

Apprentissage Automatique

Introduction

Stéphane Herbin

`stephane.herbin@onera.fr`

Aujourd'hui

- Introduction générale:
 - Exemples, définitions, problématiques, approches, vocabulaire
- Un principe fondamental: la généralisation
- Deux approches élémentaires à connaître:
 - Modélisation bayésienne
 - k plus proches voisins (kNN)

« 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

Pourquoi le « Machine Learning » ?

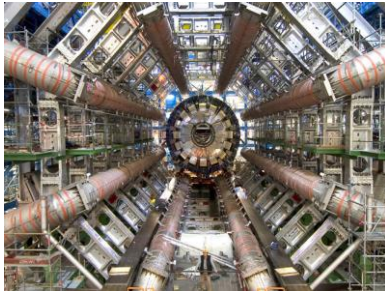
- Thème à la mode: Intelligence Artificielle, « deep learning », « big data »...
- Raison épistémologique
 - On ne sait pas modéliser les problèmes complexes
... mais on dispose d'exemples en grand nombre représentant la variété des situations
 - « Data driven » vs. « Model Based »
- Raison scientifique
 - L'apprentissage est une faculté essentielle du vivant
- Raison économique
 - La récolte de données est plus facile que le développement d'expertise

Domaines techniques utilisant du ML

- ML comme outil de conception
 - Vision & Reconnaissance des formes
 - Traitement du langage
 - Traitement de la parole
 - Robotique
 - « Data Mining »
 - Recherche dans BDD
 - Recommandations
 - Marketing...
- ML comme outil explicatif
 - Neuroscience
 - Psychologie
 - Sciences cognitives

Données = carburant du ML

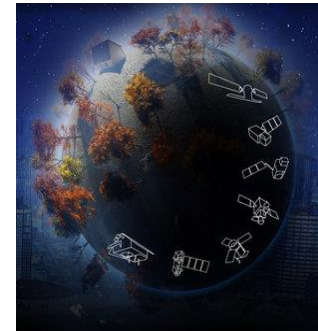
CERN /
Large Hadron Collider
~70 Po/an



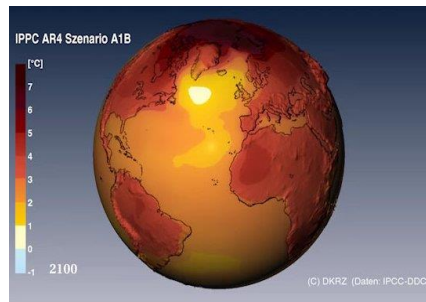
Google :
24 PetaOctets/jour



Copernicus :
> 1Po/an



DKRZ (Climat)
500 Po



Google



BIG DATA

Square Kilometer Array
1376 Po/an (en 2024)



Apprentissage automatique : applications

Anti-Spam (*Classifieur Bayésien*)



1997 : *DeepBlue bat Kasparov*

2017: *Alpha GO bat Ke Jie*

2019: *AlphaStar champion de StarCraft*



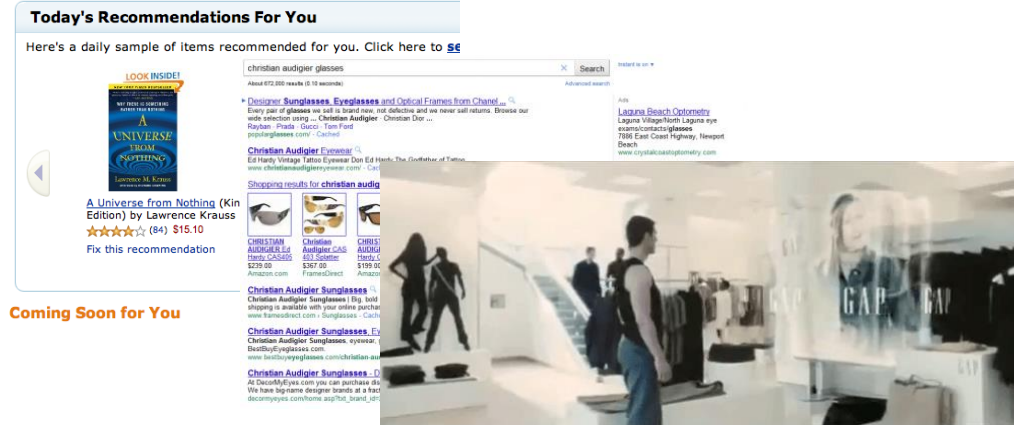
Tri postal automatique (*détection de chiffres manuscrits par réseaux de neurones*)



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)

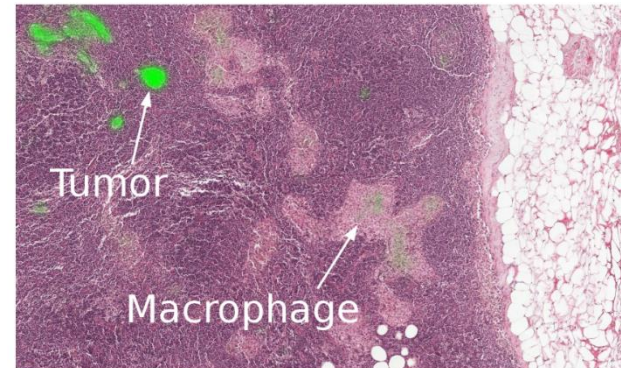


Apprentissage automatique : applications

Chat Bots
(*Réseaux de neurones*)



Diagnostic médical
(*Réseaux de neurones*)

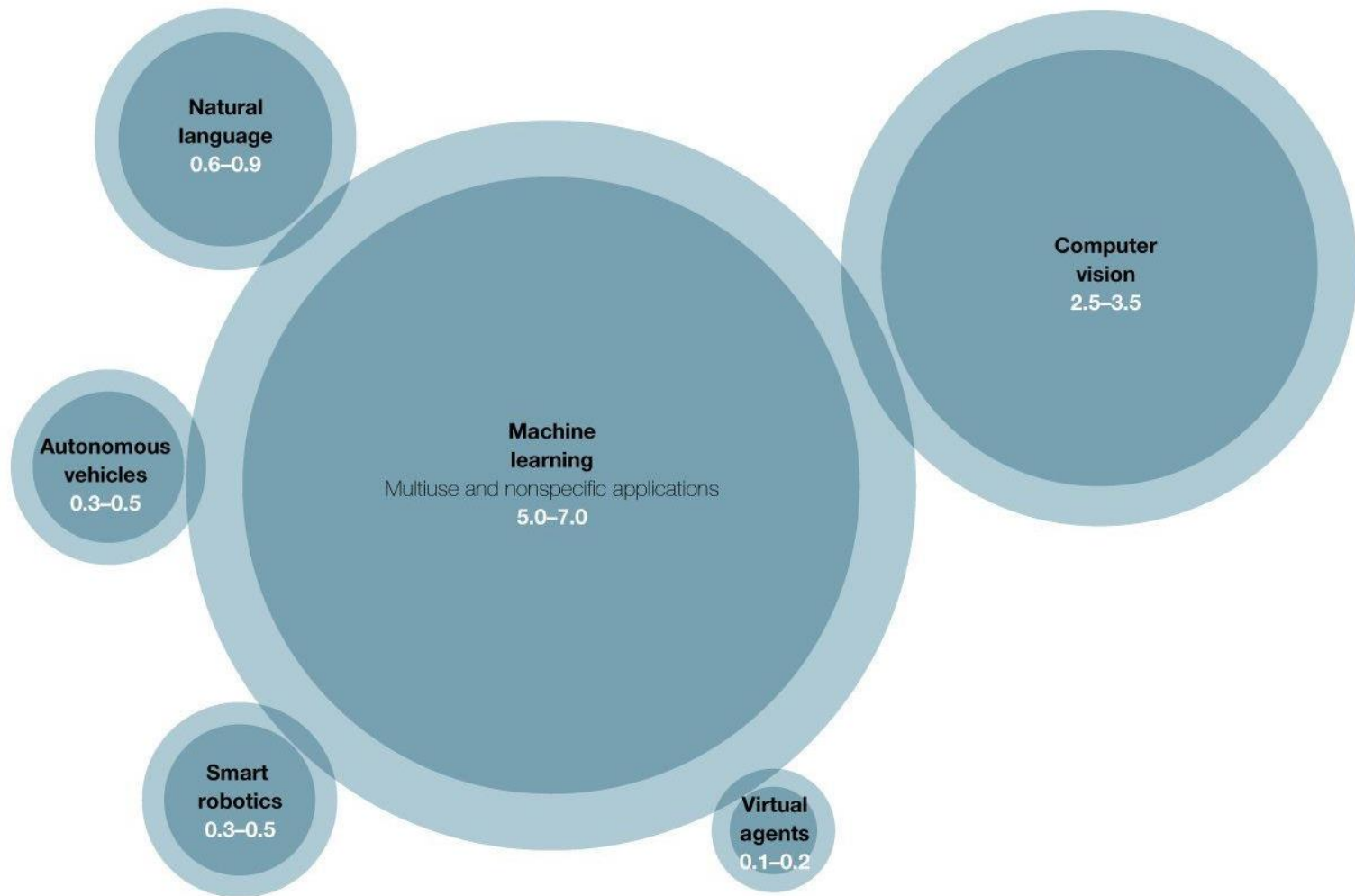


Traduction multi-lingue
(*Réseaux de neurones*)



External investment in AI-focused companies by technology category, 2016¹

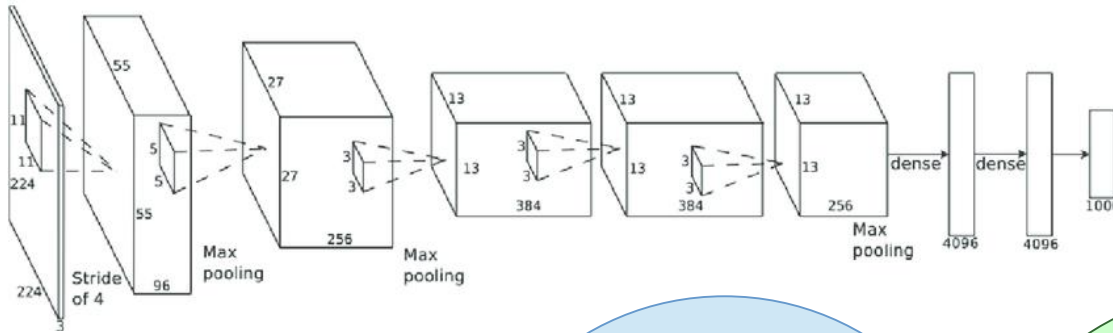
\$ billion



¹ 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



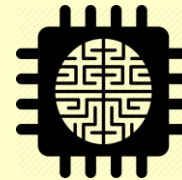
Données



Algorithmes



Moyens de calcul



Logiciels



Une rupture scientifique et technologique en apprentissage

MACHINE LEARNING

Problématique générale

Dans ce cours

L'apprentissage automatique est:

- une démarche de **conception** d'un **prédicteur**
- par une modélisation ou programmation **non explicite** à partir **d'exemples** (signaux, images, texte, mesures...)

Formalisation

- Donnée à interpréter (x)
 - Mesures, texte, image, enregistrement, vidéo ou caractéristiques extraites de ...
- Prédiction (y)
 - Décision, choix, action, réponse, préférence, groupe, commande, valeur...
- Echantillons ($D = \{(x_i, y_i)\}$)
 - Exemples de données et de (bonnes) prédictions
 - « Base d'apprentissage »: D
- Hypothèse forte: les échantillons contiennent toute l'information exploitable et utile

Formalisation

Prédicteur = Fonction paramétrique de paramètres \mathbf{W}

$$y = F(\mathbf{x}; \mathbf{W})$$

Apprentissage = trouver le \mathbf{W} qui optimise un critère L

$$\mathbf{W} = \arg \min_{\mathbf{W}'} L(\mathbf{D}, \mathbf{W}')$$

A partir de la base d'apprentissage $\mathbf{D} = \{(\mathbf{x}_i, y_i)\}$

Le critère L est « empirique » = il dépend de données

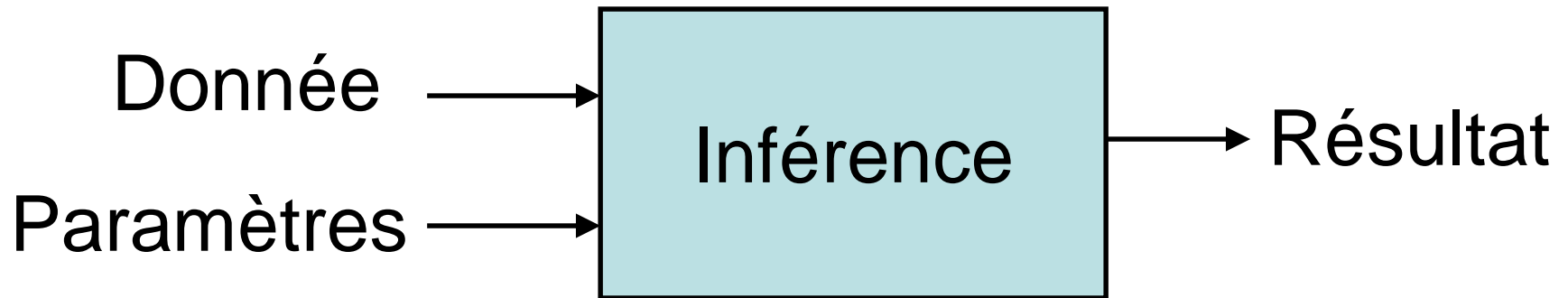
$$\text{ex. } L(\mathbf{D}, \mathbf{W}) = \sum_i \|y_i - F(\mathbf{x}_i; \mathbf{W})\|^2$$

Deux phases

Apprentissage (train)



Prédiction (test)



Exemple: Reconnaissance de chiffres manuscrits



- Comment définir les éléments ?

$$D, W, x, y$$

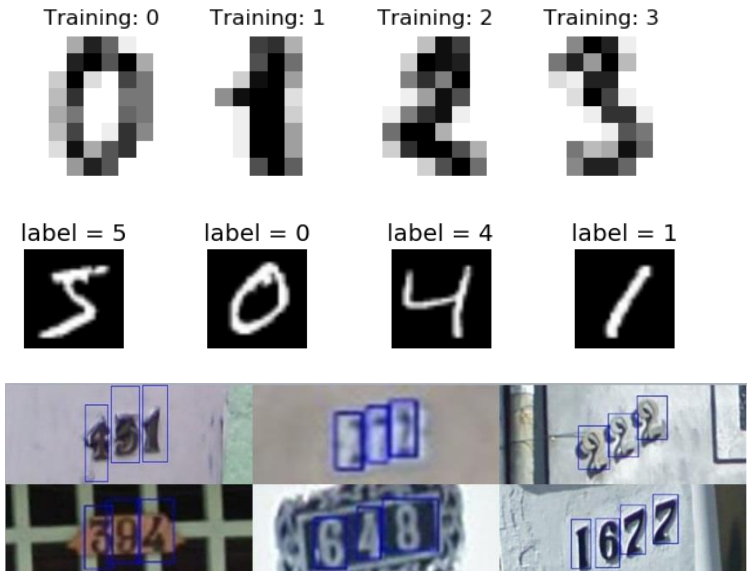
- Les fonctions d'apprentissage et de prédiction?

$$D \mapsto W$$

$$W, x \mapsto y$$

Etape 1: choix de la base de données

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



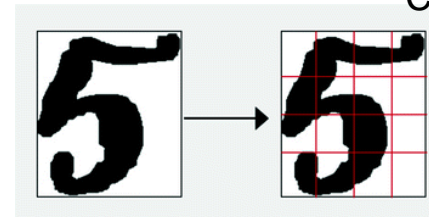
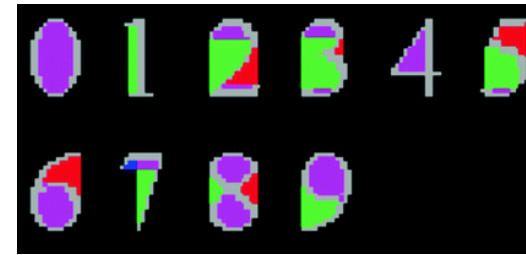
- Il faut la construire:
 - Recueil de données existantes
 - Expérimentations (photos, mesures...)

Etape 2: mise en forme des données



Extraction de
caractéristiques
Pré-traitement

Image (2D)
*grande dimension,
bruitée, hétérogène*



0
0
0
0
0
1
1
0
0
0
1
0
0

x = Vecteur
*petite dimension,
homogène, « propre »*

[Dine et al., 2017]
https://doi.org/10.1007/978-3-319-46568-5_17

Etape 3: choix de l'approche

- Quel type de fonction et de problème d'apprentissage?
 - 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
- Nature fonctionnelle des prédicteurs?
 - Arbres de décision, SVM, Réseaux de neurones...

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 »)

Types de prédictions

- **Classification**

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

- **Régression**

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

- **Structure**

- Graphe des articulations d'une personne

- **Regroupement**

- Photos dans base de données personnelle

- **Texte**

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

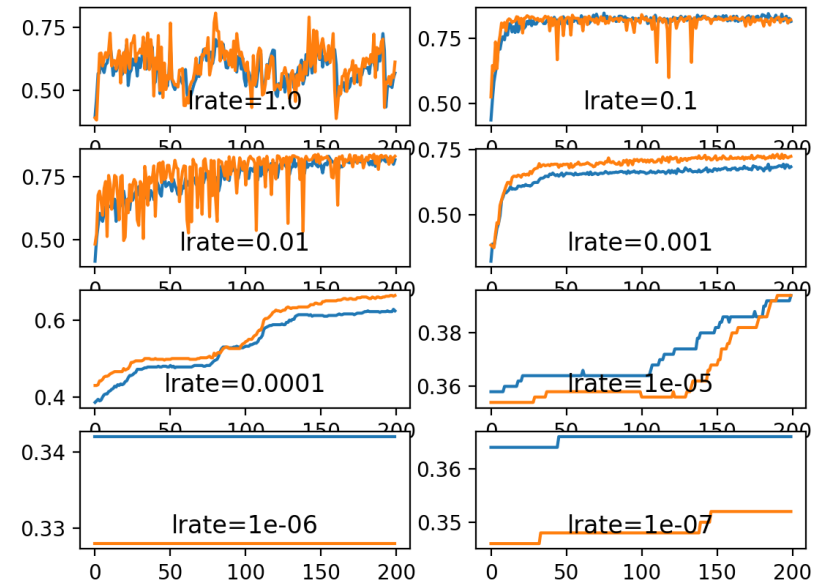
Nature fonctionnelle du prédicteur

- 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.

Etape 4: optimisation

Apprentissage =

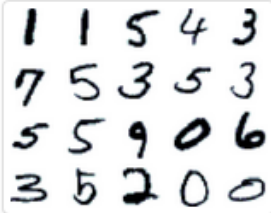
- définir un espace fonctionnel et un critère paramétrique (coût, énergie...)
- appliquer un optimiseur et régler ses paramètres
- vérifier que l'apprentissage se passe bien
 - évaluation de la capacité de généralisation
 - convergence



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








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

Métriques d'évaluation

- 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

Résumé des étapes de conception

1. Constituer des bases de données
2. Préparer les données: Analyser, visualiser, prétraiter, transformer, extraire
3. Concevoir le modèle (type de prédicteur, principe d'apprentissage)
4. Définir un critère et Optimiser (l'apprentissage proprement dit)
5. Evaluer

Travailler avec des données

Deux activités complémentaires:

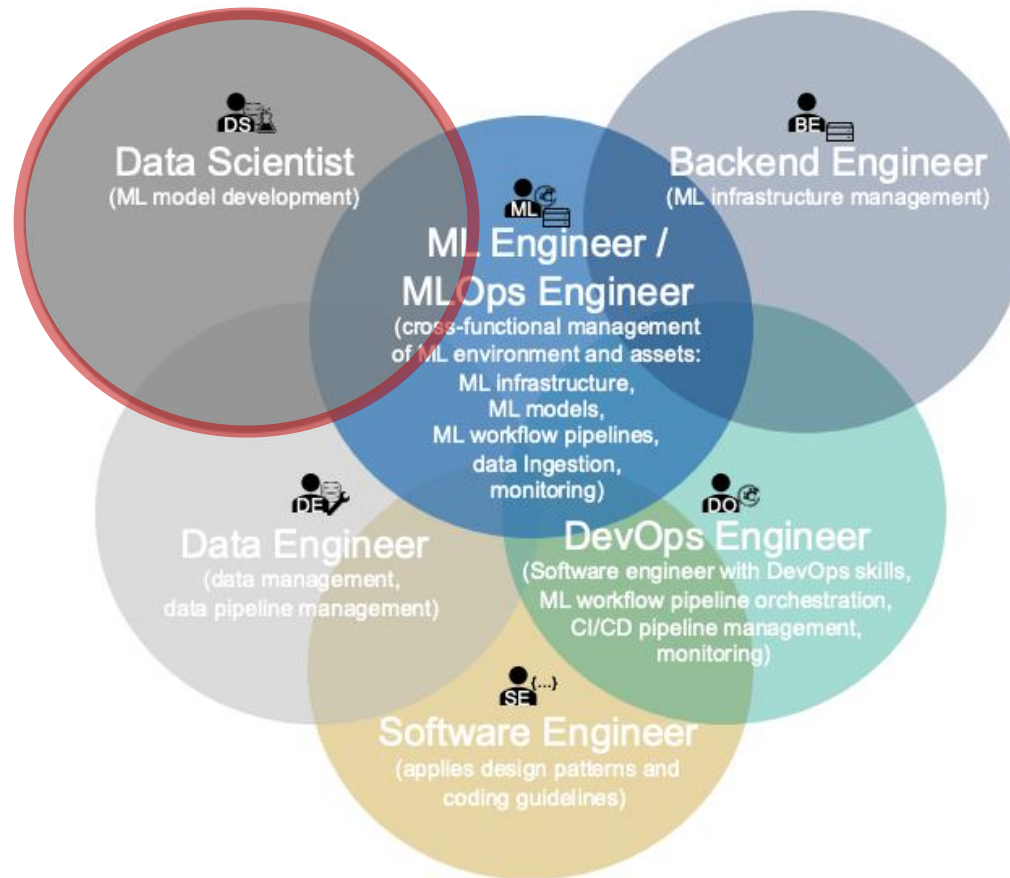
Préparer les données

- Etape coûteuse mais indispensable
- Objectif: rendre possible l'apprentissage avec des données:
 - Propres, homogènes, recalées, calibrées, organisées, facilement accessibles, renseignées...
- « Data engineering » (un nouveau métier!)

Transformer les données

- Objectif: Extraire l'information des données, leurs caractéristiques (« features »), construire leur forme

Les métiers du ML et des données



Kreuzberger, D., Kühl, N., & Hirschl, S. (2022). Machine Learning Operations (MLOps): Overview, Definition, and Architecture. *arXiv preprint arXiv:2205.02302*.

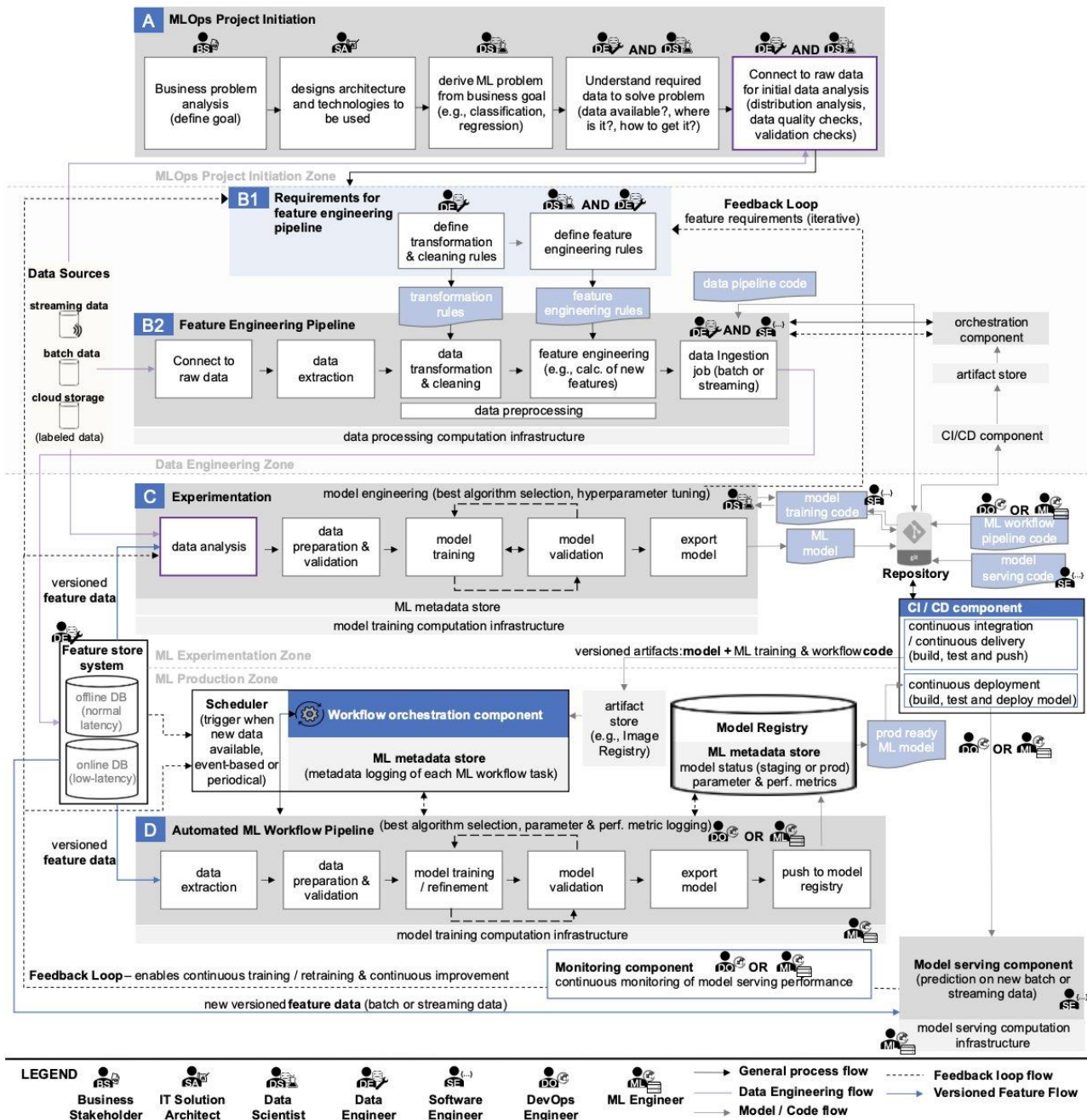
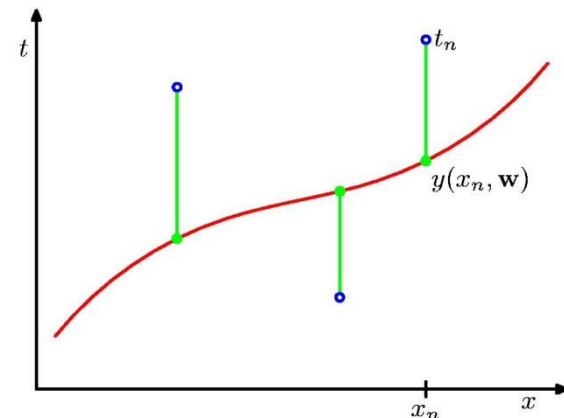
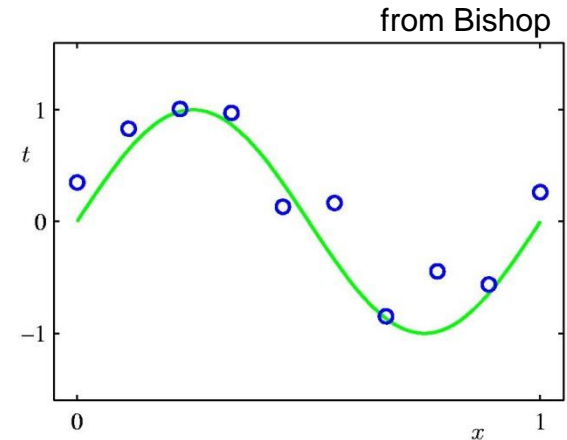


Figure 4. End-to-end MLOps architecture and workflow with functional components and roles

UN CONCEPT FONDAMENTAL: LA GÉNÉRALISATION

Exemple : régression polynomiale

- La courbe verte est la véritable fonction à estimer (non polynomiale)
- Les données sont uniformément échantillonnées en x mais bruitées en y .
- L'erreur de régression est mesurée par la distance au carré entre les points vrais et le polynôme estimé.



Modèles linéaires généralisés ($y \in \mathbb{R}$)

$$y = F(x, \mathbf{w}) = w_0 + w_1\phi_1(x) + w_2\phi_2(x) + \dots + w_M\phi_M(x)$$

- Prédiction utilise des **fonctions de base** encodant les données source (« features »): $\Phi(x) = x^k$
- **Apprentissage** = Maximum de Vraisemblance:

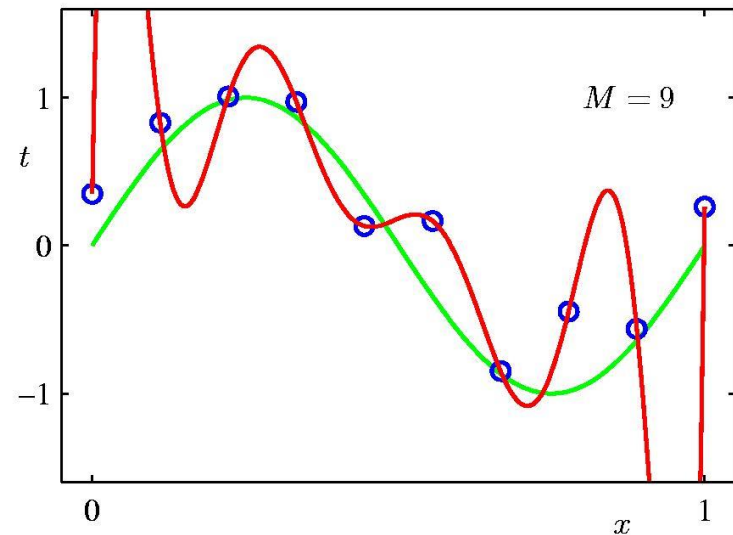
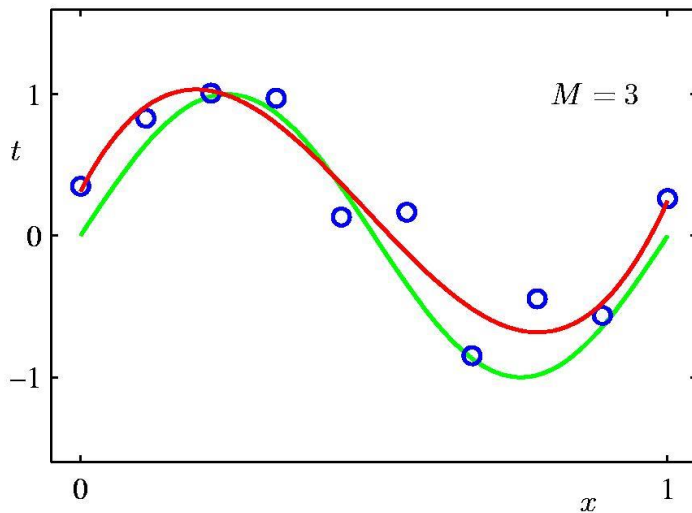
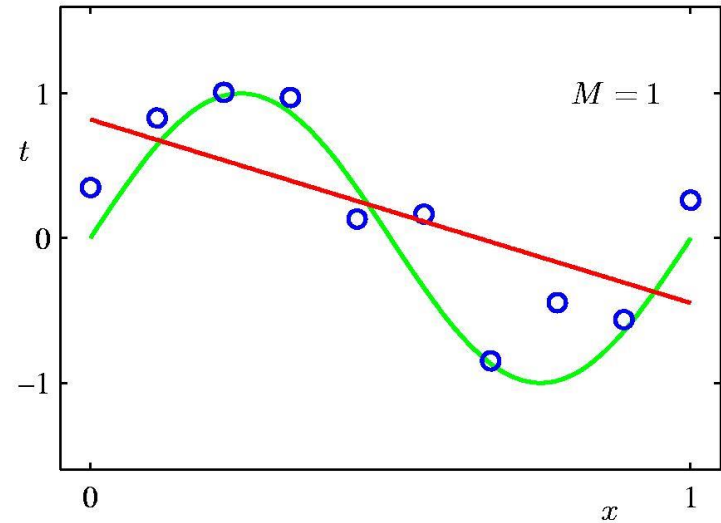
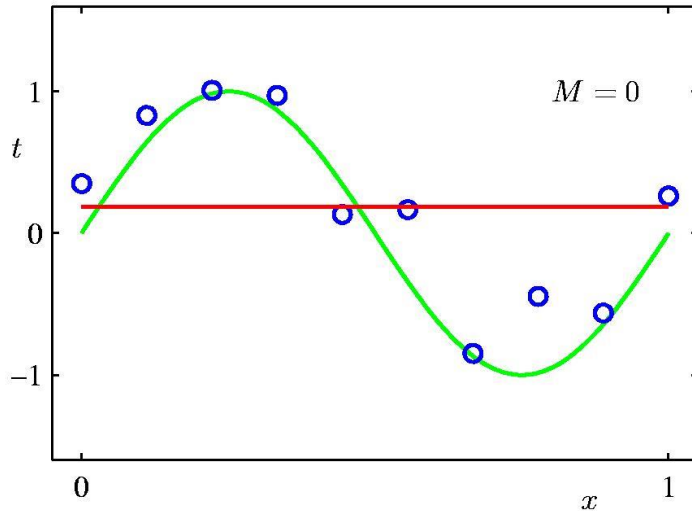
$$\mathbf{w}_{\text{ML}} = \left(\Phi^T \Phi \right)^{-1} \Phi^T \mathbf{t}$$

$$\Phi = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}.$$

- Que vaut cet apprentissage?

Régression polynomiale: $\Phi(x) = x^k$

from Bishop



“Training vs. Test”

Erreur de régression est calculée sur des données $\mathbf{D} = \{(\mathbf{x}_i, y_i)\}$:

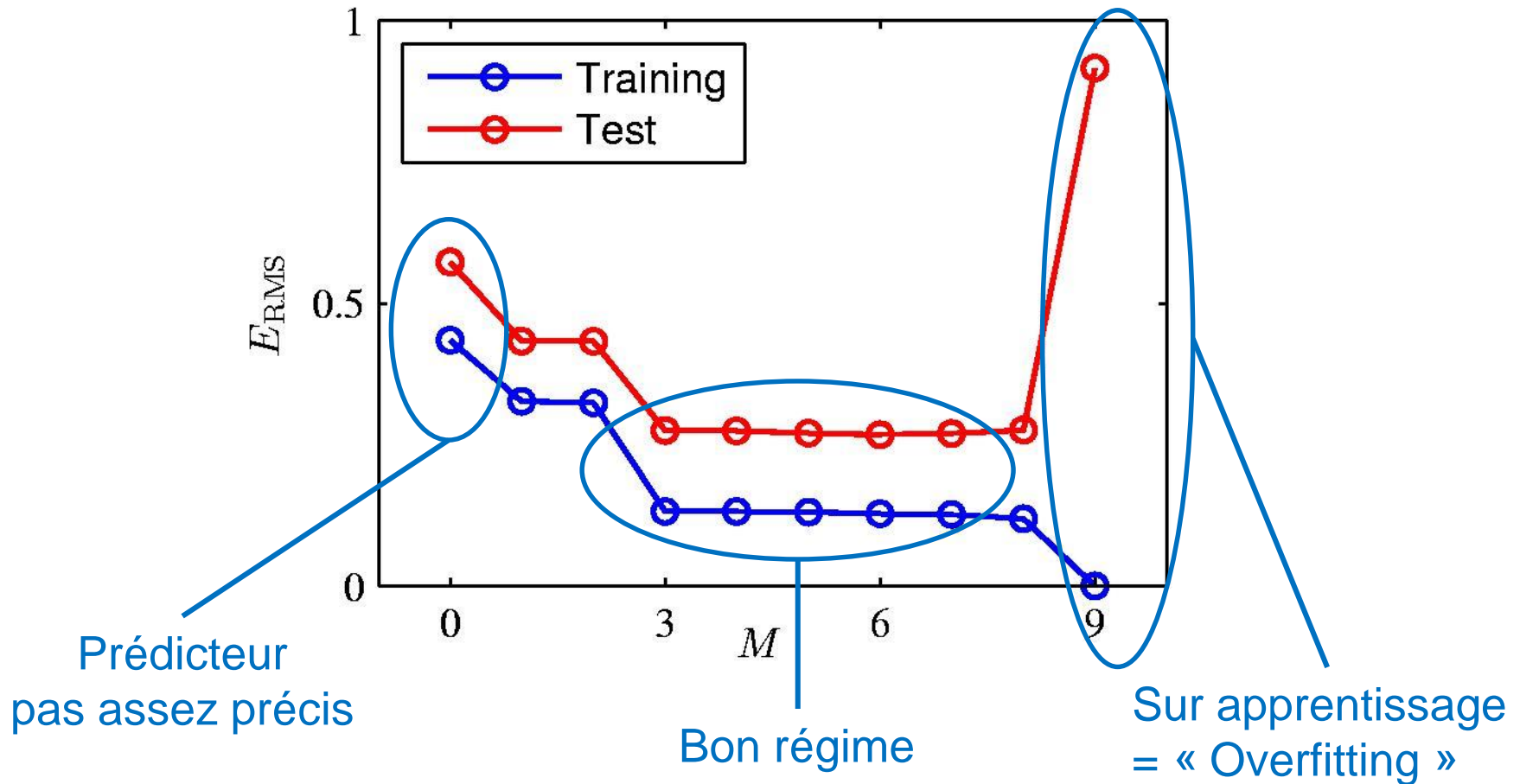
$$\mathcal{E}_{\text{RMS}}(\mathbf{w}) = \sum_{i=1}^N (\mathbf{w}^T \cdot \phi(x_i) - y_i)^2$$

Mais quelles données?

- Données d'apprentissage (Training):
 - c'est un moyen de modélisation
- Données opérationnelles (Test):
 - c'est la situation réelle
 - Celles pour lesquelles on veut une bonne prédiction

Comportement des erreurs

$$\mathcal{E}_{\text{RMS}}(\mathbf{w}) = \sum_{i=1}^N (\mathbf{w}^T \cdot \phi(x_i) - y_i)^2$$



Erreur de généralisation

- La mesure de bon fonctionnement est l'erreur sur des données nouvelles
généraliser \neq mémoriser (par cœur)
- Problème: les données nouvelles sont par nature inconnues! (sinon, elles seraient utilisées)
- ➔ Il est nécessaire de faire des hypothèses sur leur nature et sur le modèle de prédiction.

Deux phénomènes à contrôler (éviter)

- **Simplisme:** modélisation trop grossière pour rendre compte de la variété des données
 - Erreur d'apprentissage et de test importantes
- **Sur-apprentissage (« Overfitting »):** modèle trop complexe se spécialisant sur les données d'apprentissage
 - Ecart entre erreur d'apprentissage et erreur de test

Construire son chantier d'apprentissage

- Préparer les données
 - Simplifier/compléter/formater/homogénéiser/calibrer...
- Diviser en deux ensembles:
 - **Apprentissage** (« Train ») pour optimiser les paramètres du modèle.
 - **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 $P(y)$

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

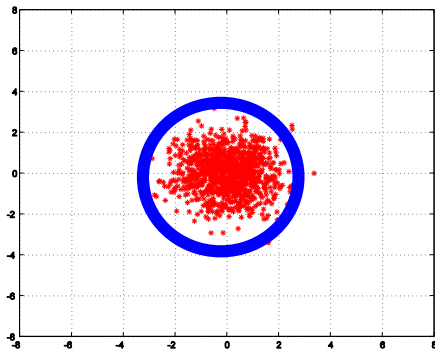
Approche Bayésienne multivariée

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

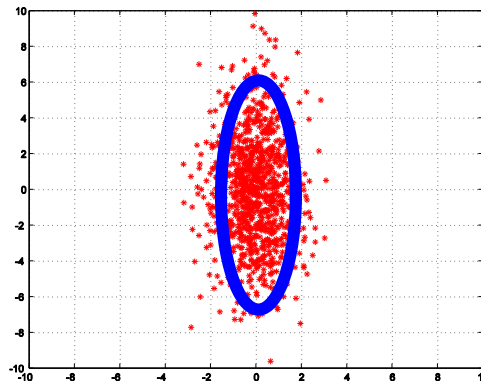
$$P(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{d/2} \sqrt{|\boldsymbol{\Sigma}|}} \exp \left[-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right]$$

- 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

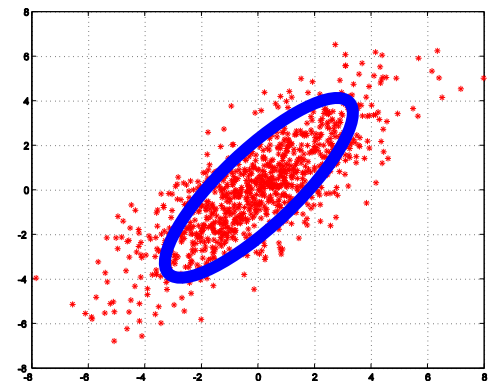
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



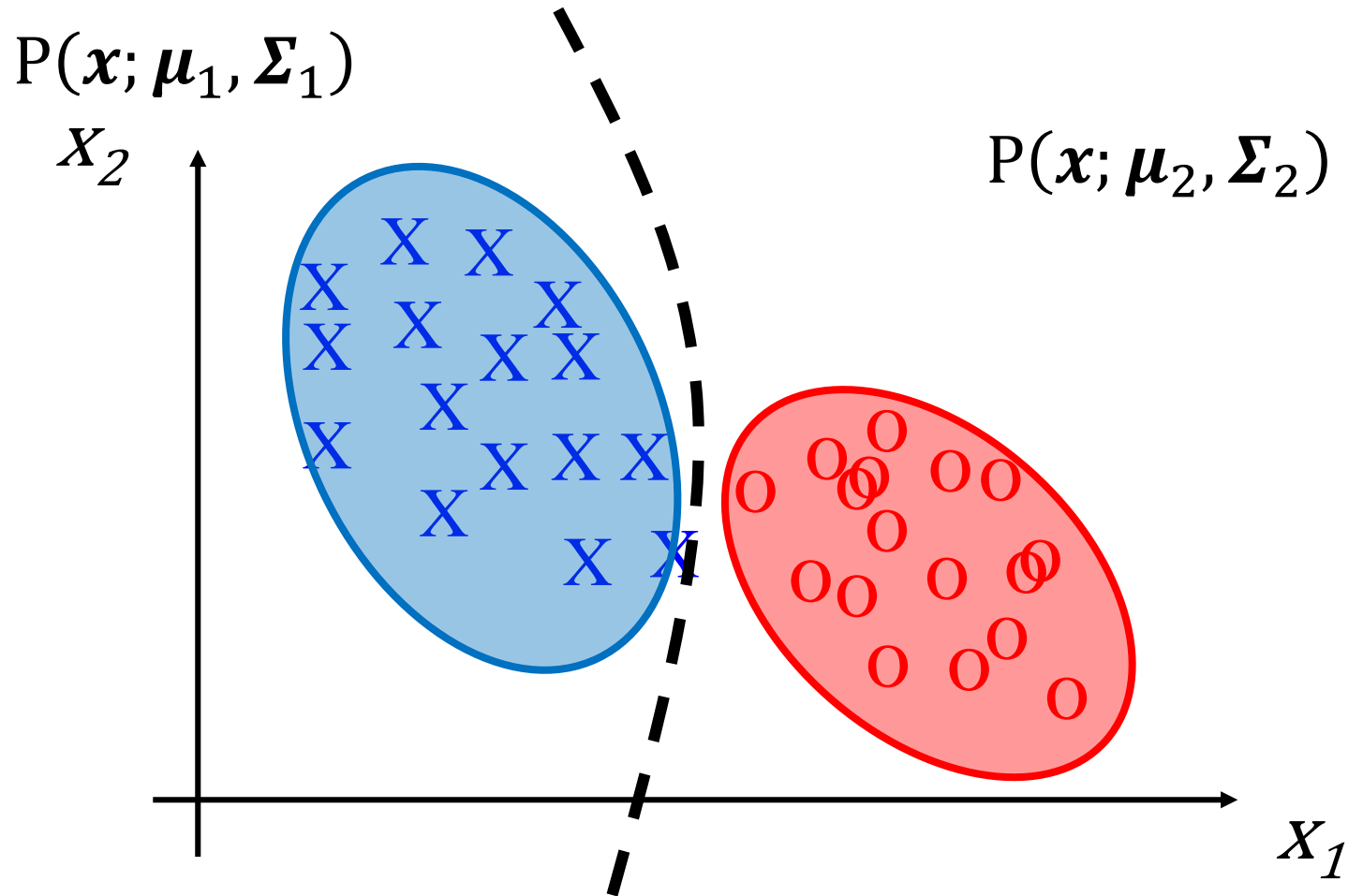
$$\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 9 \end{bmatrix}$$



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



Approche gaussienne multivariée



Séparatrice = Forme quadratique

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

Approche Bayésienne Naïve

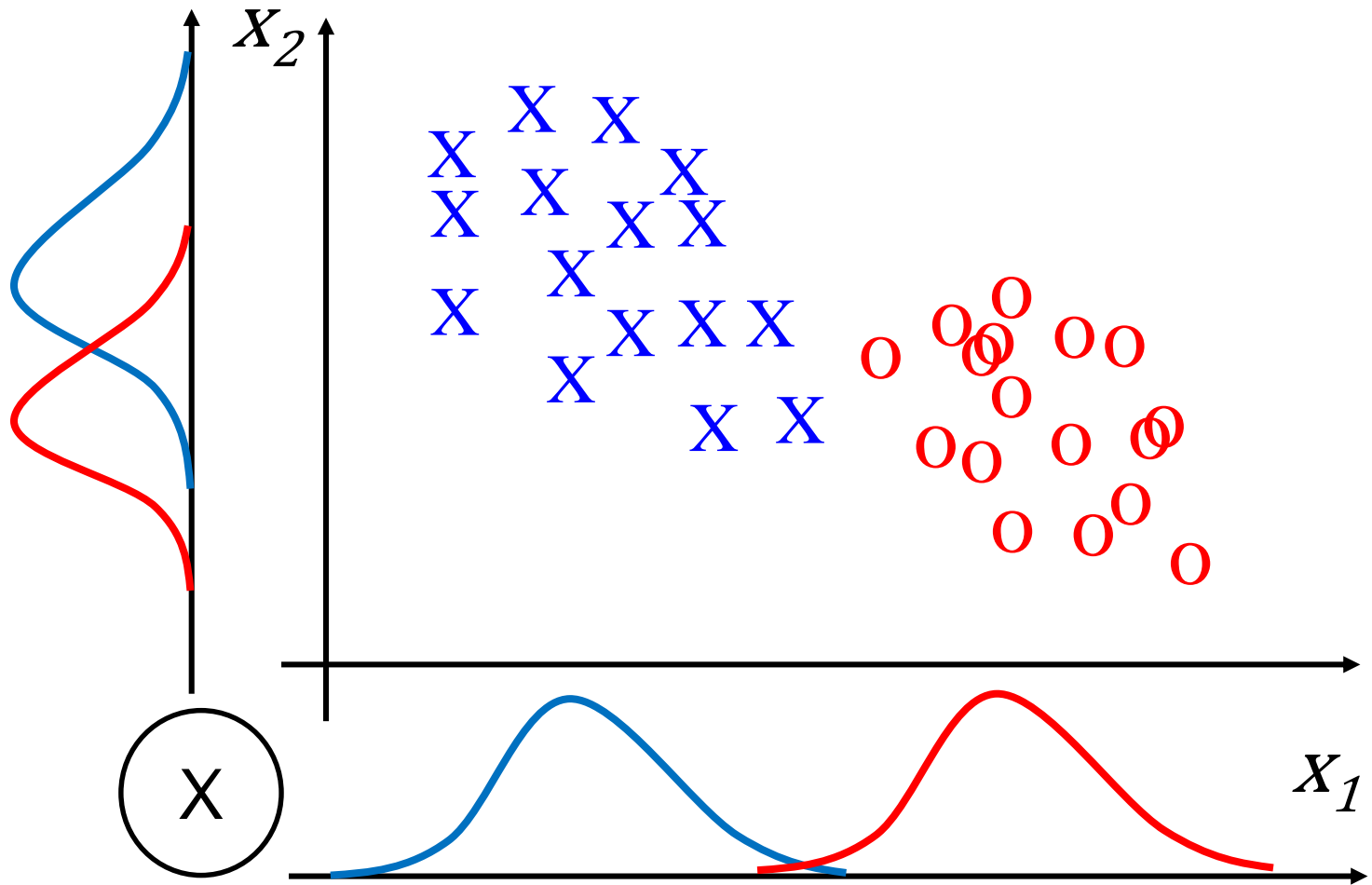
- Calcul de la loi conditionnelle: hypothèse d'indépendance.

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

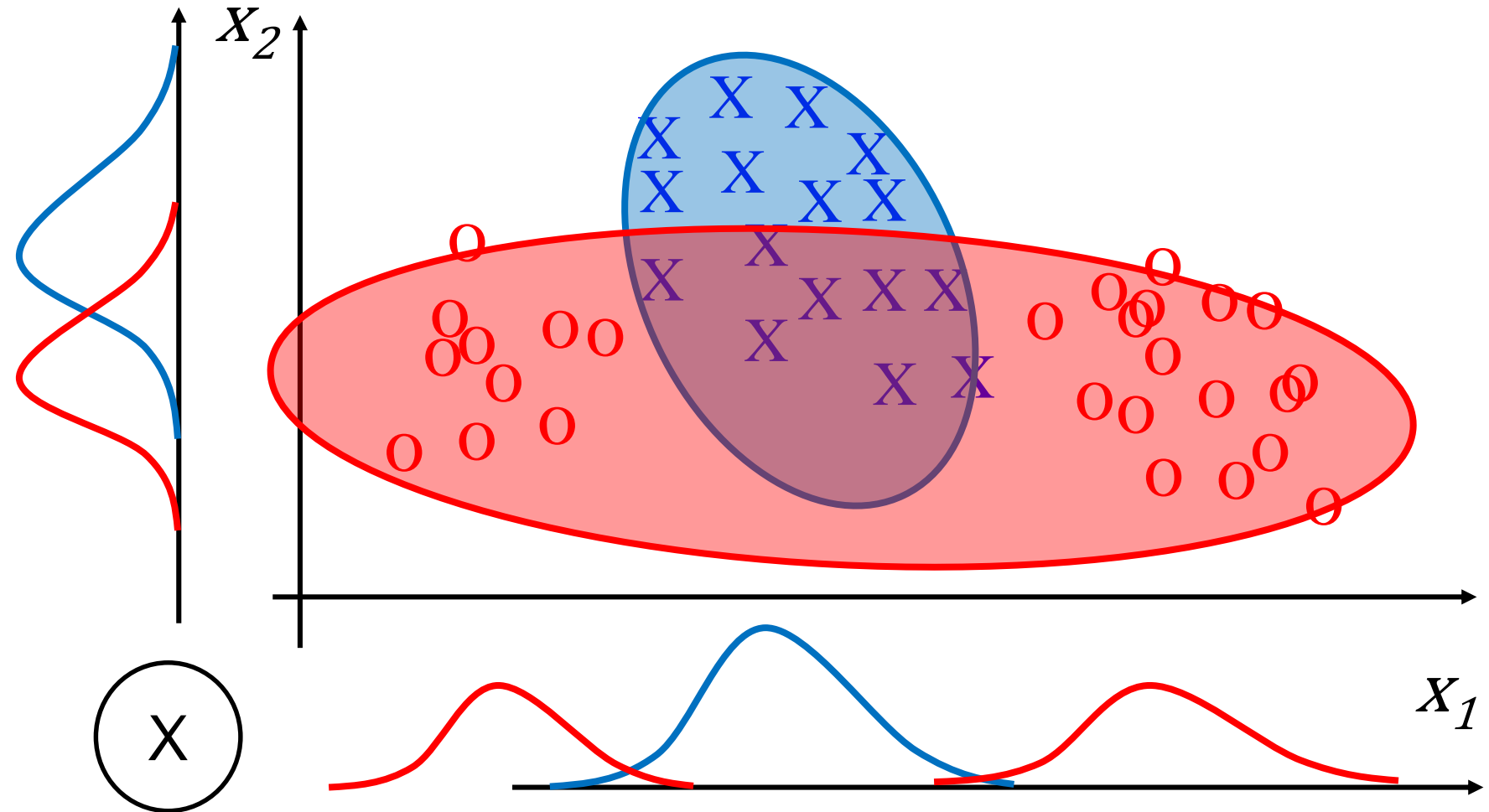
- On calcule la vraisemblance globale dimension par dimension
 - ➔ Problème 1D, modèles plus faciles à estimer (gaussien, binomial, histogrammes, mélange de gaussiennes...)
 - ➔ Permet de traiter des problèmes de plus grande dimension
- En pratique, on calcule plutôt la log-vraisemblance pour des questions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

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

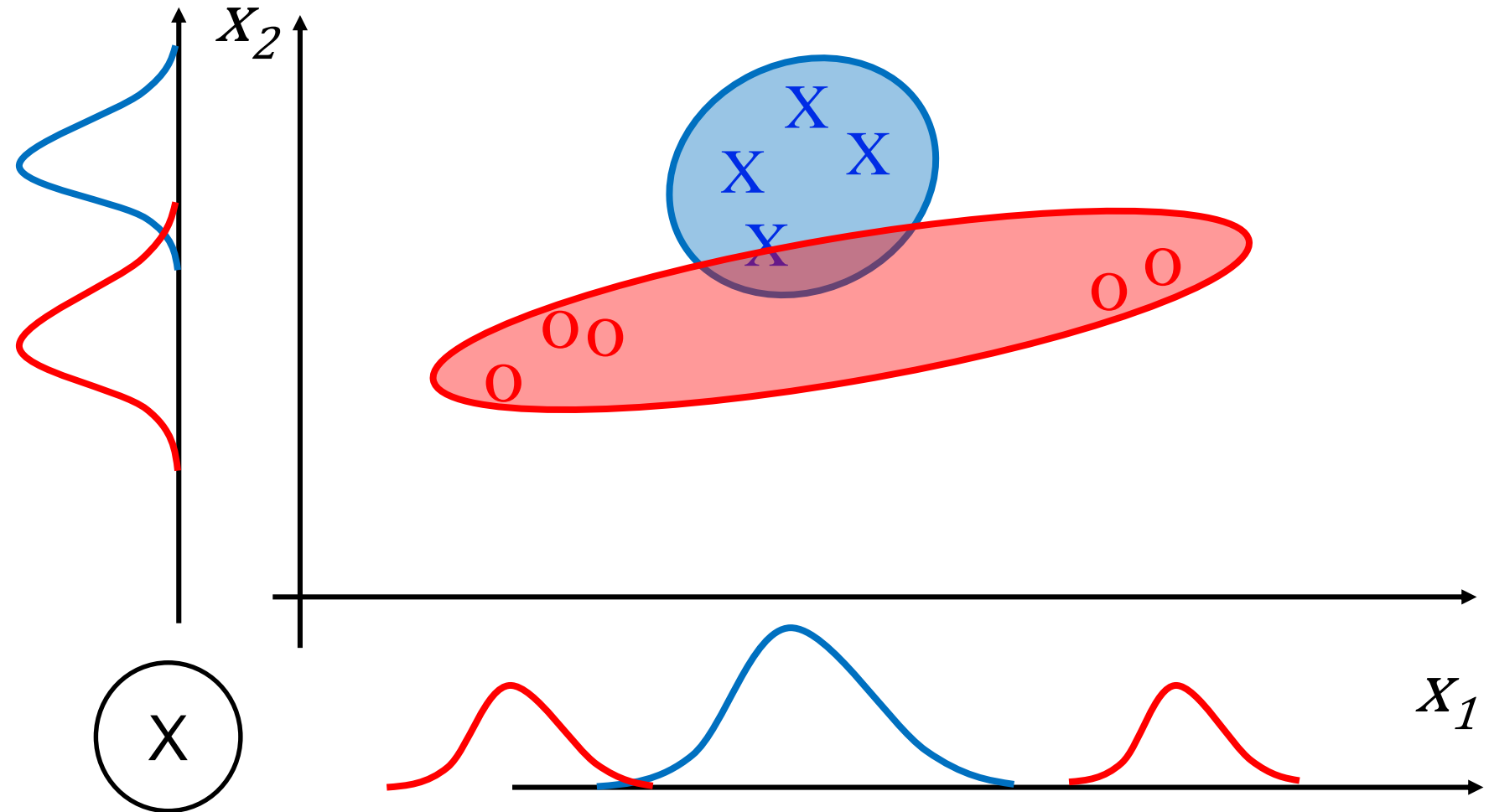
Approche bayésienne naïve



Approche bayésienne naïve vs. multivariée



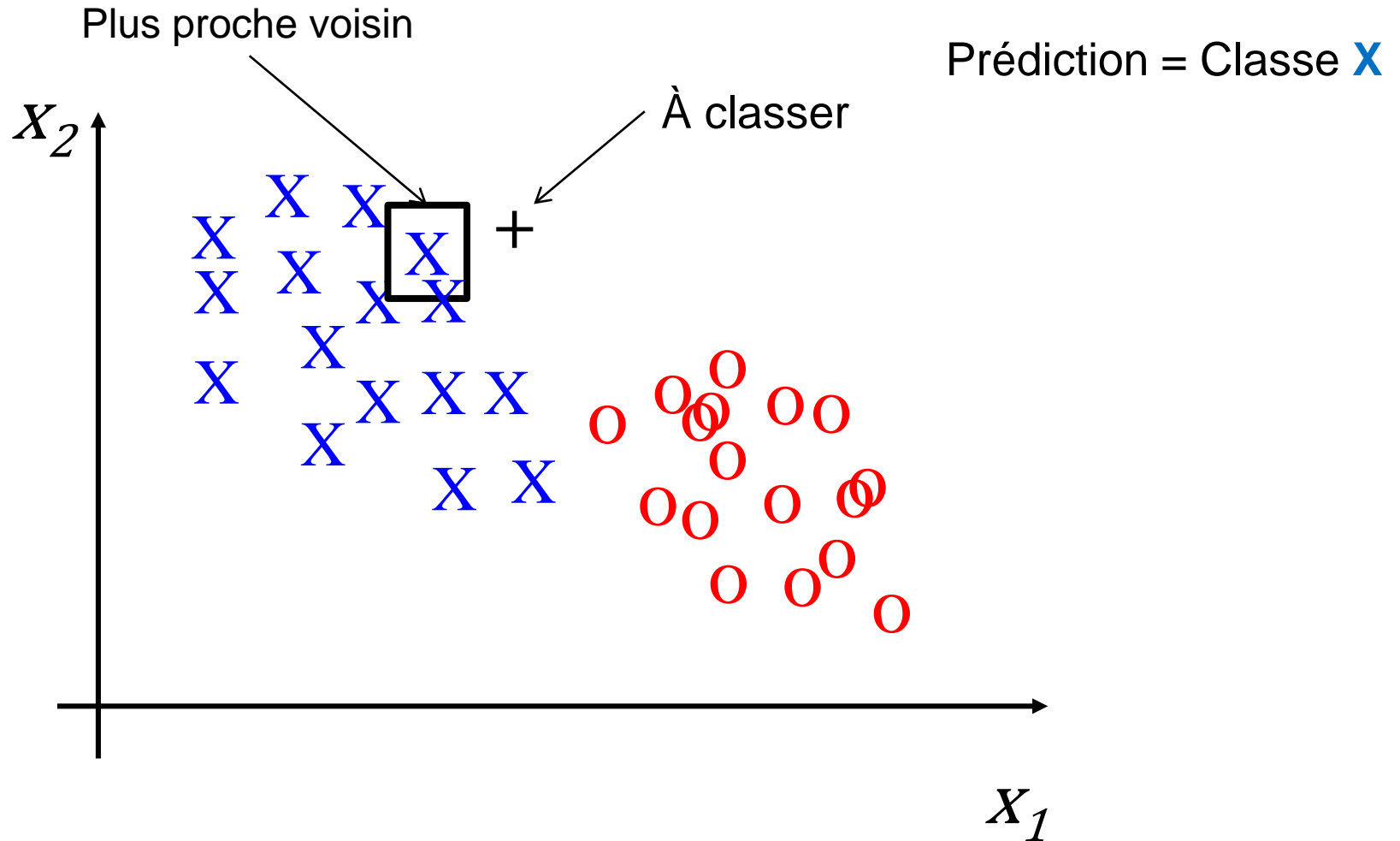
Approche bayésienne naïve vs. multivariée



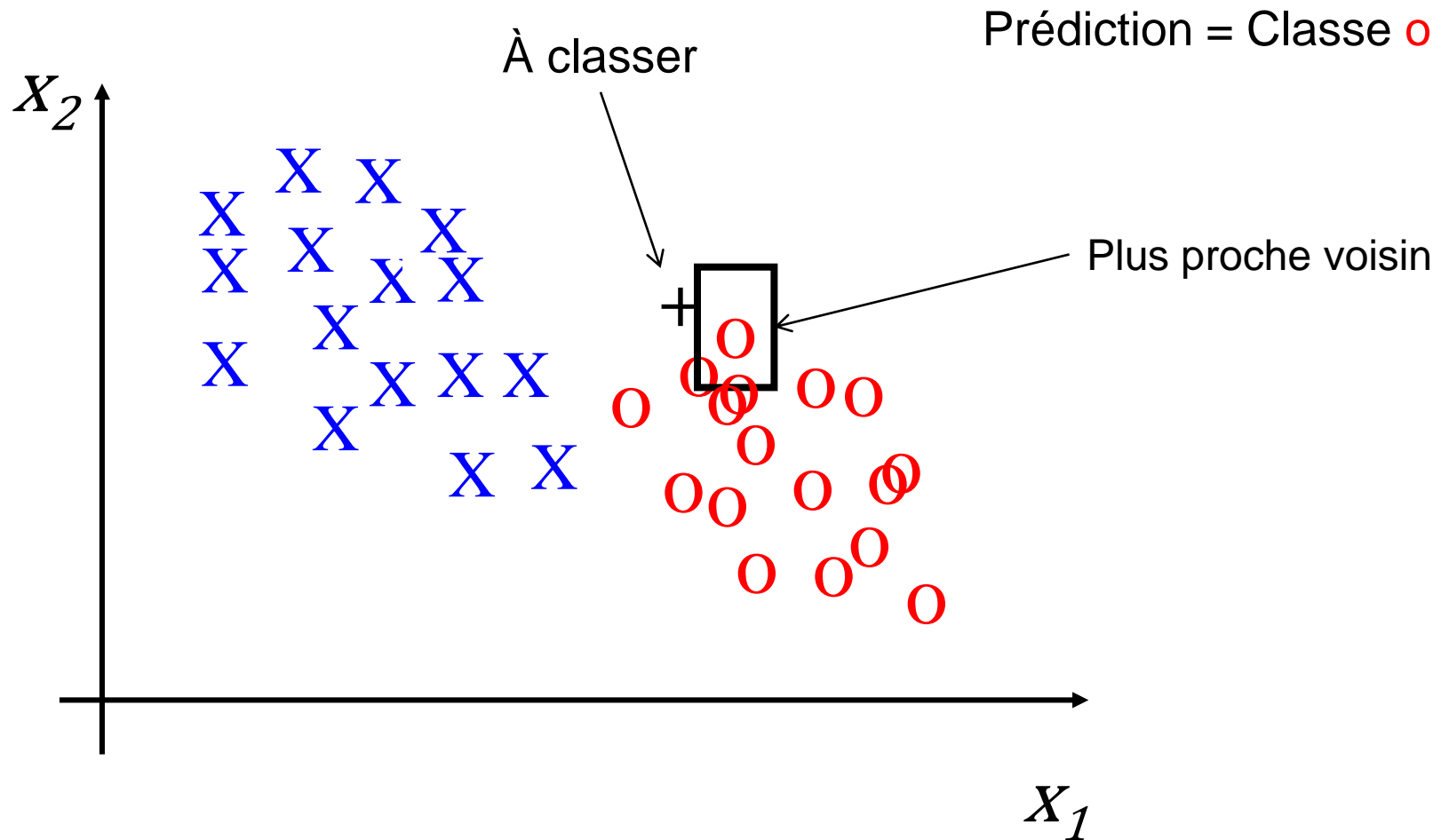
Approche bayésienne: résumé

- 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)

Classification ppv



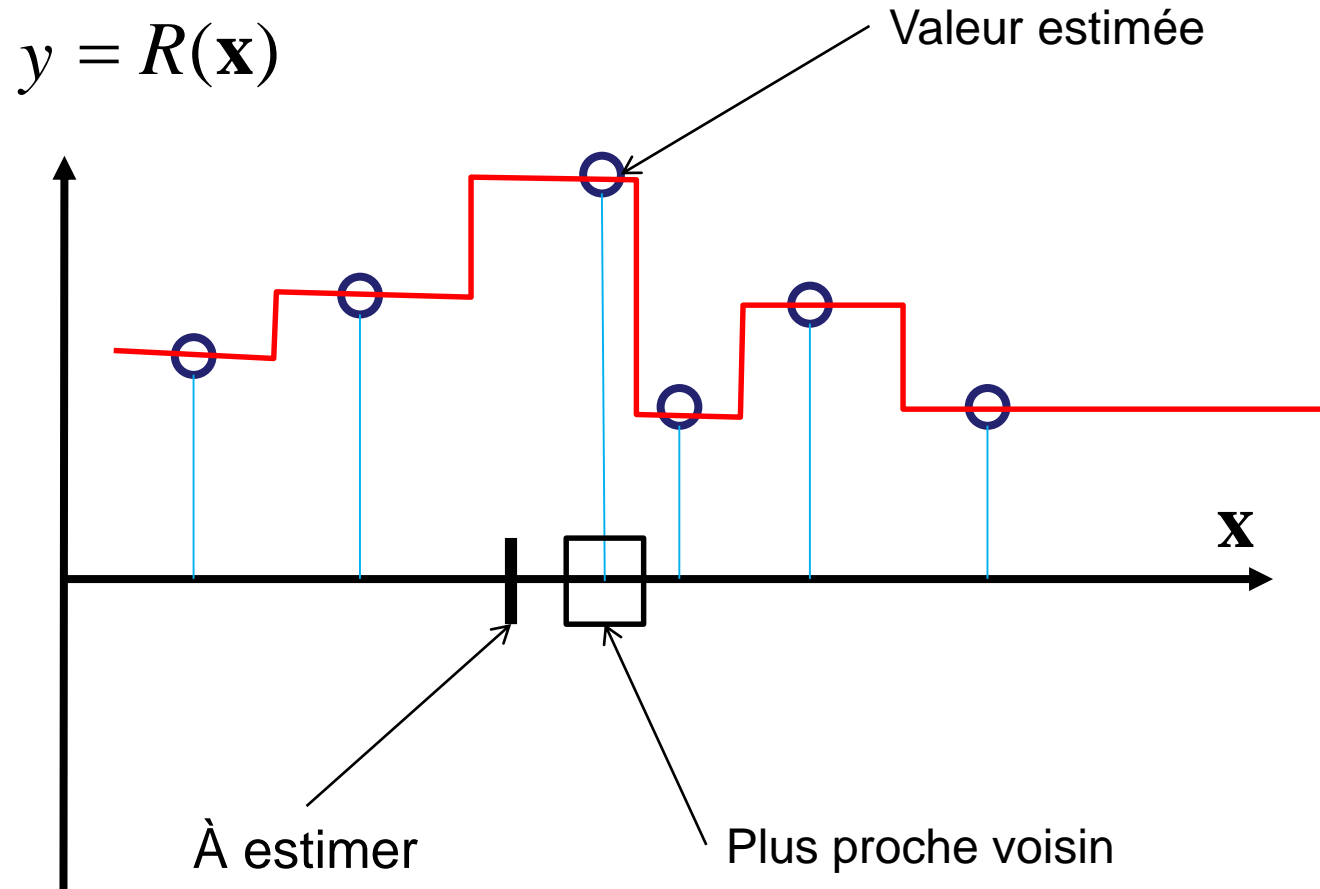
Classification ppv



Plus proche(s) voisin(s)

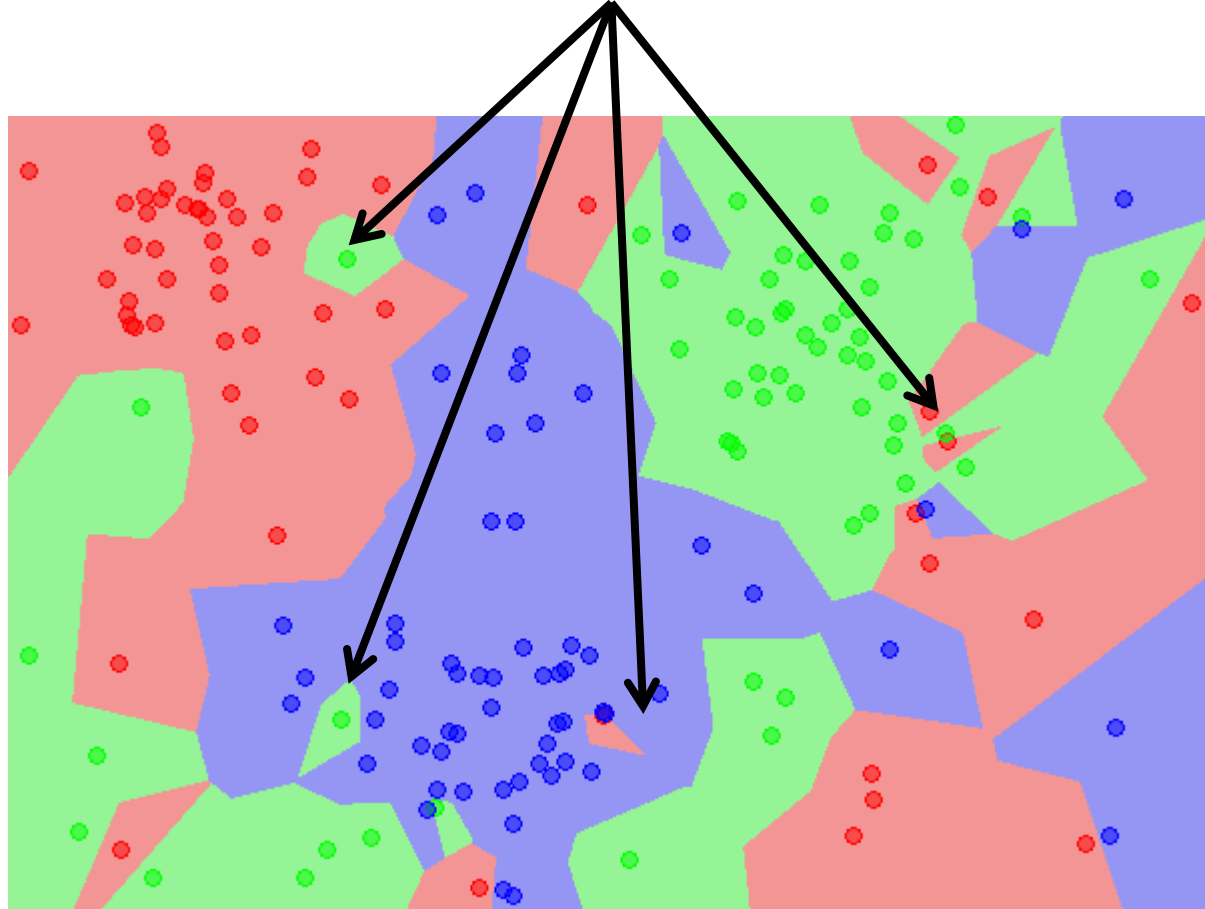
- Principe:
 - Deux échantillons **proches** dans l'espace de représentation ont les mêmes prédictions
 - Pour prédire, il suffit de trouver l'exemple annoté **le plus** proche, et d'associer son annotation (étiquette, valeur...)
- Que veut dire « proche » ?
 - Nécessite la définition d'une métrique ou mesure de similarité $d(x, x')$
 - 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 $\mathcal{L} = \{(x_1, y_1), (x_2, y_2), \dots (x_N, y_N)\}$
 - Recherche de l'échantillon le plus proche: $i^* = \arg \min_i d(x, x_i)$
 - Attribue comme prédiction l'annotation du plus proche: $y^* = y_{i^*}$

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

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

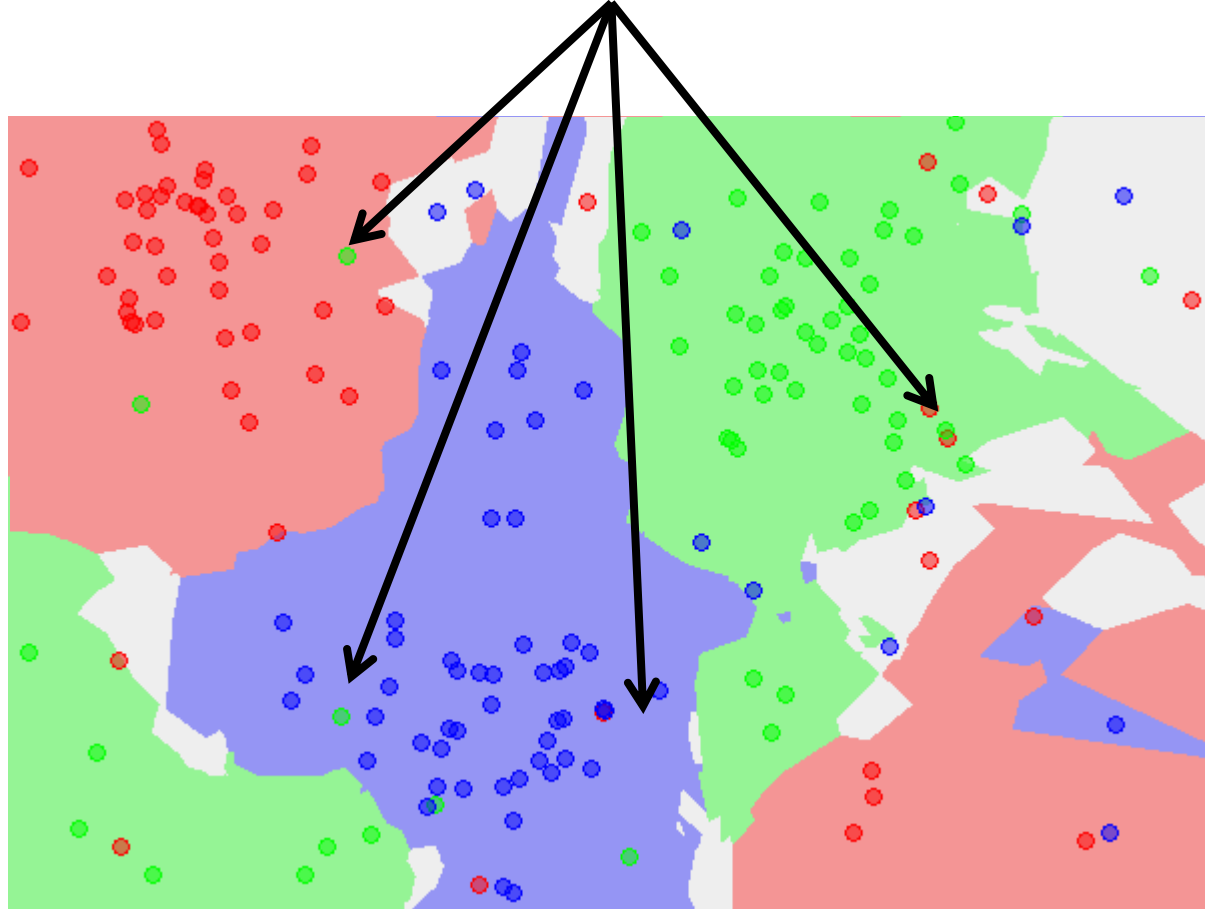
Où δ est la fonction de Kronecker (elle vaut 1 si égal, 0 sinon)

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

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

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 théorique. »

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 $[0,1]^d$, que:

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

- Ce n'est plus vrai si les distributions ont une structure...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 → apprentissage non supervisé

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

« 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

A retenir

- « Programmer à partir des données »
 - Deux phases: apprentissage et prédiction
 - Plusieurs variétés de prédicteurs et d'apprentissage
- Démarche générique:
 - Constitution d'une base d'apprentissage
 - Analyse préliminaire des données + préparation
 - Conception du modèle
 - Optimisation
 - Evaluation
- Objectif principal: minimiser l'erreur de généralisation
 - Train vs. Test
- Deux approches élémentaires:
 - Modélisation bayésienne
 - Plus proches voisins

Références

- R.O. Duda and P.E. Hart, Pattern classification and scene analysis, John Wiley & Sons, New York, 1973.
- P.A. Devijver and J. Kittler, Pattern Recognition, a Statistical Approach, Prentice Hall, Englewood Cliffs, 1982)
- K. Fukunaga, Introduction to Statistical Pattern Recognition (Second Edition), Academic Press, New York, 1990.
- 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, (<https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf>).
- Jerome H. Friedman, Robert Tibshirani et Trevor Hastie, The Elements of Statistical Learning: Data Mining, Inference, and Prediction (<https://web.stanford.edu/~hastie/ElemStatLearn/>).
- 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
 - CV on line: <https://computervisiononline.com/datasets>
 - Kitti: <http://www.cvlibs.net/datasets/kitti/>
 - Waymo: <https://waymo.com/open>

Journaux

- Journal of Machine Learning Research 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)
- International Conference on Learning Representations (ICLR)
- Uncertainty in Artificial Intelligence (UAI)
- International Joint Conference on Artificial Intelligence (IJCAI)
- International Conference on Neural Networks (ICNN)
- 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 MOOC (Français et Anglais)
- Des tutoriaux associés aux conférences (orientés recherche)
- Des cours en français:
 - <https://gricad-gitlab.univ-grenoble-alpes.fr/talks/fidle>
 - https://www.college-de-france.fr/site/stephane-mallat/_course.htm
- Des « cheat sheets »
 - <https://stanford.edu/~shervine/teaching/>

Logiciels

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



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 « .ipynb »)
- Ressources de calcul distantes (GPU)
- C'est proposé par Google
- <https://colab.research.google.com/>

Etales

- Se créer un gmail (ou utiliser le votre)
- Télécharger le TD (fichiers données et notebook) sur le gdrive du compte
- Se connecter à Colab
- Ouvrir le Notebook du TD (td1_knn_bayesien.ipynb)
- Monter le drive dans Colab (première étape du TD)
- Modifiez directement le notebook.

Utilisation des « Notebook Python »

td2_classification_supervisee (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

plt.show()

après chaque fonction de visualisation.

Activité 1.1 : Mon premier classifieur.

Repérer les paramètres utiles des fonctions et la manière de les utiliser. Lancer une première chaîne de calcul pour apprendre un classifieur linéaire pour le jeu de données 0 et des valeurs par défaut. Visualiser les résultats et calculer l'erreur sur les jeux d'apprentissage et de test avec la fonction `score`. Utiliser la fonction `predict` et comparer les sorties produites avec les vraies valeurs. Vérifier avec la fonction `score` que les valeurs d'erreur sont les mêmes. Recommencer la séquence d'apprentissage avec les autres distributions de données (1 à 3).

```
[1]: # Librairies utiles standard
import numpy as np
import matplotlib.pyplot as plt

# Pour visualiser et récupérer les données
import iogs_td_util as td

# L'algorithme SVM dans la bibliothèque scikit-learn
from sklearn import svm

import random
```

```
In [2]: # Classifieur
svc = svm.SVC(kernel='linear', max_iter=-1)

# Premier jeu de données
trainX, trainY, testX, testY=td.generate_data(0)
```

Texte

Exécution des
cellules

Code

Pour se mettre à jour sur Python & Numpy

- Intro à Numpy et Matplotlib
 - <https://sebastianraschka.com/blog/2020/numpy-intro.html>
 - <https://cs231n.github.io/python-numpy-tutorial/>
- « Cheat sheets »
 - <https://www.datacamp.com/community/data-science-cheatsheets>