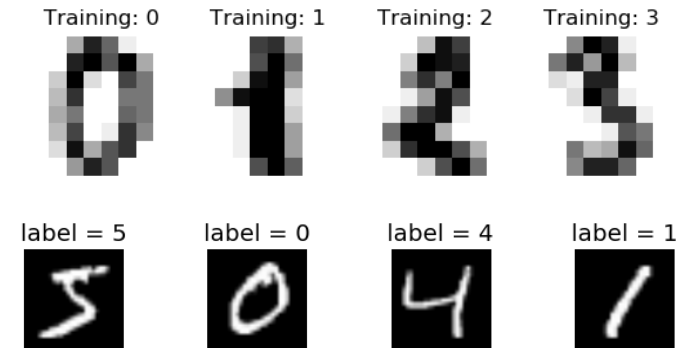
- Les fonctions d'apprentissage et de prédiction?

$$\mathcal{L} \mapsto \mathbf{W}$$

$$\mathbf{W}, \mathbf{x} \mapsto y$$

# Etape 1: choix de la base d'apprentissage

- Elle existe:
  - Scikit-learn:
  - MNIST:
  - SVHN:



- Il faut la construire:
  - Recueil de données existantes
  - Expérimentations (photos, mesures...)

# Etape 2: analyse & représentation des données

0	0	0	0
0	1	0.5	0
0	0.5	1	0.5
0	0	0.5	1

**Image**

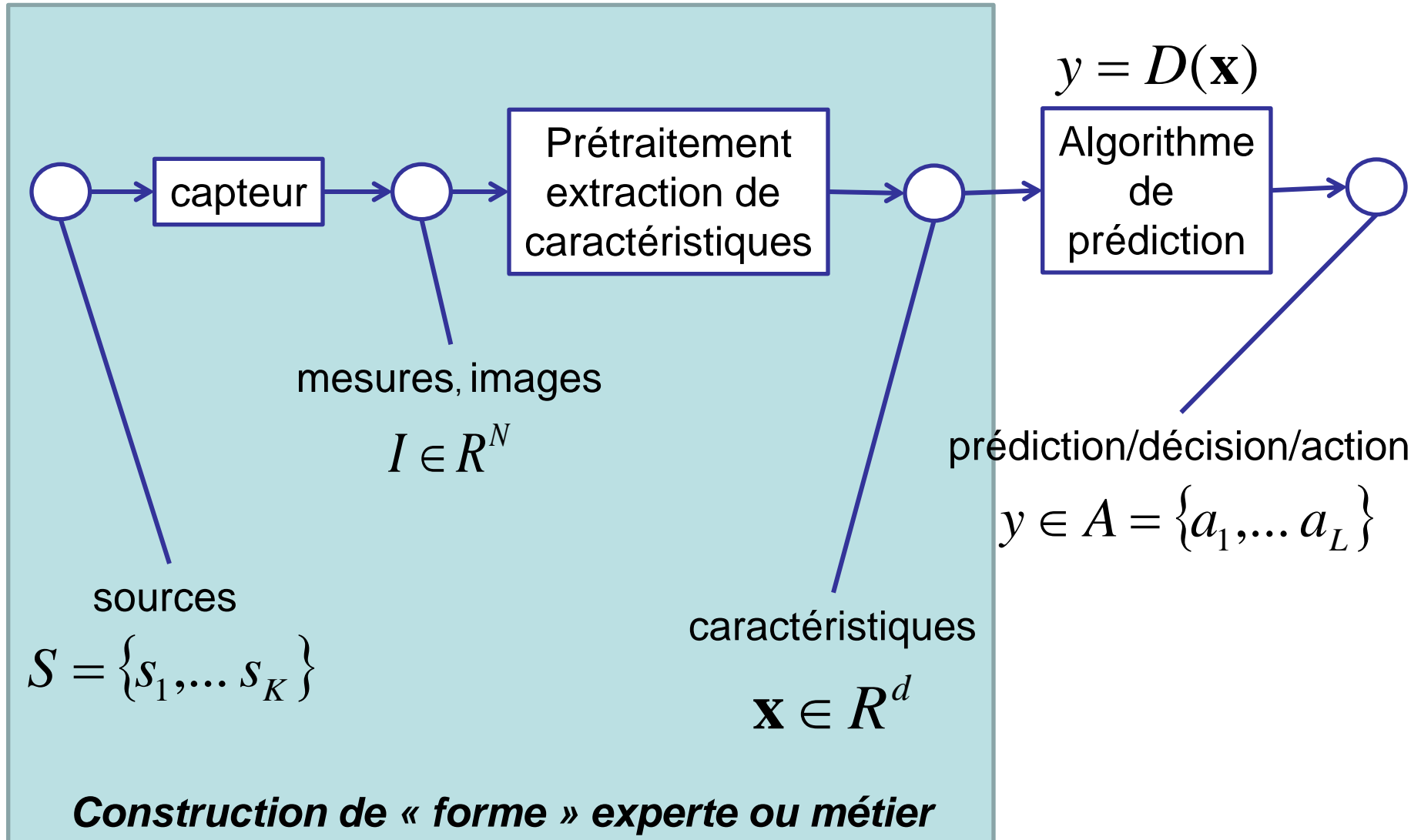
**Extraction de  
caractéristiques**

*Extraction  
minimale = image  
vectorisée  
2D→1D*

0
0
0
0
0
1
0.5
0
0
0.5
1
0.5
0
0
0.5
1

**Vecteur**

# Extraction de caractéristiques = Transformer les données brutes



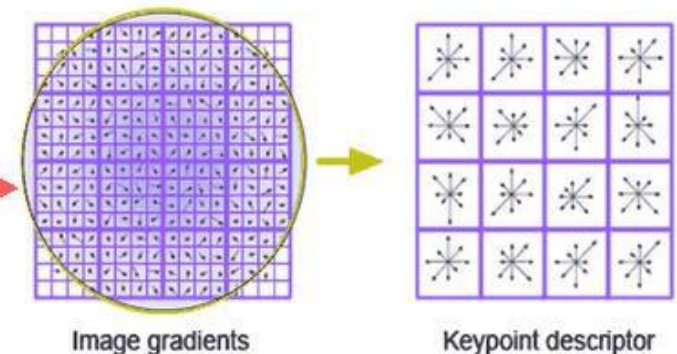
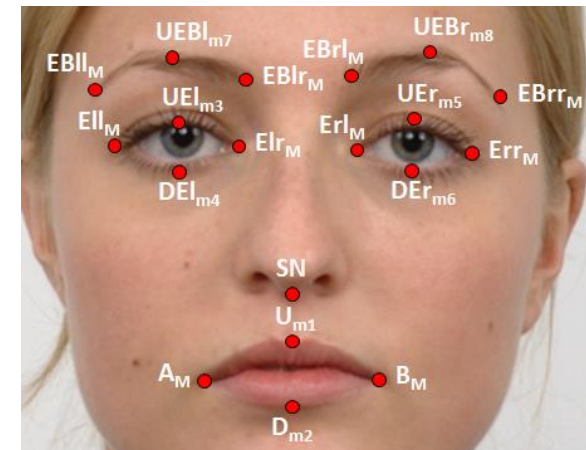
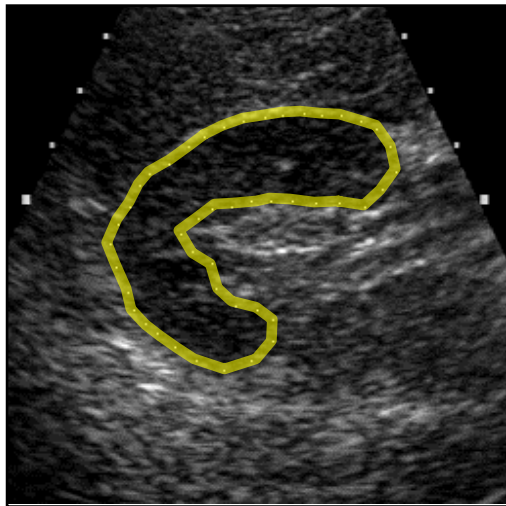
# Extraction de caractéristiques

- « Feature extraction » en anglais
- Données brutes pas exploitables directement:
  - Bruitées
  - Grandes dimensions (image, enregistrement)
  - Information utile noyée
- Etape critique
  - Caractéristiques trop simples: pas assez d'information, confusion
  - Caractéristiques trop riches: complexité, bruit, grande variabilité
  - ➔ Compromis difficile à régler entre invariance/robustesse/taille/coût de calcul...
- Deux cas de figure:
  - On sait ce qui est important et pourquoi  
➔ modélisation
  - On ne sait pas décrire ce qui est important  
➔ on l'apprend!

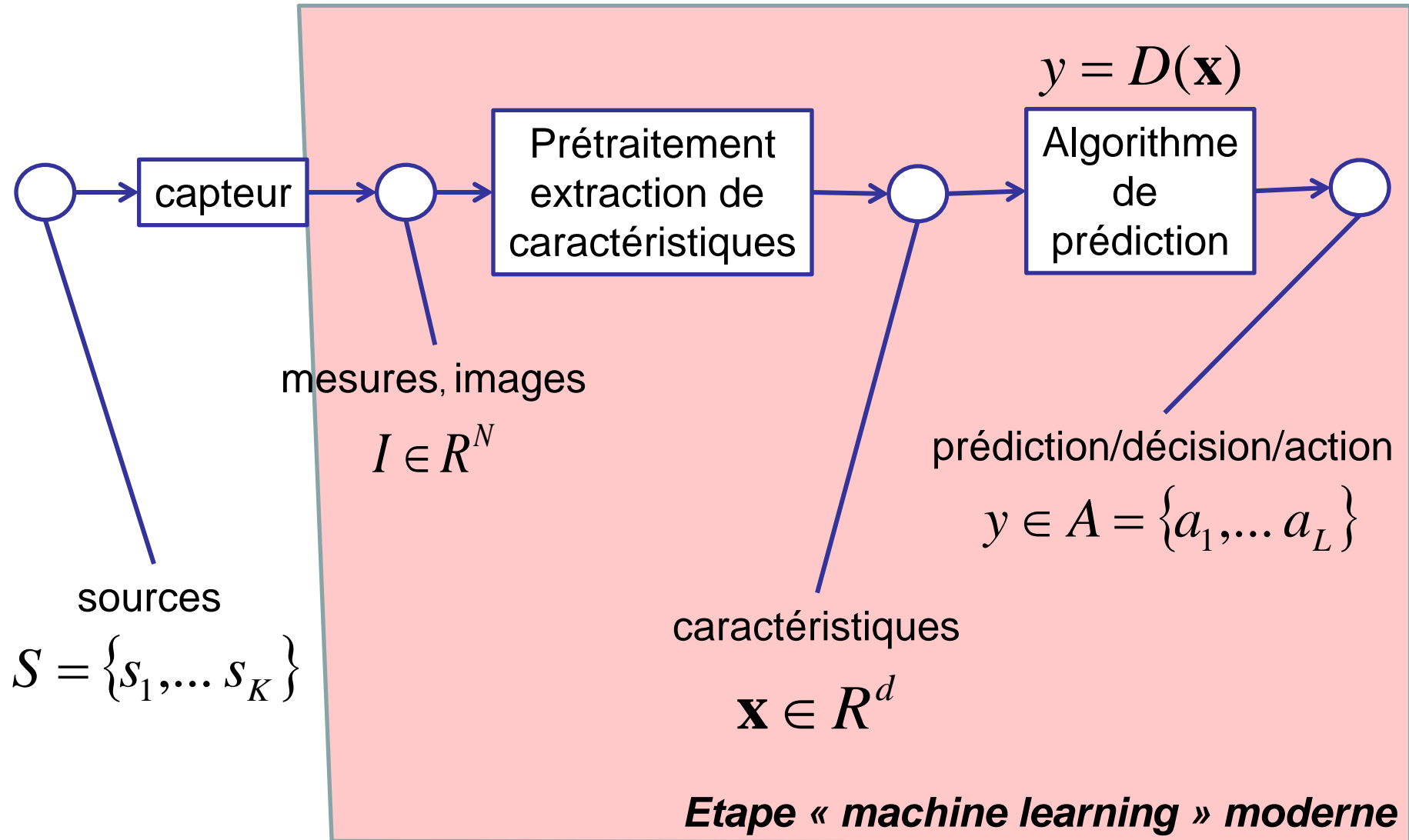


# Quelques exemples en image

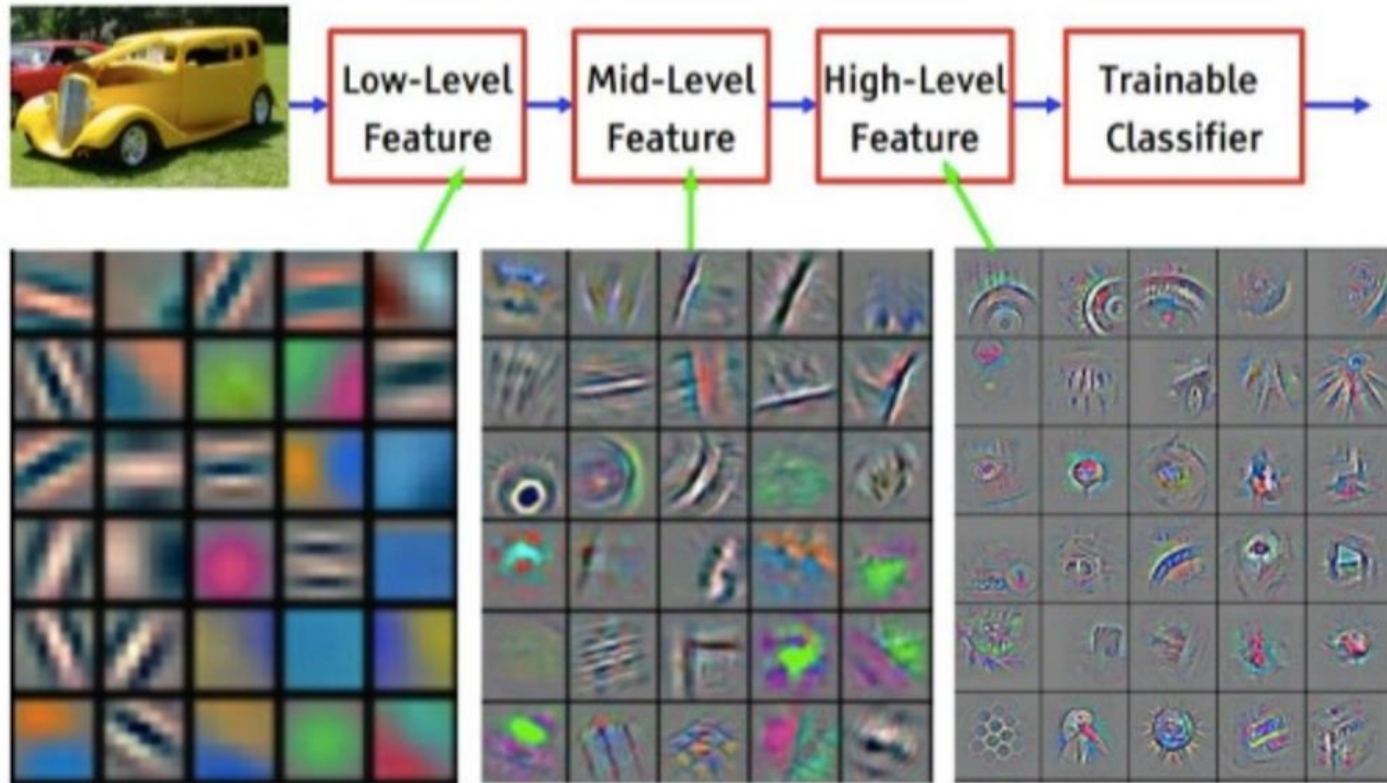
- Deux grandes classes: forme ou texture
- Forme
  - Dépend d'une étape de séparation du fond (segmentation, saillance)
  - Caractéristiques structurelles ou
- Texture
  - Globales et/ou locales
  - Plus difficiles à associer à un objet précis



# Prédire & transformer et en même temps



# « Deep features »



On peut apprendre les caractéristiques image  
Réseaux convolutifs (cours N° 6)

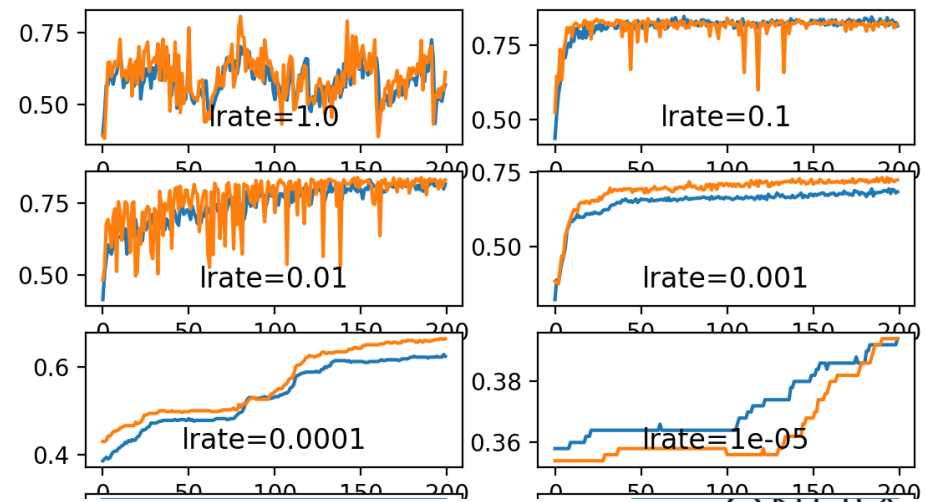
# Etape 3: choix de l'approche

- Quel type de prédicteur?
  - Classification
  - On connaît les classes cibles → Apprentissage supervisé
- Nature des données?
  - Vecteurs de taille fixe mais grands → algorithmes avec bon contrôle de la régularisation
- Taille de la base de données?
  - Grande (> 10000 exemples) → optimisation efficace
- Plusieurs choix possibles:
  - Arbres de décision, SVM, Réseaux de neurones...

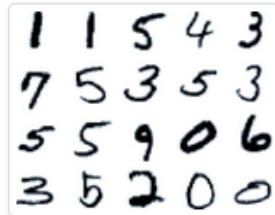
# Etape 4: optimisation

Apprentissage =

- définir un critère paramétrique fonction des données (coût, énergie...)
- appliquer un optimiseur
- régler ses paramètres
- vérifier que l'apprentissage se passe bien
  - évaluation de la capacité de généralisation
  - convergence










# Etape 5: évaluation



**MNIST** 50 results collected

Units: error %

Classify handwritten digits. Some additional results are available on the [original dataset page](#).

Result	Method	Venue	Details
0.21%	<a href="#">Regularization of Neural Networks using DropConnect</a> 	ICML 2013	
0.23%	<a href="#">Multi-column Deep Neural Networks for Image Classification</a> 	CVPR 2012	
0.23%	<a href="#">APAC: Augmented PAttern Classification with Neural Networks</a> 	arXiv 2015	
0.24%	<a href="#">Batch-normalized Maxout Network in Network</a> 	arXiv 2015	<button>Details</button>
0.29%	<a href="#">Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree</a> 	AISTATS 2016	<button>Details</button>
0.31%	<a href="#">Recurrent Convolutional Neural Network for Object Recognition</a> 	CVPR 2015	
0.31%	<a href="#">On the Importance of Normalisation Layers in Deep Learning with Piecewise Linear Activation Units</a> 	arXiv 2015	

# Résumé des étapes

1. Constituer une base d'apprentissage
2. Analyser et visualiser les données (format, distribution, représentation, dimensions, quantité...)
3. Concevoir le modèle (type de prédicteur, principe d'apprentissage)
4. Optimiser (l'apprentissage proprement dit)
5. Evaluer

# LES VARIÉTÉS D'APPRENTISSAGE



# Types de prédictions

- **Classification**

- Binaire: spam / non spam
- Identification: « tata Monique »

- **Régression**

- Prédiction de température, de cours de bourse
- Localisation d'objet dans image

- **Regroupement**

- Photos dans base de données personnelle

- **Texte**

- « C'est un chat qui saute sur une table. »

# Types d'apprentissage

- **Apprentissage supervisé**
  - Les données d'apprentissage contiennent les objectifs de prédiction (annotations)
- **Apprentissage non supervisé**
  - Les données d'apprentissage sont brutes
- **Apprentissage semi-supervisé**
  - Les données d'apprentissage sont partiellement annotées
- **Apprentissage par transfert**
  - Les données d'apprentissage sont proches du problème visé
- **Apprentissage par renforcement**
  - Les prédictions sont issues d'une séquence d'actions et sont caractérisées par une mesure de qualité (« reward »)

# Optimisation

- **Optimisation convexe**
  - Ex. Minimisation séquentielle de problème quadratique
- **Optimisation stochastique**
  - Ex. Descente de gradient stochastique, Algorithmes génétiques
- **Optimisation sous contraintes**
  - Ex. Programmation linéaire
- **Optimisation combinatoire**
  - Ex. Algorithmes gloutons

# Format du prédicteur

- Dépend de la forme des données (vecteurs, listes, réels/discret) et du type de prédiction
- Exemples
  - Plus proches voisins
  - Machines à vecteurs de supports (SVM)
  - Arbre de décision
  - Ensembles de classifieurs (forêts aléatoires, « boosting »...)
  - Réseaux de neurones
  - Règles/Programmation logique
  - Modèles probabilistes (Réseaux bayésiens, Chaînes ou champs de Markov...)
  - Etc.

# Evaluation

- Dépend du type de prédiction
- Classification
  - Taux d'erreur moyen
  - Matrice de confusion
  - Précision/rappel
  - Courbe ROC
- Régression
  - Erreur quadratique
- Détection
  - Taux de recouvrement moyen

# Construire son chantier d'apprentissage

- Préparer les données
  - Simplifier/débruiter/formater/homogénéiser
- Diviser en trois ensembles:
  - **Apprentissage** (“Train”) pour optimiser les paramètres du modèle.
  - **Validation** pour évaluer la qualité de l'apprentissage et optimiser les hyper-paramètres
  - **Test** pour estimer la qualité de l'apprentissage dans son contexte d'utilisation, i.e. **l'erreur de généralisation**.
- L'ensemble de test n'est jamais utilisé pour l'apprentissage (optimisation), seulement pour son évaluation.

# DEUX APPROCHES ÉLÉMENTAIRES

Modélisation bayésienne  
Plus proches voisins

# Théorie Bayésienne de la décision

- On considère les données  $x, y$  comme des variables aléatoires.
- On les modélise par des lois de probabilités:
  - $P(x), P(y)$  : lois a priori (ou marginales)
  - $P(x, y)$  : loi jointe
  - $P(x | y)$  : vraisemblance conditionnelle
  - $P(y | x)$  : loi a posteriori
- Classification:  $y \in \{1, 2 \dots N\}$  est une étiquette
- On cherche à prédire une unique étiquette  $y^*$  à partir de  $x$
- Théorie de la décision démontre que le meilleur choix est:

$$y^* = \arg \max_y P(y | x)$$



# Théorie Bayésienne de la décision

- Deux questions:
  - Comment calculer  $P(y | x)$  = apprentissage
  - Comment trouver le max = prédiction
- « Astuce »: utiliser la loi de Bayes

$$P(y | x) = \frac{P(x | y) P(y)}{P(x)}$$

- On connaît en général la fréquence d'occurrence des classes  $y$
- On sait plus facilement calculer la **vraisemblance**:  $P(x | y)$ 
  - « Si je sais dans quelle classe je suis, je sais décrire le comportement/distribution de mes données »
- Le max sur  $y$  ne dépend que de  $P(x | y)$  et  $P(y)$

$$y^* = \arg \max_y P(x | y) P(y)$$

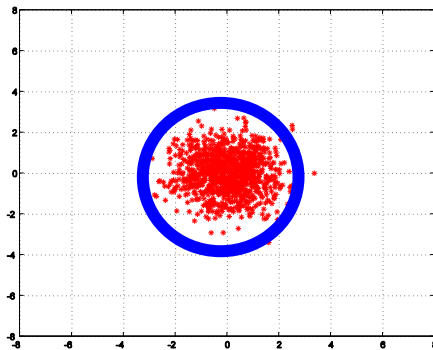
# Approche Bayésienne multivariée

- Calcul de la loi conditionnelle: Modèle multivarié
- Par ex. modèle gaussien décrivant  $\mathbf{x} = [x_1, x_2 \dots x_d] \in \mathbb{R}^d$  :

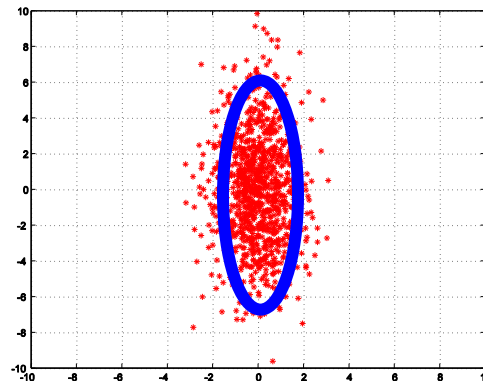
$$P(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2} \sqrt{|\boldsymbol{\Sigma}|}} \exp \left[ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right]$$

- Permet de décrire les corrélations entre dimensions.
- Mais demande de connaître la forme des distributions + limitation à petites dimensions.
- Si modélisation gaussienne et deux classes, la prédiction se réduit à calculer une fonction de degré 2

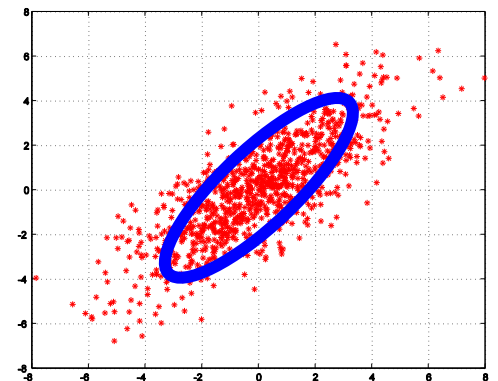
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



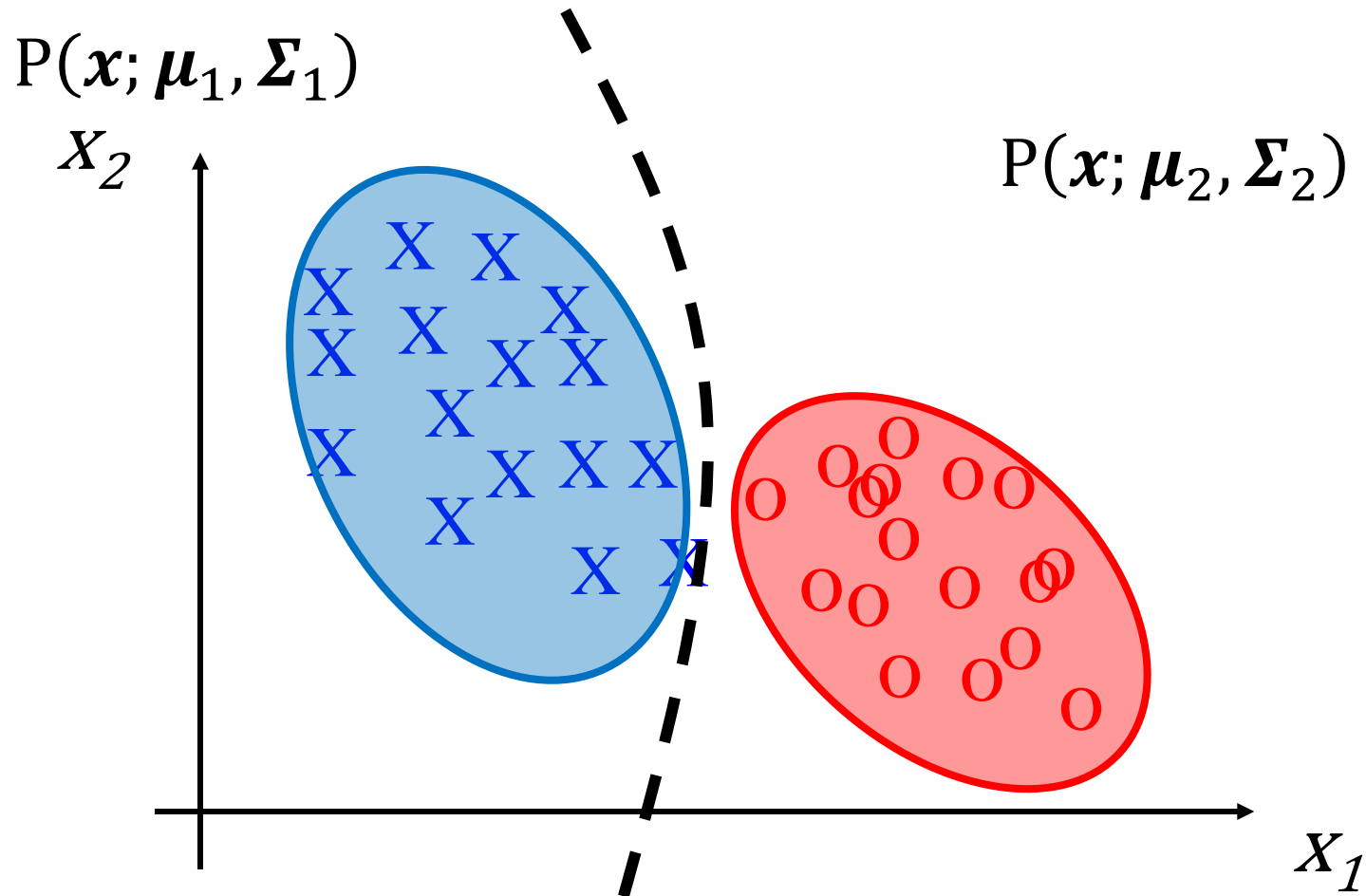
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix}$$



$$\Sigma = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix} = R \begin{bmatrix} 9 & 0 \\ 0 & 1 \end{bmatrix} R^{-1}$$



# Approche gaussienne multivariée



*Séparatrice = Forme quadratique*

$$(\mathbf{x} - \boldsymbol{\mu}_1)' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) - (\mathbf{x} - \boldsymbol{\mu}_2)' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}_2) \geq cste$$

# Approche Bayésienne Naïve

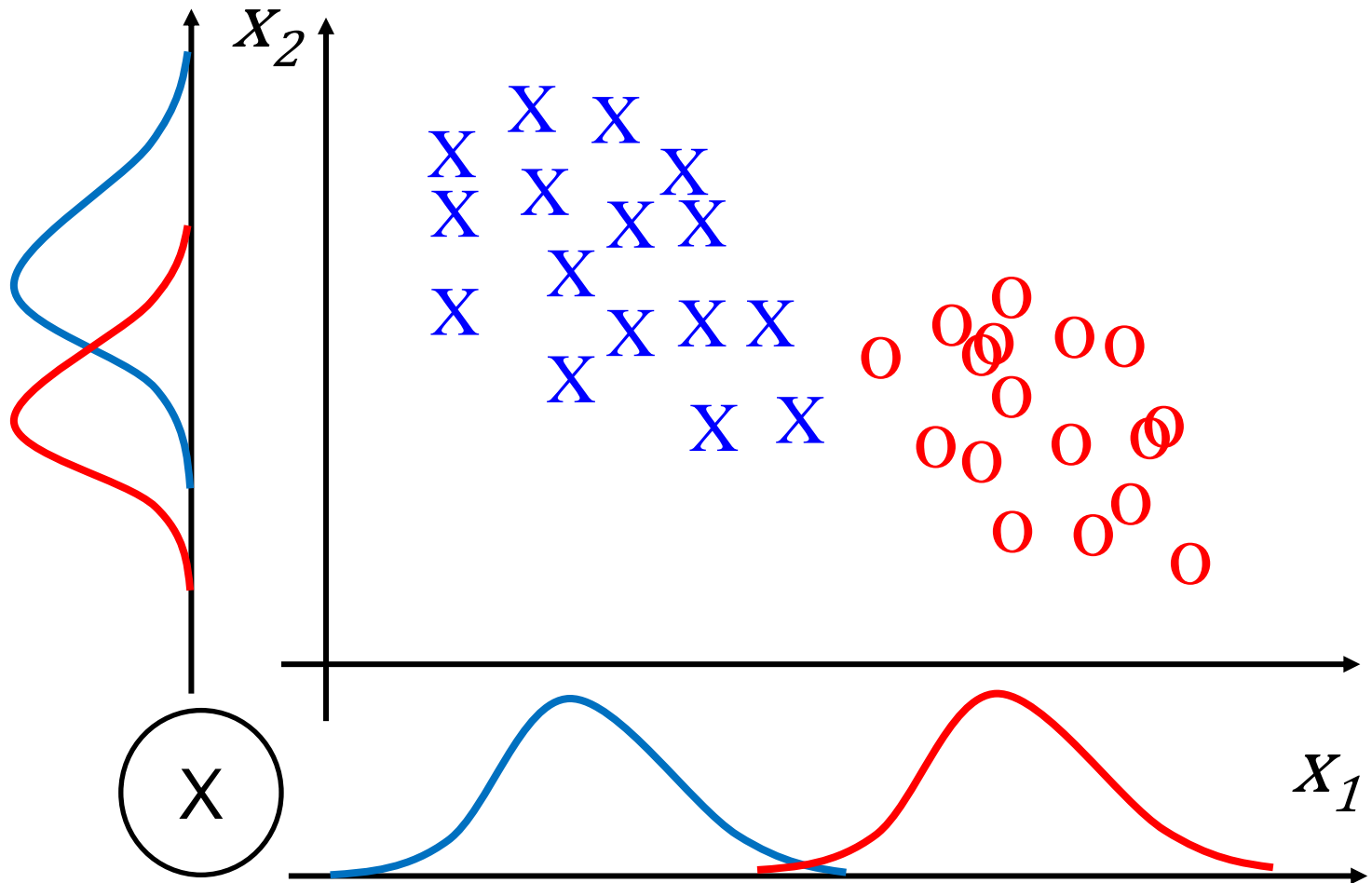
- Calcul de la loi conditionnelle: hypothèse d'indépendance.

$$\begin{aligned}P(x_1, x_2 \dots x_d | y) &= P(x_1 | x_2 \dots x_d, y) P(x_2 \dots x_d | y) \\&= P(x_1 | y) P(x_2 \dots x_d | y) \\&= P(x_1 | y) P(x_2 | y) \dots P(x_d | y)\end{aligned}$$

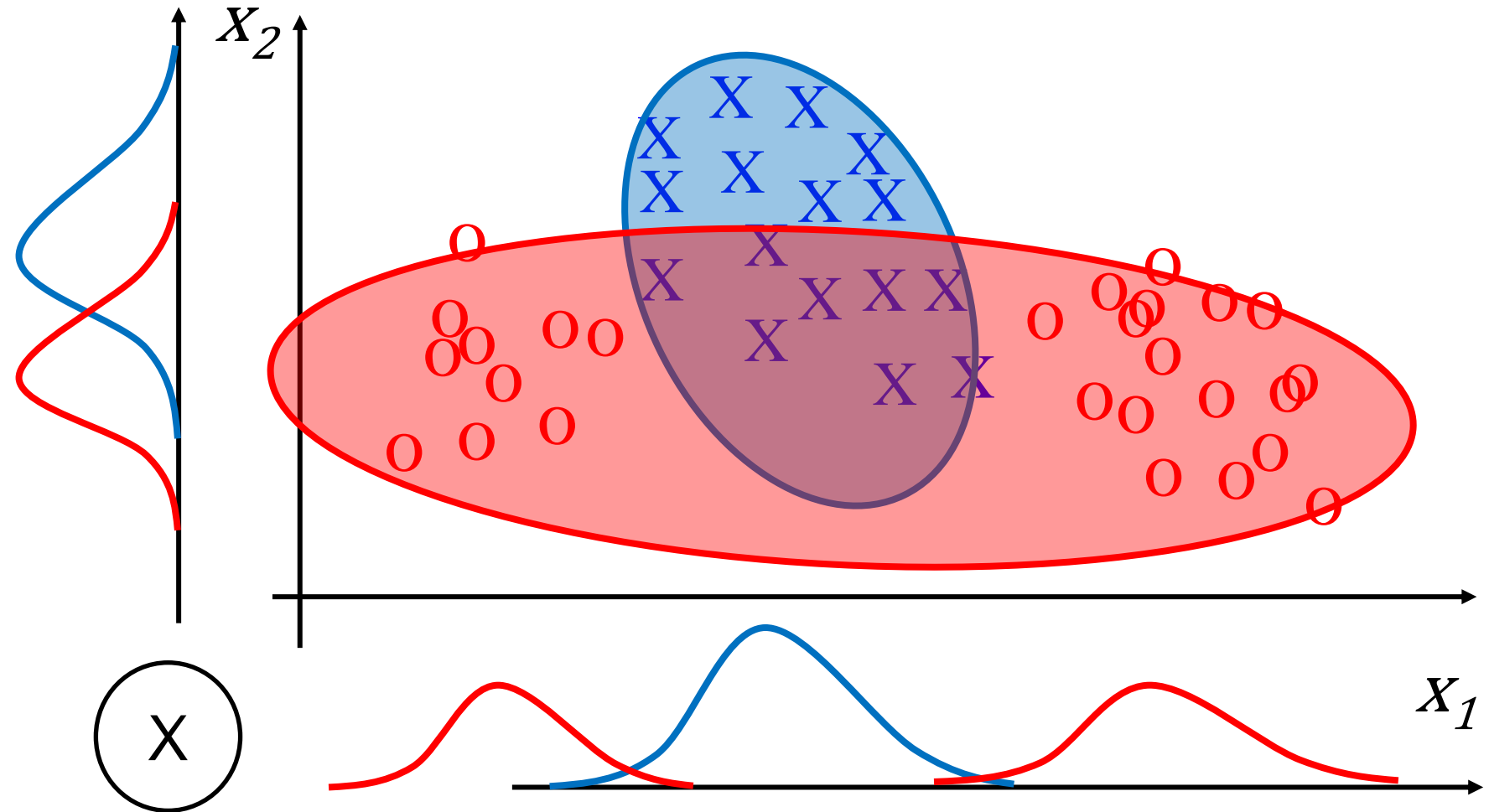
- On calcule la vraisemblance globale dimension par dimension
  - ➔ Problème 1D, modèles plus faciles à estimer (gaussien, binomial, histogrammes, mélange de gaussiennes...)
  - ➔ Permet de traiter des problèmes de plus grande dimension
- En pratique, on calcule plutôt la log-vraisemblance pour des questions de stabilité numérique

$$\begin{aligned}\log P(\mathbf{x} | y) &= \sum_i \log P(x_i | y) \\y^* &= \arg \max_y \log P(\mathbf{x} | y) + \log P(y)\end{aligned}$$

# Approche bayésienne naïve



# Approche bayésienne naïve vs. multivariée



# Approche bayésienne: résumé

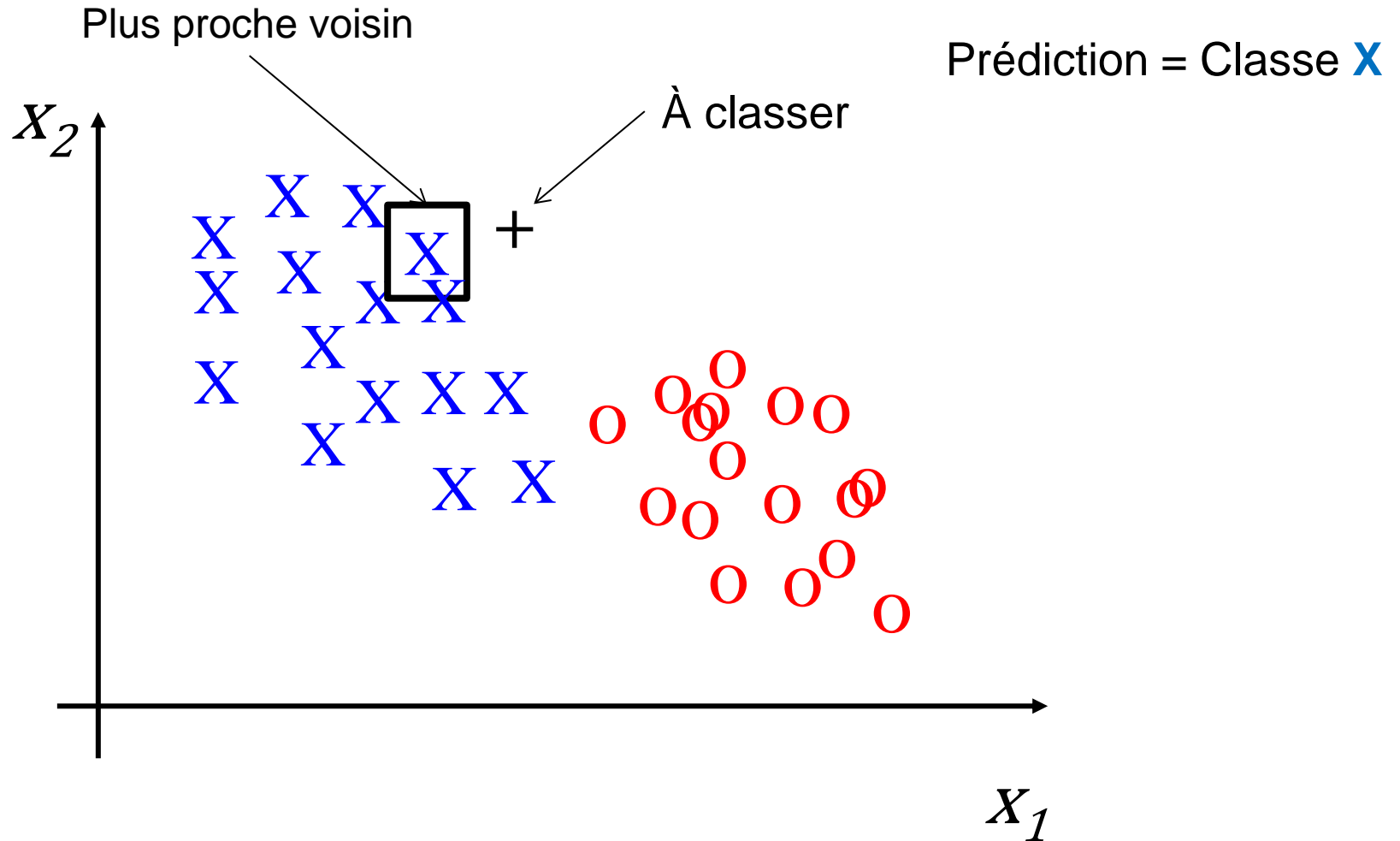
- Théorie probabiliste de la décision → calcul de la loi a posteriori
- Expression de la loi a posteriori:
  - Hypothèse d'indépendance conditionnelle.
  - Modèle gaussien multivarié
- Apprentissage
  - Estimation de lois paramétriques simples
- Prédiction
  - Calcul de log-vraisemblance et max sur hypothèses
- Quand l'utiliser? (limitations)
  - Petits problèmes bien modélisés (gaussien multivarié)
  - Caractéristiques non corrélées (bayésien naïf, mais ça peut aussi marcher si c'est corrélé)

# Plus proche(s) voisin(s)

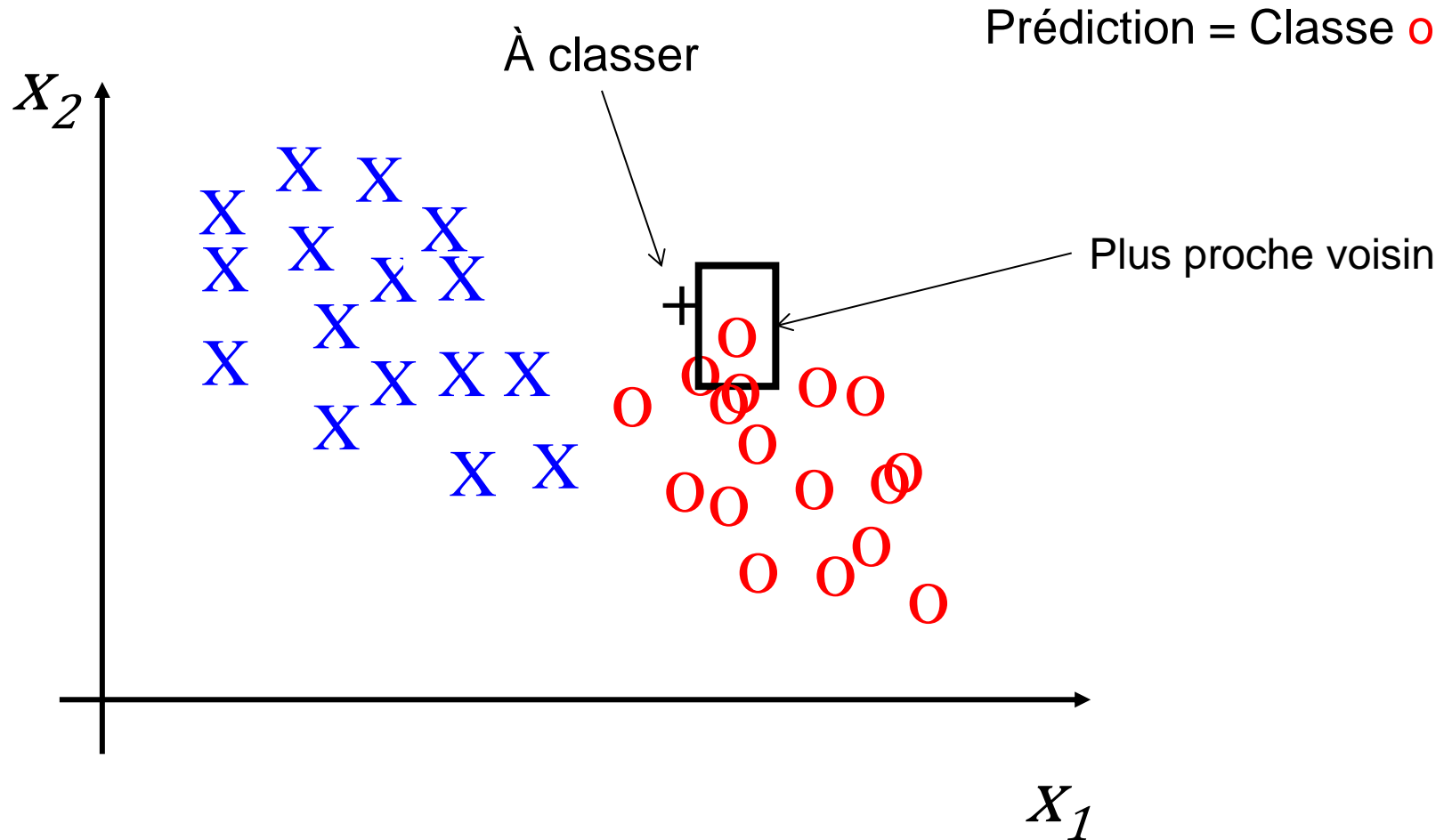
- Principe:
  - Deux échantillons **proches** dans l'espace de représentation ont les mêmes prédictions
  - Pour prédire, il suffit de trouver l'exemple annoté **le plus** proche, et d'associer son annotation (étiquette, valeur...)
- Que veut dire « proche » ?
  - Nécessite la définition d'une métrique ou mesure de similarité  $d(\mathbf{x}, \mathbf{x}')$
  - Plusieurs métriques possibles: distance euclidienne (L2), city-block (L1), Minkowski, Mahalanobis...
  - On peut aussi « apprendre » la métrique ou mesure de similarité
- Que veut dire « le plus proche » ?
  - Base d'échantillons annotés  $\mathcal{L} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots (\mathbf{x}_N, y_N)\}$
  - Recherche de l'échantillon le plus proche:  $i^* = \arg \max_i d(\mathbf{x}, \mathbf{x}_i)$
  - Assigne comme prédiction l'annotation du plus proche:  $y^* = y_{i^*}$



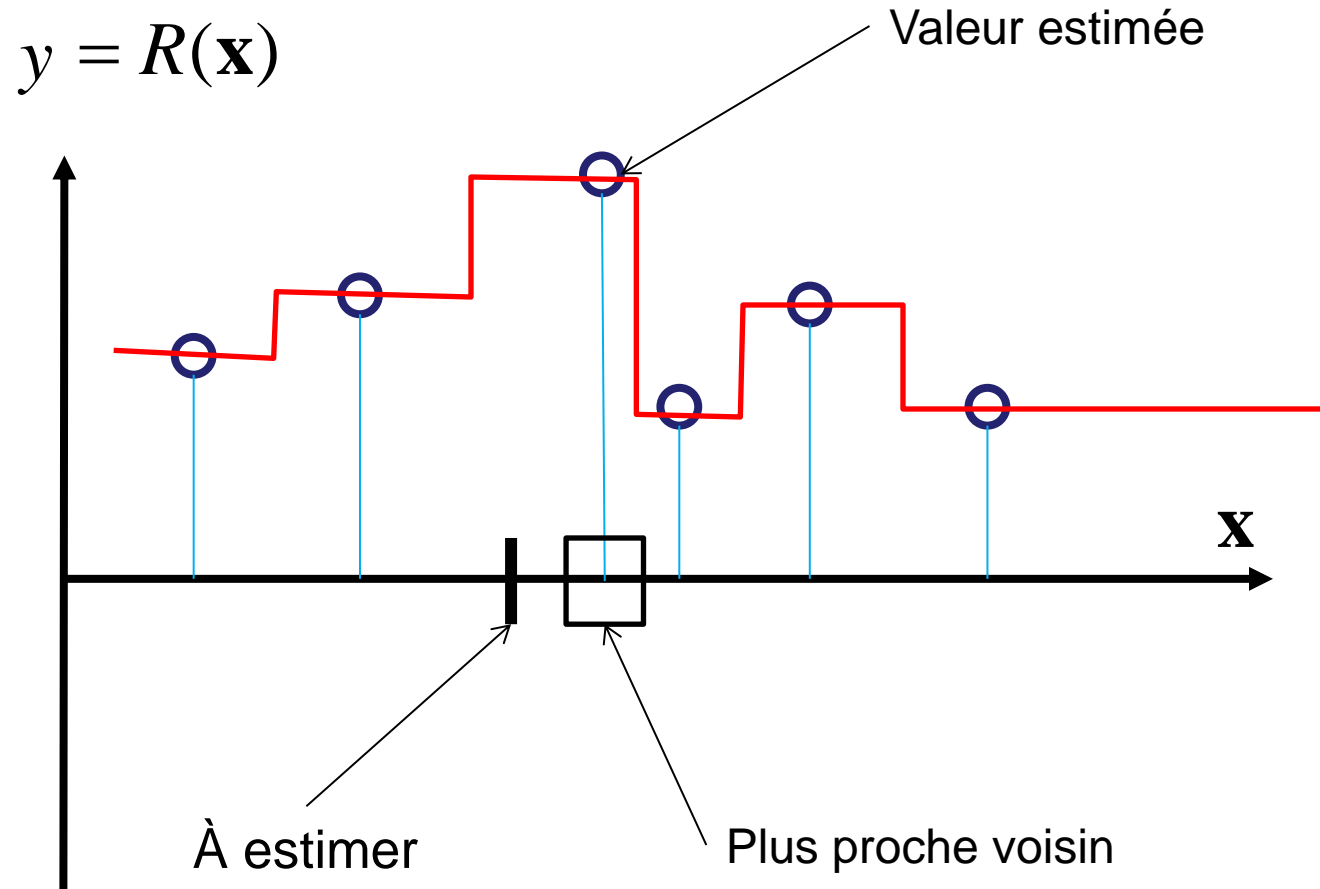
# Classification ppv



# Classification ppv

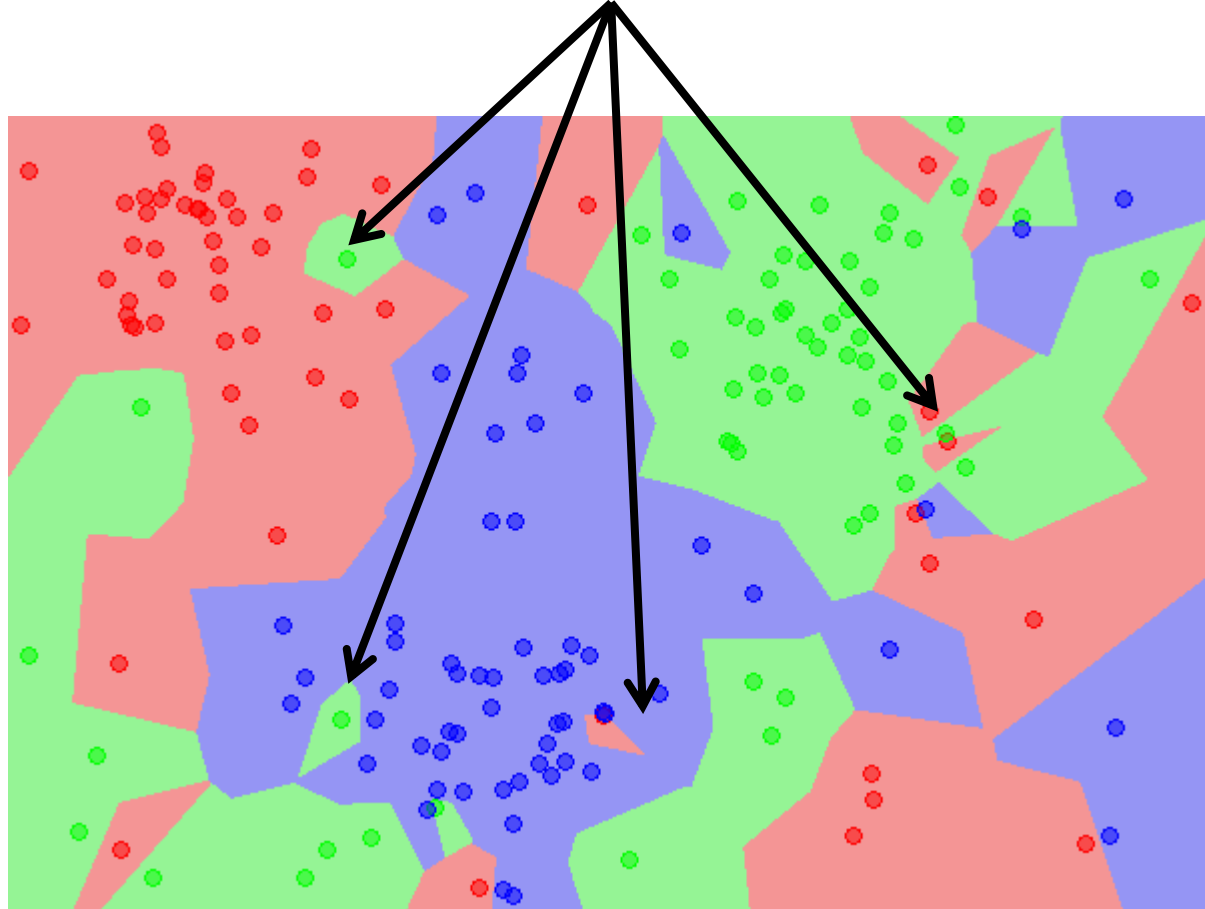


# Régression PPV



# Fonction de classification

Données bruitées → Régions isolées → mauvaise régularité des prédictions



Chaque échantillon définit une région homogène de l'espace de représentation

# k-plus proches voisins (« k-NN »)

- Principe: décision à partir de plusieurs exemples de la base de données d'apprentissage
- On ordonne les échantillons d'apprentissage en fonction de leur distance à la donnée à classer:

$$d(\mathbf{x}, \mathbf{x}_{(1)}) \leq d(\mathbf{x}, \mathbf{x}_{(2)}) \leq \dots \leq d(\mathbf{x}, \mathbf{x}_{(N)})$$

- On choisit les  $k$  plus proches
- On prédit en choisissant la classe recueillant le plus de votes

$$y^* = \arg \max_y \sum_{i=1}^k \delta(y, y_{(i)})$$

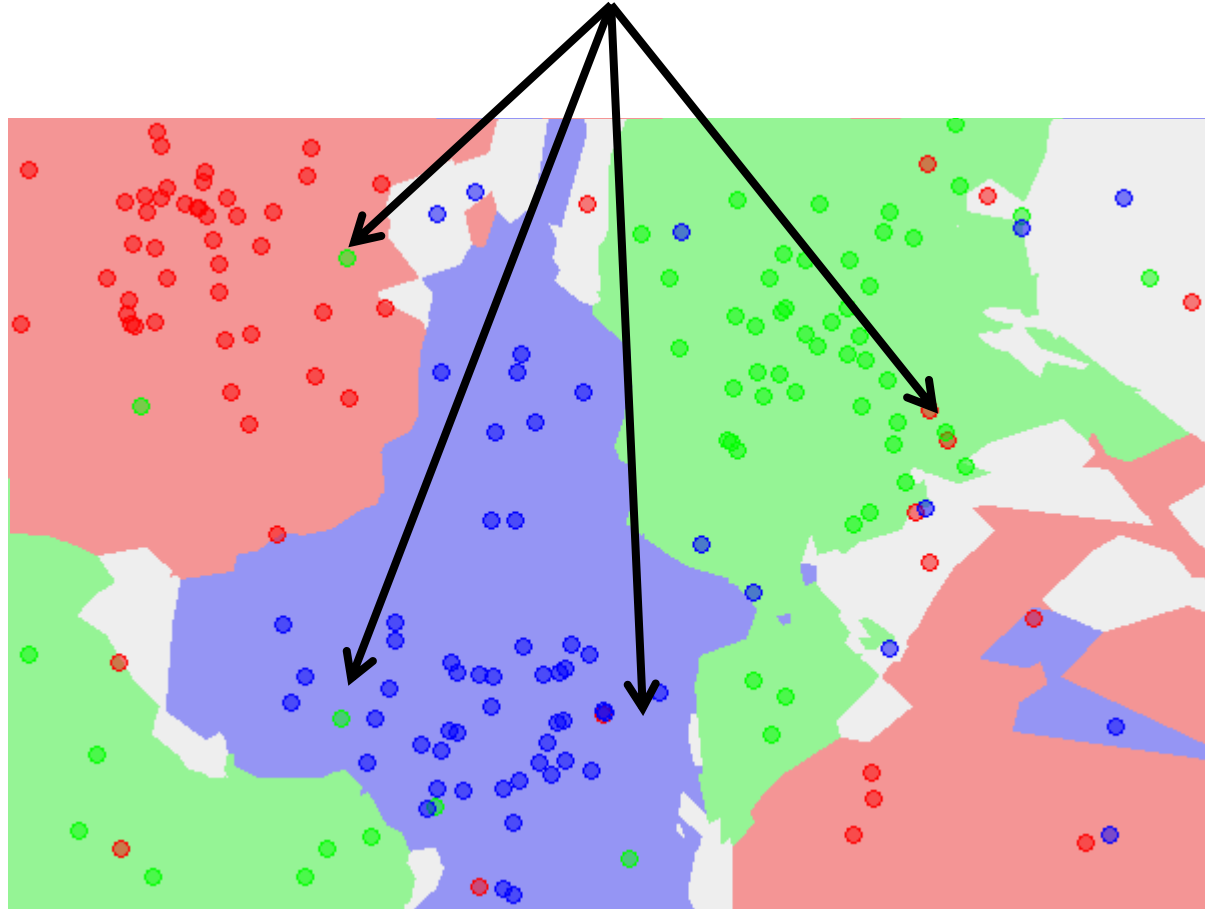
Où  $\delta$  est la fonction de Kronecker (elle vaut 1 si égal, 0 sinon)

- Si pas de max (ambiguïté sur la prédiction) on ne décide pas!
- On peut aussi pondérer les votes:

$$y^* = \arg \max_y \sum_{i=1}^k K(\mathbf{x}, \mathbf{x}_{(i)}) \delta(y, y_{(i)})$$

# Fonction de classification 5 ppv

Données bruitées → Régions isolées → mauvaise régularité des prédictions



Chaque échantillon définit une région homogène de l'espace de représentation

# Propriétés statistiques

Bornes statistiques asymptotiques ( $N \rightarrow \infty$ )

$$E \leq E_{kNN} \leq E \left( 2 - \frac{LE}{L-1} \right)$$

Où  $E$  est l'erreur théorique optimale (Bayes),  $L$  est le nombre de classes et  $E_{kNN}$  est l'erreur des k-ppv.

« L'erreur du k-NN est au plus deux fois moins bonne que l'erreur minimale. »

# Coût de la prédiction du k-ppv

- Calcul de la prédiction dépend pour chaque exemple  $x$  d'un calcul + tri par rapport aux  $N$  exemples de la base:

$$d(x, x_{(1)}) \leq d(x, x_{(2)}) \leq \dots \leq d(x, x_{(N)})$$

- Pour  $N$  et  $d$  grands, coût important de la recherche exhaustive  $O(Nd)$ . Il existe:
  - Des algorithmes efficaces de recherche pour problèmes de tailles moyennes (KDtree)  
J. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transaction on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.
  - Des algorithmes d'approximation pour les grandes bases ( $>10^6$ ).  
Jegou, H., Douze, M., & Schmid, C. (2011). Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1), 117-128.
- Autre manière: pré-calculer les surfaces de séparation entre classes. La complexité de prédiction est alors liée à la complexité de la surface et/ou de son approximation. On verra comment d'autres approches permettent de l'estimer directement.



# La malédiction des grandes dimensions

- Lorsque la dimension  $d$  de l'espace de représentation augmente, les points sont tous aussi proches ou aussi loin.
- On peut montrer, pour une distribution quelconque de  $N$  points tirés de manière indépendante dans  $\mathbb{R}^d$ , que:

$$\lim_{d \rightarrow \infty} E \left[ \frac{d_{max} - d_{min}}{d_{min}} \right] = 0$$

- Ce n'est plus vrai si les points sont corrélés...heureusement!
- On peut interpréter les techniques de Machine Learning comme des moyens de repérer les bonnes corrélations entre données.
- Conséquence pour les approches « plus proches voisins »:
  - Ca ne marche que pour les faibles dimensions
  - Ou il faut **réduire** les dimensions de représentation avant de calculer les distances.

# Comportement des PPV

- Avantages
  - Schéma flexible, facile à mettre en œuvre, dépendant de la définition d'une similarité entre données.
  - Bonnes propriétés statistiques ( $N \rightarrow \infty$ )
- Mais...
  - Temps de calcul prohibitif pour grandes bases
    - Algorithmes efficaces de recherche optimaux ou sous-optimaux
  - Régularité dépend des données, pas de l'apprentissage
    - Le k-PPV (« kNN ») pour lisser et réduire le bruit
  - Malédiction des grandes dimensions (« Curse of dimensionality »)
    - Réduire la dimension de représentation (cf. cours N°5)

# « Plus proches voisins »: résumé

- Hypothèse de régularité = Si observations proches, même comportement
- Deux questions:
  - Que veut dire « proche »?
  - Comment trouver les plus proches?
- Apprentissage
  - Aucun
- Prédiction
  - Tri des distances aux échantillons + vote
- Quand l'utiliser? (limitations)
  - Efficace sur petits problèmes (dimensions & nombre d'exemples)
  - Pb du « curse of dimensionality » + temps de calcul
  - Disposer d'une mesure de similarité adaptée aux données

# Références

- K. Fukunaga, Introduction to Statistical Pattern Recognition (Second Edition), Academic Press, New York, 1990.
- P.A. Devijver and J. Kittler, Pattern Recognition, a Statistical Approach, Prentice Hall, Englewood Cliffs, 1982)
- R.O. Duda and P.E. Hart, Pattern classification and scene analysis, John Wiley & Sons, New York, 1973.
- L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, Classification and regression trees, Wadsworth, 1984.
- S. Haykin, Neural Networks, a Comprehensive Foundation. (Macmillan, New York, NY., 1994)
- L. Devroye, L. Györfi and G. Lugosi, A Probabilistic Theory of Pattern Recognition, (Springer-Verlag 1996)
- V. N. Vapnik, The nature of statistical learning theory (Springer-Verlag, 1995)
- C. Bishop, Pattern Recognition and Machine Learning, (Springer-Verlag, 2006).
- Jerome H. Friedman, Robert Tibshirani et Trevor Hastie, The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Springer-Verlag 2009).
- Ian Goodfellow and Yoshua Bengio and Aaron Courville, Deep Learning, An MIT Press book (<http://www.deeplearningbook.org>)
- Kevin Murphy, Machine Learning: a Probabilistic Perspective, (MIT Press, 2013)
- Hal Daumé III, A Course in Machine Learning (<http://ciml.info/>)

# Bases de données

- UCI Repository: <http://www.ics.uci.edu/~mlearn/MLRepository.html>
- UCI KDD Archive: <http://kdd.ics.uci.edu/summary.data.application.html>
- Statlib: <http://lib.stat.cmu.edu/>
- Delve: <http://www.cs.utoronto.ca/~delve/>
- Kaggle: <https://www.kaggle.com/>
- Benchmarks (Vision):
  - ImageNet: <http://image-net.org/>
  - MS COCO: <http://cocodataset.org/>
  - MNIST et plus:  
[http://rodrigob.github.io/are\\_we\\_there\\_yet/build/classification\\_datasets\\_results.html](http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html)
  - CV on line: <https://computervisiononline.com/datasets>

# Journaux

- Journal of Machine Learning Research [www.jmlr.org](http://www.jmlr.org)
- Machine Learning
- Neural Computation
- Neural Networks
- IEEE Transactions on Neural Networks
- IEEE Transactions on Pattern Analysis and Machine Intelligence
- Annals of Statistics
- Journal of the American Statistical Association
- ...

# Conférences

- International Conference on Machine Learning (ICML)
- European Conference on Machine Learning (ECML)
- Neural Information Processing Systems (NIPS)
- Uncertainty in Artificial Intelligence (UAI)
- Computational Learning Theory (COLT)
- International Joint Conference on Artificial Intelligence (IJCAI)
- International Conference on Neural Networks (Europe)
- Conference of the American Association for Artificial Intelligence (AAAI)
- IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
- European Conference on Computer Vision (ECCV)
- International Conference on Computer Vision (ICCV)
- IEEE International Conference on Data Mining (ICDM)
- ...

# Cours & tutoriaux

- Des tutoriaux nombreux sur le web
- MOOC Stanford:  
<https://fr.coursera.org/courses?query=machine%20learning%20andrew%20ng>
- Cours S. Mallat (plutôt matheux): [https://www.college-de-france.fr/site/stephane-mallat/\\_course.htm](https://www.college-de-france.fr/site/stephane-mallat/_course.htm)



# Logiciels

- Environnement génériques: Matlab, ScikitLearn
- Environnements Deep Learning: Tensor Flow, Pytorch, mxnet, CNTK...
- Beaucoup de codes sur GitHub



# A retenir pour aujourd'hui

- Apprentissage = programmer à partir des données
- Plusieurs types d'apprentissage
- Démarche générique:
  - Constitution d'une base d'apprentissage
  - Analyse préliminaire des données
  - Conception du modèle
  - Optimisation
  - Evaluation
- Deux approches élémentaires:
  - Modélisation bayésienne
  - Plus proches voisins

# Le TD1

- Partie 1: Les deux approches élémentaires sur une première base
  - Programmation Python
  - Application de la démarche
- Partie 2: Utilisation de la bibliothèque scikit-learn
  - Les deux approches sur une autre base

# Utilisation de Colab

- Environnement de développement Python (Notebook)
- Ressources de calcul distantes (GPU)
- C'est proposé par Google
- <https://colab.research.google.com/>

## Etapas

- Se créer un gmail (ou utiliser le votre)
- Télécharger le TD sur le gdrive du compte
- Se connecter sur Colab
- Ouvrir le Notebook du TD (td1\_knn\_bayesien.ipynb )
- Monter le drive dans Colab (première étape du TD)
- Modifiez directement le notebook, et sauvegardez-le régulièrement.