

Apprentissage

Cours 2: Introduction à l'apprentissage

Alexis Lechervy

Apprendre ? Qu'est ce que c'est ?

Apprendre c'est s'adapter.

- Apprendre c'est s'adapter à des situations **nouvelles et inconnues** en prenant en compte l'expérience passée.
- Apprendre est une **propriété humaine essentielle**.
- Apprendre signifie "s'améliorer afin d'être meilleur" (selon un critère donné) lorsqu'une situation similaire se présente.

Apprendre ce n'est pas du "par-coeur".

- Ne pas confondre l'apprentissage et la **récitation par-coeur**.
- N'importe quel ordinateur peut réciter "par-coeur", la difficulté est de généraliser à des situations nouvelles et inconnues.

Mais pourquoi apprendre à un ordinateur ?

- Pouvoir gérer une quantité de données très importante de manière automatique.
- Pouvoir effectuer une action dans un contexte non prévu préalablement sans l'intervention d'un humain.
- Pouvoir prévoir des comportements ou des évolutions pour aider à la prise de décision.

L'Apprentissage en informatique, quand faut-il l'utiliser ?

Quand l'utiliser ?

- L'expertise humaine n'est pas possible (ex : navigation sur Mars)
- La quantité d'information est trop grande pour être traitée par un humain (ex : recherche d'une personne dans une base d'image)
- Besoin de traitement temps réel (ex : mise au point d'un appareil photo sur un visage)
- Automatisation d'une chaîne de traitement (ex : détection d'anomalie sur une chaîne de montage)
- Les êtres humains ne savent pas expliquer leurs expertises (ex : reconnaissance de la parole)
- Trouver une solution optimale à un problème (ex : trouver le meilleur modèle)

Quand ne pas l'utiliser ?

- Pour réaliser une application simple dont le comportement est facilement définissable,
- Pour réaliser une tâche impossible,
- Lorsque les données sont mal définies notamment si elles contiennent des biais dûs à des préjugés ou des stéréotypes.
- Pour confirmer ou invalider l'avis des experts sans discernement.

⇒ l'apprentissage se fait à partir de données, l'algorithme apprendra et reproduira les défauts qu'elles peuvent contenir.

Définitions

Définition intuitive

L'apprentissage automatique consiste en la conception et le développement d'algorithmes qui permettent aux ordinateurs (machines) d'améliorer leurs performances au fil du temps sur une base de données.

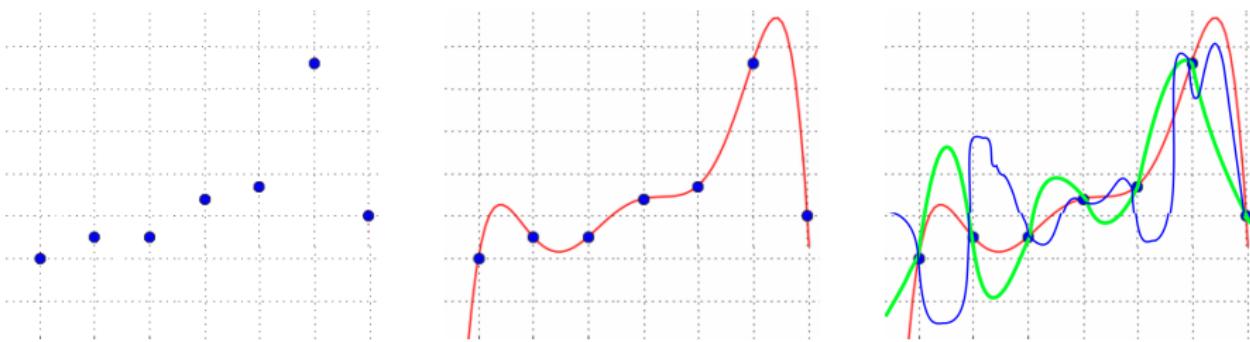
Définition formelle (Mitchell, 1998)

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

(Machine Learning de Tom M. Mitchell, Chapitre 1, page 2).

Pourquoi est ce difficile ?

- Etant donné un **nombre fini de données d'apprentissage**, il faut en déduire une **relation pour un domaine infini**.
- Problème : il y a un **nombre infini** de telles **relations**.
- Comment définir cette relation ?
- Quelle relation est la plus appropriée ?



Alors comment faire ?

Principe du rasoir d'Occam

William of Occam (un Moine du 14^{ième} siècle) propose d'appliquer le **principe de parcimonie** :

On ne devrait pas augmenter, au-delà de ce qui est nécessaire, le nombre d'entités requises pour expliquer quoi que ce soit.

En d'autre mot : Lorsque de nombreuses solutions sont disponibles pour un problème donné, nous devons sélectionner **la plus simple**.

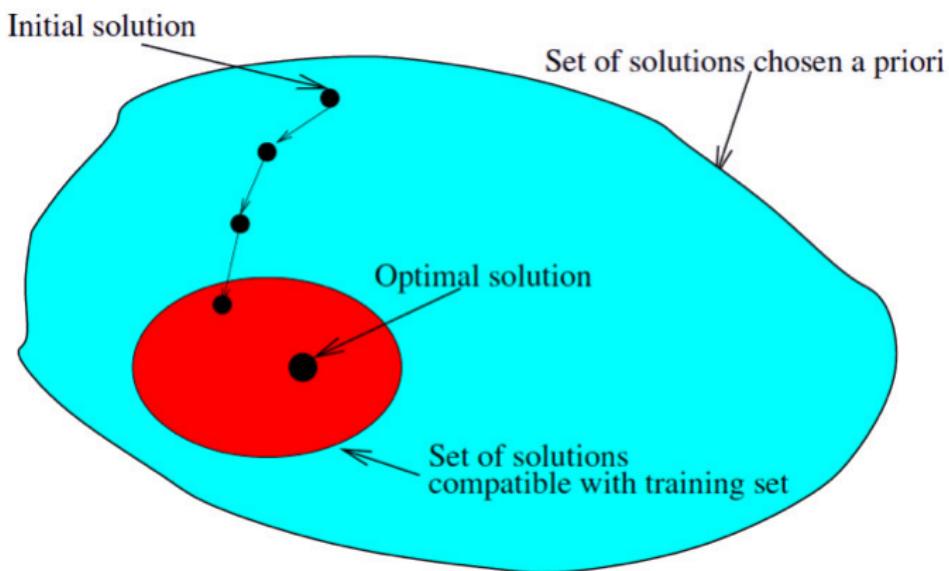
⇒ Mais que signifie la plus simple dans notre cas ?

Comment définir la simplicité d'une solution ?

- Nous devons utiliser des connaissances apriori du problème à résoudre pour définir ce qui est une solution simple.
- Exemples d'apriori : solution lisse, solution à peu de coefficients...

L'apprentissage : une recherche d'une solution parmi un ensemble de possibilité.

L'apprentissage consiste à choisir une solution parmi un ensemble de solution défini selon un a priori. On recherche la solution optimale. C'est-à-dire la solution "la plus simple" qui répond à notre problème.



Un peu d'histoire.

Histoire

- 1960 : IA (Intelligence Artificielle) symbolique, les ordinateurs apprennent des règles à partir des données mais une analyse des statistiques sous-jacentes est rarement faite.
- Fin 1960, début 1970 : introduction de modèles de discrimination linéaires dont le fameux perceptron de Minsky et Papert, 1969.
- 1980 : Introduction des réseaux de neurones artificiels.
- 1990-2000 : ère de l'apprentissage statistique (méthodes à noyaux, SVM, arbres de décisions, boosting, modèles graphiques).
- 2010- : ère du BigData et du BigLearning.

Pourquoi l'apprentissage a mis tant de temps à se développer ?

- Des ordinateurs plus rapides avec une plus grande mémoire pour représenter des modèles sont désormais disponibles.
- Les méthodes d'analyse numériques sont disponibles sur l'ordinateur de bureau.
- De grandes quantités de données disponibles qui peuvent être exploitées (profils utilisateurs, flickr, réseaux sociaux...).
- De nombreux ensembles de données avec des jeux de données partiels / incomplets.

Un domaine d'avenir.

Avec l'accroissement toujours plus important des données, le traitement automatique est devenu indispensable.

Facebook adopte le Deep Learning pour mieux comprendre ses utilisateurs

Par Ruolin Yang | 01 octobre 2013 | [Laisser un commentaire](#)

 **Mots-clés :** Innovation, Facebook, Google, Microsoft, YouTube, Amériques, analyse de données, ciblage publicitaire, Deep Learning, MIT, Stanford, Asie, EMEA



[Tweeter](#) 75 [g+](#) 15 [Recommander](#) 28 [Share](#) 15

En explorant cette branche de l'intelligence artificielle, le réseau social entend donner sens à des millions de posts créés chaque jour.

Machine Learning @amazon

YAHOO!
LABS

[Home](#) [Areas](#) [News](#)

Machine Learning

Research
at Google

[Home](#) [Research Areas & Publications](#) [People](#) [Research Programs](#) [Work at Google](#)

Artificial Intelligence and Machine Learning

Sommaire

1 Les familles d'algorithmes d'apprentissages

- Différents types d'apprentissage
- Problèmes typiques

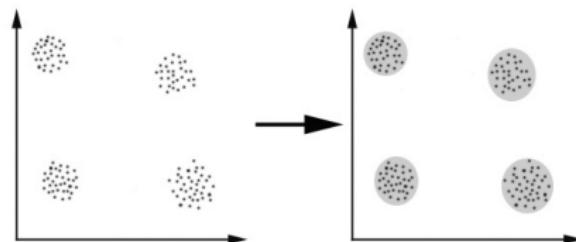
2 Les étapes d'un algorithme d'apprentissage

L'apprentissage non supervisé (Clustering)

Principes

- Diviser les données en plusieurs groupes séparés,
- Extraire une connaissance organisée sans intervention humaine,
- les données les plus similaires sont associées au sein d'un groupe homogène
- les données considérées comme différentes se retrouvent dans d'autres groupes distincts
- Pas d'apriori sur les données
- Il y a une seule entrée, les données collectées

Exemple de clustering



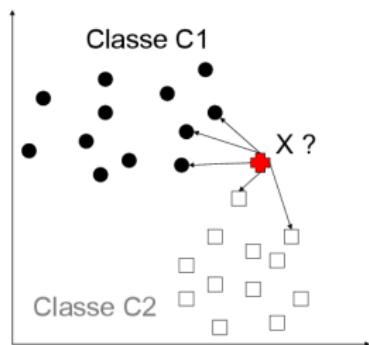
L'apprentissage supervisé

Principes

- On détermine automatiquement une règle à partir de données d'apprentissage annotées par un expert,
- Un expert a défini un ensemble de couples (donnée,label),
- Il y a un a priori sur les données,
- Les données entrées sont des couples (données collectées, observations).

Un exemple simple : les k-plus-proches voisins (kPPV ou kNN)

- La classe d'un exemple est un vote majoritaire des k plus proches voisins.
- On regarde la classe des voisins les plus proches de l'exemple à tester en se limitant au k plus proche.
- On attribue à l'exemple la classe la plus présente parmi ces voisins



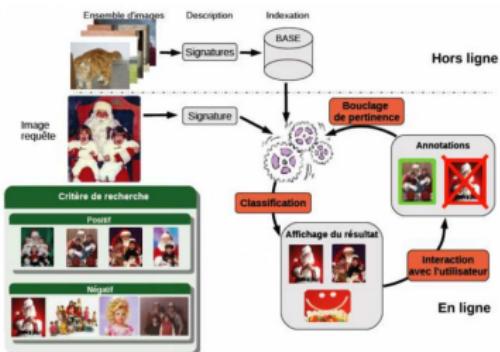
L'apprentissage semi-supervisé

Principes

- On dispose de quelques exemples labellisés.
- Les autres ne le sont pas.
- Permet de travailler avec moins de labels.

Exemples d'apprentissage non supervisé

- L'apprentissage interactif.
- L'apprentissage sur des ensembles de données trop important.

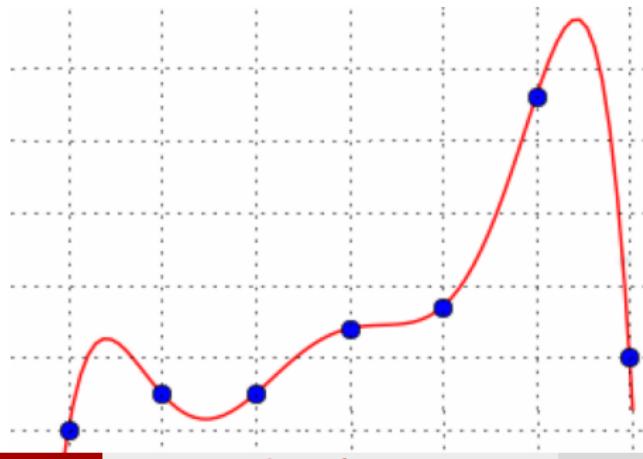


La régression

Principes

- Trouver la relation entre une variable et une ou plusieurs autres variables.
- Trouver la fonction qui minimise un critère d'erreur entre les valeurs de la fonction aux points d'apprentissage et les observations.
- Recherche d'une fonction de type $y = f(x, a, b) = (ax + b)$.

Exemple

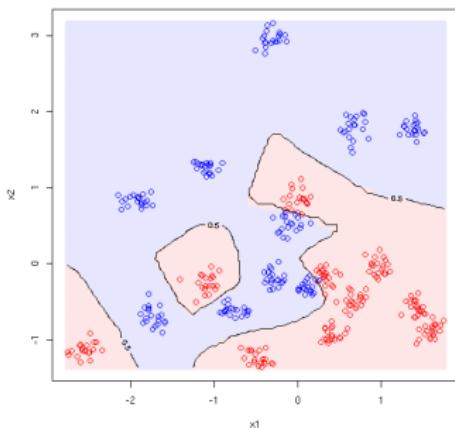


La classification

Principes

- Attribuer une classe à chaque objet.
- Utilise des données statistiques sur les objets pour choisir la classe.
- Recherche d'une fonction $y = f(x, a, b) = \text{sign}(ax + b)$

Exemple



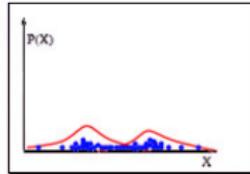
Estimation de densité

Principes

- Trouver les paramètres d'une loi de probabilité permettant d'estimer au mieux une distribution de points.
- Recherche d'une fonction

$$p(z) = f(z; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{|z|}{2}} \sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(z - \mu)^\top \Sigma^{-1}(z - \mu)\right).$$

Exemples

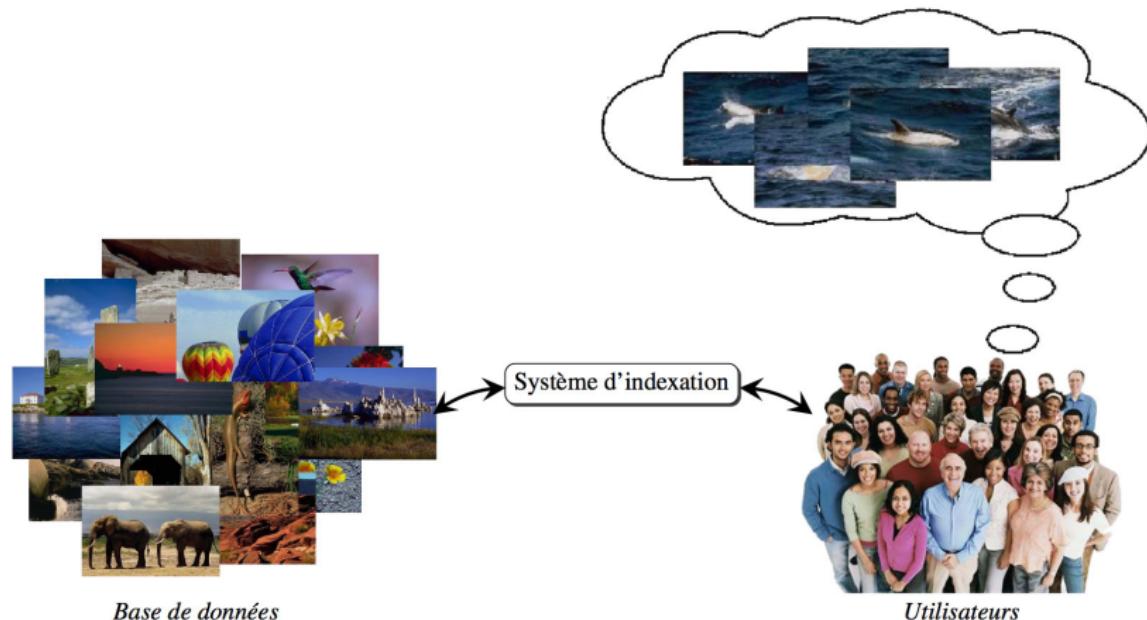


Sommaire

1 Les familles d'algorithmes d'apprentissages

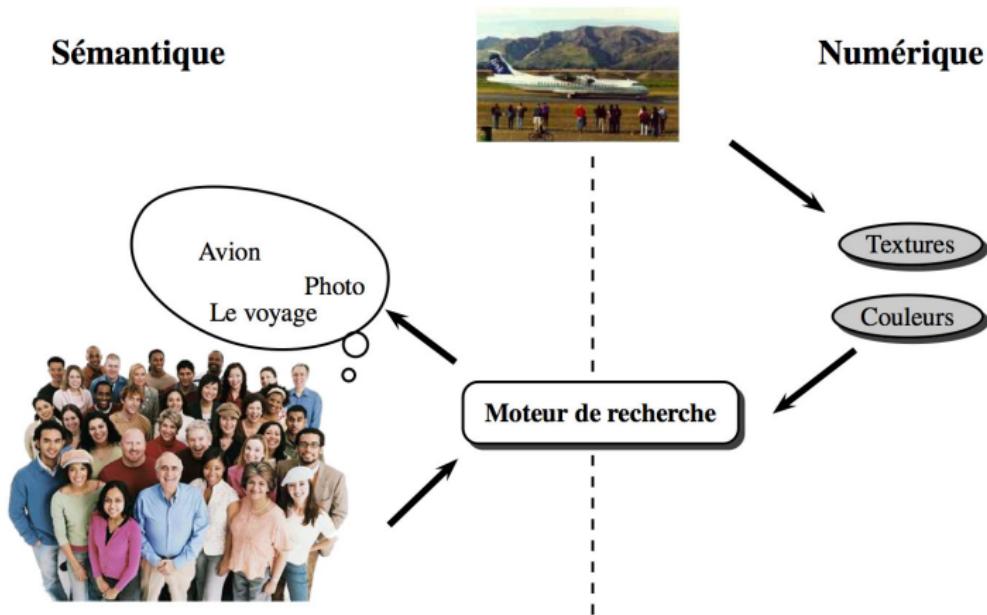
2 Les étapes d'un algorithme d'apprentissage

Définir le problème d'apprentissage et construire la base de donnée



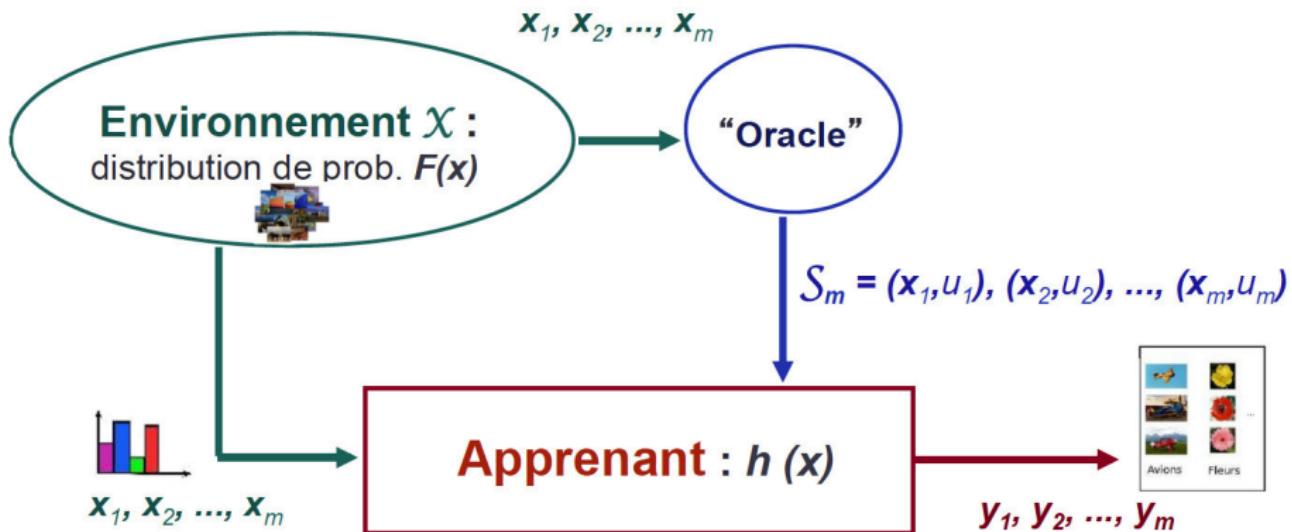
Comment aider les utilisateurs à traiter des bases de données toujours plus vastes ?

Définir la représentation des données



Défi : Combler le fossé sémantique/numérique entre les concepts sémantiques de l'utilisateur et la représentation numérique des images

Le déroulement



Induction : Proposer des **lois générales** à partir de l'observation de **cas particuliers**.

Définition formelle du problème

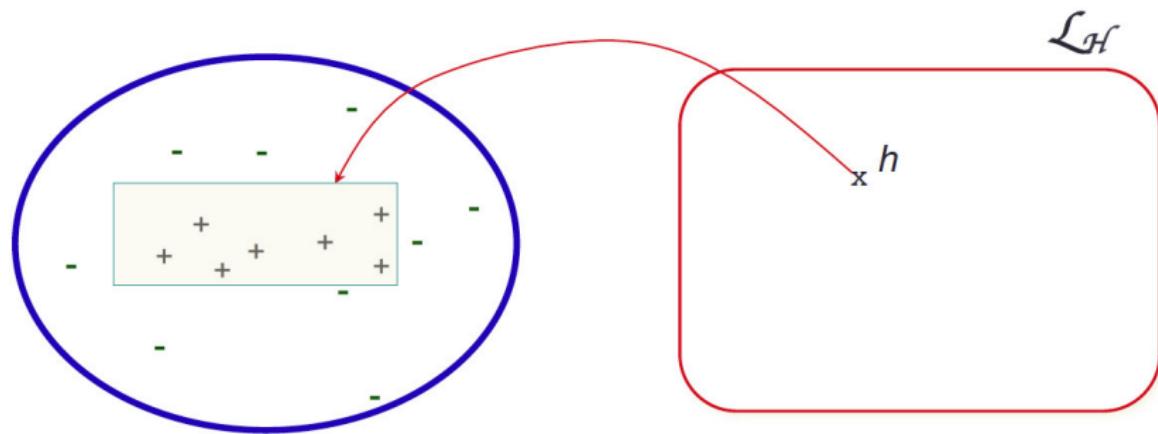
Hypothèses

- Les données caractérisent une dépendance probabiliste \mathcal{P} entre l'espace \mathcal{X} des descriptions et l'espace \mathcal{Y} des étiquettes.
- $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$: variables aléatoires sur $(\Omega, \mathcal{B}, \mathcal{P})$, où \mathcal{P} est inconnue.
- $\mathcal{S} = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\} \in (\mathcal{X} \times \mathcal{Y})^N$.
- Échantillon d'apprentissage :
 - Les observations sont i.i.d. (indépendantes et identiquement distribuées) suivant \mathcal{P} .
 - \mathcal{H} : famille (éventuellement infinie) de fonctions h définies sur \mathcal{X} .

Objectif

Prédire l'étiquette y , à l'aide d'une fonction h , connaissant l'observation x .

Apprendre = un jeu entre espaces

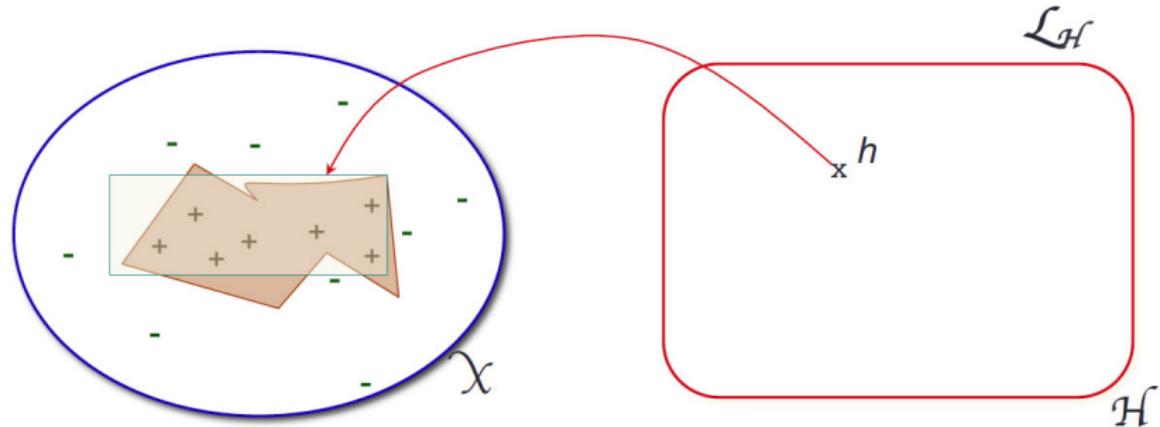


Espace des exemples : \mathcal{X}

Espace des hypothèses : \mathcal{H}

➡ Comment choisir l'espace des hypothèses (i.e. le langage $\mathcal{L}_{\mathcal{H}}$) ?

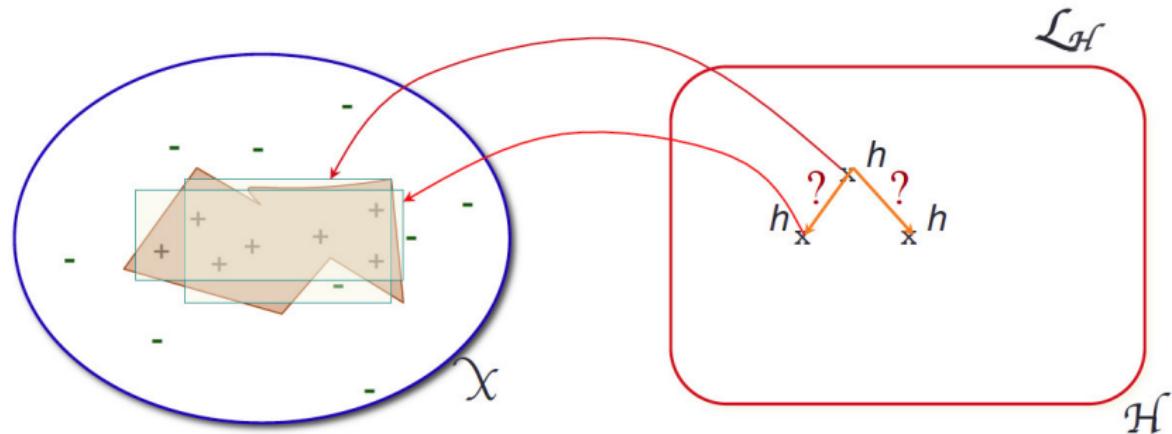
Le critère inductif



→ Quel *critère inductif* ?

→ Qu'est-ce qu'une hypothèse optimale étant donné l'échantillon d'apprentissage ?

L'exploration de \mathcal{H}



➡ Quelle méthode d'exploration de \mathcal{H} ?

Trois interrogations fondamentales

1- Quel critère inductif ?

Quelle hypothèse devrait-on choisir étant donné l'échantillon d'apprentissage ?

2- Quel espace d'hypothèses ?

Quel espace d'hypothèses est approprié ?

3- Comment explorer l'espace des hypothèses ?

Résolution d'un problème d'optimisation

Critère de performance

Objectif

Trouver une hypothèse $h \in \mathcal{H}$ minimisant le risque réel (espérance de risque, erreur en généralisation).

$$R(h) = \int_{X \times Y} l(h(x), u) dP(x, y)$$

**Fonction de perte
(ou loss)**

Étiquette
prédite

Étiquette vraie
(ou désirée)

Loi de probabilité
jointe sur $X \times Y$

Problème

$P(x, y)$ est inconnue et l'on ne peut accéder à $l(h(x), u)$ pour tout x (on dispose uniquement d'une partie de l'espace des exemples).

Exemples de fonctions de perte

Régression

$$l(h(x_i), u_i) = [h(x_i) - u_i]^2$$

Classification

$$l(h(x_i), u_i) = \begin{cases} 0 & \text{si } u_i = h(x_i) \\ 1 & \text{si } u_i \neq h(x_i) \end{cases}$$

Estimation de densité

$$l(h(x_i)) = -\log(h(x_i))$$

Critères inductifs

- ➊ Principe de minimisation du risque empirique (ERM).
- ➋ Principe du maximum de vraisemblance (approche bayésienne).
- ➌ Principe de compression maximale.

Le principe inductif ERM

Risque réel

$$R(h) = \int_{\mathcal{X} \times \mathcal{Y}} I(h(x, u)) \, dP(x, y)$$

- On ne connaît pas le risque réel, en particulier pas la loi de probabilité $P(\mathcal{X}, \mathcal{Y})$.

Risque empirique

Le principe ERM (minimisation du risque empirique) prescrit de chercher l'hypothèse $h \in \mathcal{H}$ minimisant le risque empirique

$$R_{emp}(h) = \sum_{i=1}^N I(h(x_i, u_i)).$$

Approche bayésienne

Hypothèse

On suppose qu'il existe une distribution de probabilités a priori sur l'espace \mathcal{H} :
 $P_{\mathcal{H}}(h)$

Principe du Maximum A Posteriori (MAP)

On cherche l'hypothèse h la plus probable après observation des données \mathcal{S} .

$$h^* = \operatorname{argmax}_{h \in \mathcal{H}} P(h|\mathcal{S}) = \operatorname{argmax}_{h \in \mathcal{H}} \frac{P(\mathcal{S}|h)P(h)}{P(\mathcal{S})} = \operatorname{argmax}_{h \in \mathcal{H}} P(\mathcal{S}|h)P(h)$$

Compression maximale

Hypothèse

On suppose qu'il existe :

- un coût associé à la **transmission d'un codage** (modèle des données) : $L(h)$
- un coût associé à la **transmission des données brutes** (E.D. h) : $L(x|h)$

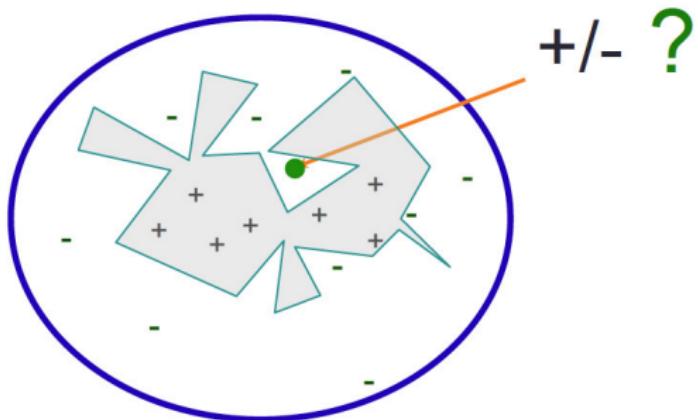
Principe de compression maximale

- S'inspire de la théorie du codage de l'information.
- Rasoir d'Occam.
- On cherche le modèle (ou hypothèse) permettant la transmission la plus économique de l'échantillon de données.

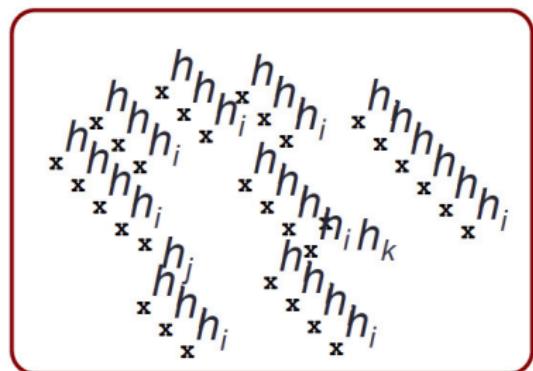
$$h^* = \operatorname{argmax}_{h \in \mathcal{H}} L(h) + L(x|h)$$

Choix de l'espace d'hypothèses

- Apprendre (pour prédire) est impossible ...
... sans limitation sur l' espace des hypothèses



Espace des exemples : \mathcal{X}



Espace des hypothèses : \mathcal{H}

Choix de l'espace d'hypothèses

Théorème de Vapnik-Chervonenkis

Soit $0 < \delta \leq 1$, on a avec une probabilité d'au moins $1 - \delta$ d'avoir

$$\forall h \in \mathcal{H} \quad R_{\text{réel}} \leq R_{\text{emp}} + 2 \sqrt{\frac{G^{\mathcal{H}}(2N) - \log(\frac{\delta}{4})}{N}}.$$

où $G^{\mathcal{H}}(2N)$ est la fonction de croissance (growth function). Elle représente la "richesse" de \mathcal{H} ou plus exactement elle mesure le nombre maximal de classificateurs qui peuvent être générés à partir d'un ensemble de données de taille donnée.

Pistes pour l'apprentissage

1. De quelles informations doit-on disposer ?

- Compromis entre taille de l'échantillon d'apprentissage nécessaire ET "richesse" de l'espace d'hypothèses.

2. Quel principe inductif ?

- Par le principe "naïf" : minimiser l'erreur sur l'échantillon d'apprentissage en pariant que l'erreur sera également minimisée sur les exemples non vus.
- Un nouveau principe : minimiser à la fois
 - L'erreur sur l'échantillon d'apprentissage.
 - Et une mesure de la richesse de \mathcal{H} .

Le principe inductif ERM

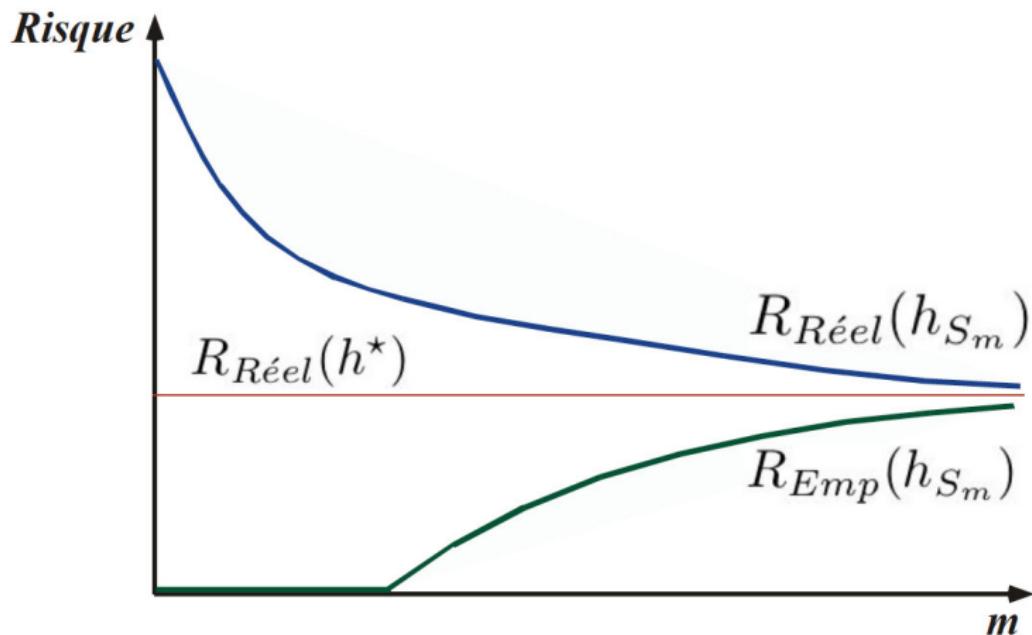
$$R(h) = \int_{\mathcal{X} \times \mathcal{Y}} I(h(x), u) dP(x, y).$$

- On ne connaît pas le risque réel, en particulier pas la loi de probabilité $P(\mathcal{X}, \mathcal{Y})$.
- Le principe ERM (minimisation du risque empirique) consiste à chercher l'hypothèse $h \in \mathcal{H}$ minimisant le risque empirique.

$$R_{Emp}(h) = \sum_{i=1}^N I(h(x_i), y_i).$$

Le principe ERM est-il pertinent ?

- h^* : hypothèse optimale dans \mathcal{H} suivant le risque réel.
- h_{S_m} : hypothèse optimale dans \mathcal{H} suivant le risque empirique mesuré sur l'échantillon S_m .

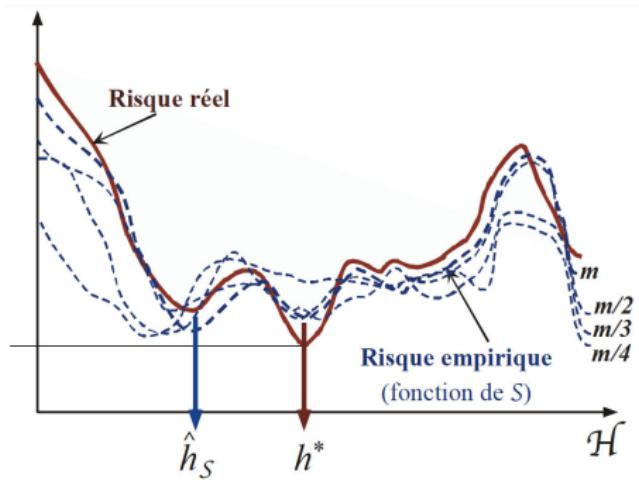


Analyse statistique du principe ERM

Étude de la corrélation entre

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}} R_{\text{réel}}(h), \quad h_S = \underset{h \in \mathcal{H}}{\operatorname{argmin}} R_{\text{Emp}}(h).$$

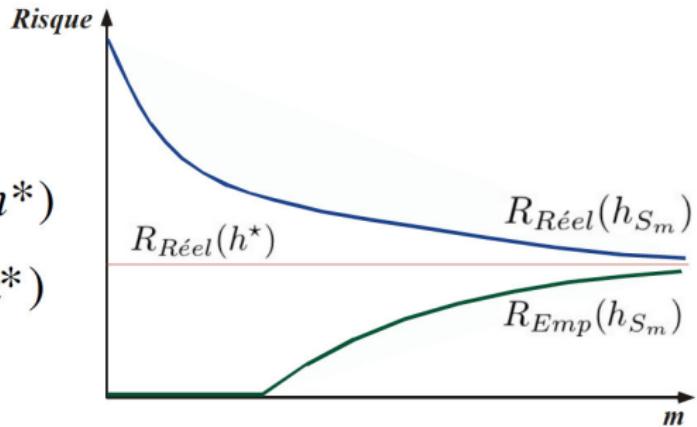
$$\forall 0 \leq \epsilon, \delta \leq 1 : \exists n \text{ tel que } P(|R_{\text{Réel}}(h_S) - R_{\text{Réel}}(h^*)| \geq \epsilon) < \delta$$



Pertinence (consistance) du principe ERM

On dit que le principe ERM est pertinent (ou consistant) si le risque réel inconnu $R(h_S)$ et le risque empirique $R_{\text{Emp}}(h_S)$ convergent vers la même limite $R(h^*)$ lorsque la taille m de l'échantillon S tend vers ∞ .

$$\left\{ \begin{array}{l} R(h_S) \xrightarrow[m \rightarrow \infty]{} R(h^*) \\ R_{Emp}(h_S) \xrightarrow[m \rightarrow \infty]{} R(h^*) \end{array} \right.$$



Erreur d'apprentissage - Bornes

- Est-ce que l'erreur d'apprentissage (c.-à-d. mesurée sur les données apprises) est un bon estimateur de l'erreur ?

$$R_{\text{Emp}}(h^*(S_N)) = \min_{h \in \mathcal{H}} R_{\text{Emp}}(h(S_N))?$$

- **NON** : c'est un estimateur biaisé, car la solution h^* trouvée est meilleure sur S_N que sur n'importe quel autre ensemble issu de l'espace des exemples.
- Peut-on borner la différence entre l'erreur d'apprentissage et l'erreur de généralisation ? $R_{\text{Emp}}(h_S) - R_{\text{Réel}}(h_S) \leq ?$
- **Oui sous certaines conditions qui dépendent de la capacité de l'espace d'hypothèses.**

Capacité de l'espace d'hypothèse

- La capacité de l'espace d'hypothèse mesure sa taille ou sa complexité.
- Pour un problème de classification :

$$R_{\text{Emp}}(h) = \frac{1}{N} \sum_{i=1}^N l_{0-1}(h(x_i) - y_i) = \frac{1}{N} \sum_{i=1}^N 1_{\{(x_i, y_i) \in S \mid h(x_i) \neq y_i\}}.$$

- La capacité de l'espace $d_{\mathcal{H}}$ est le plus grand entier n tel qu'il existe un ensemble d'exemples S_N tel que l'on puisse toujours trouver une hypothèse $h \in \mathcal{H}$ qui prédit le bon label pour tous les exemples de S_N , et ceci quel que soit la labélisation.
- Exemple : pour l'ensemble des fonctions linéaires ($y = wx + b$), dans d dimensions, la capacité est $d + 1$.

Bornes sur le risque [Vapnik]

Théorème de Vapnik

Soit $0 < \delta \leq 1$, on a une probabilité d'au moins $1 - \delta$ d'avoir

$$R_{\text{réel}} \leq R_{\text{emp}} + 2 \sqrt{\frac{d_{\mathcal{H}} \left(\log \left(\frac{2N}{d_{\mathcal{H}}} \right) + 1 \right) - \log \left(\frac{\delta}{2} \right)}{N}}.$$

Interprétation

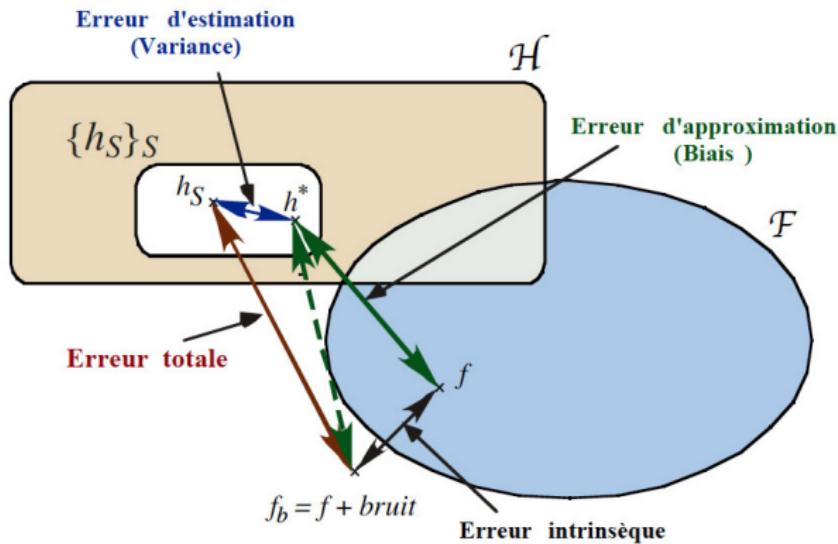
Plutôt que de chercher seulement l'hypothèse minimisant le risque empirique (valable seulement si $\frac{N}{d_{\mathcal{H}}}$ très grand), il faut aussi tenir compte des caractéristiques de l'espace des hypothèses \mathcal{H} , et chercher une hypothèse satisfaisant au mieux un compromis entre :

- un risque empirique faible : bonne adéquation aux données
- et un espace d'hypothèse d'expressivité bien réglée

Le dilemme biais-variance

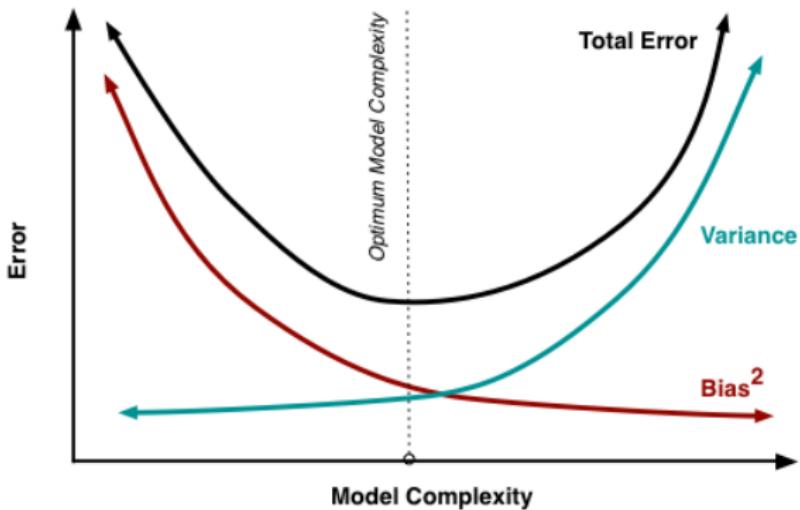
L'erreur de généralisation peut être décomposée en trois parties :

- **Le biais** : erreur systématique (par rapport à la réalité).
 - **La variance** : sensibilité du modèle aux données.
 - **Le bruit** : même la solution optimale peut être fausse (si plusieurs labels possibles pour un même exemple).

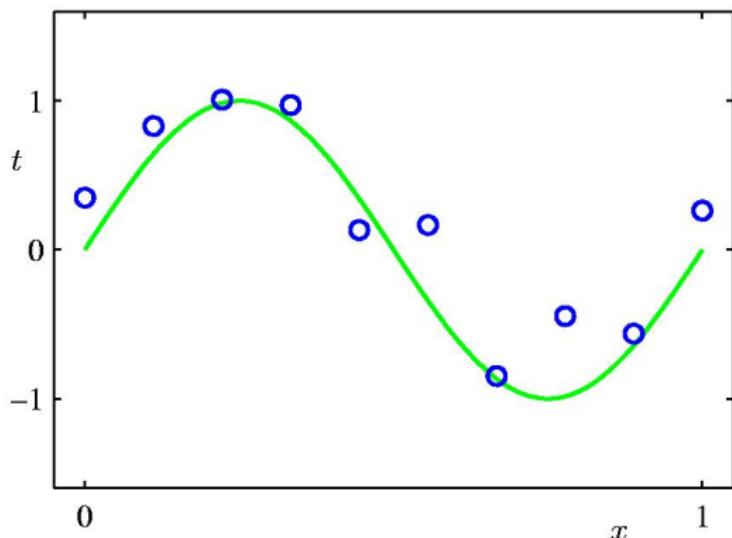


Le dilemme biais-variance

Quand la capacité $d_{\mathcal{H}}$ augmente, le biais décroît, mais la variance augmente.



Exemple : Régression par une courbe polynomiale



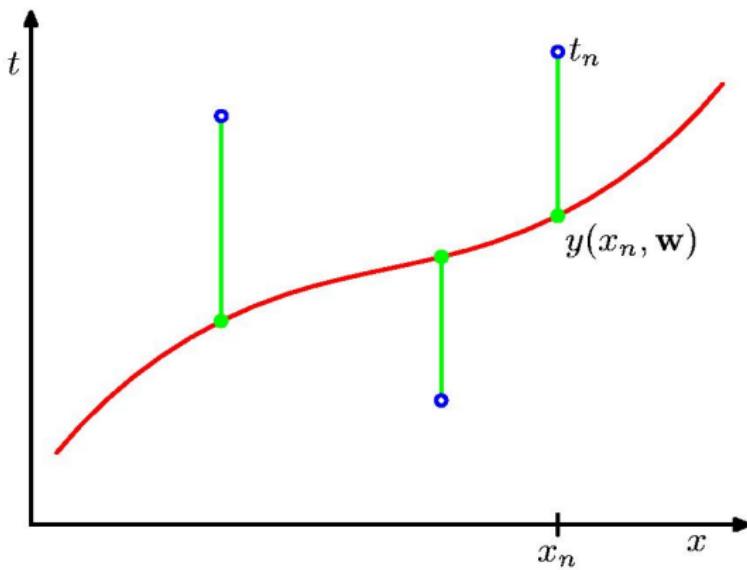
On recherche le polynôme le plus proche des points. Il est de la forme :

$$f(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \cdots + w_Mx^M = \sum_{j=0}^M w_jx^j.$$

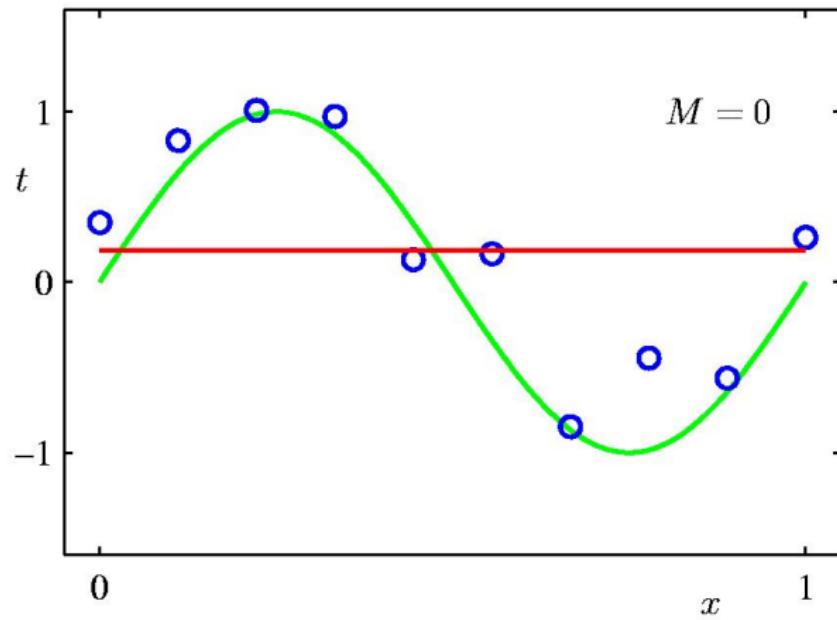
Estimation de l'erreur

Formule

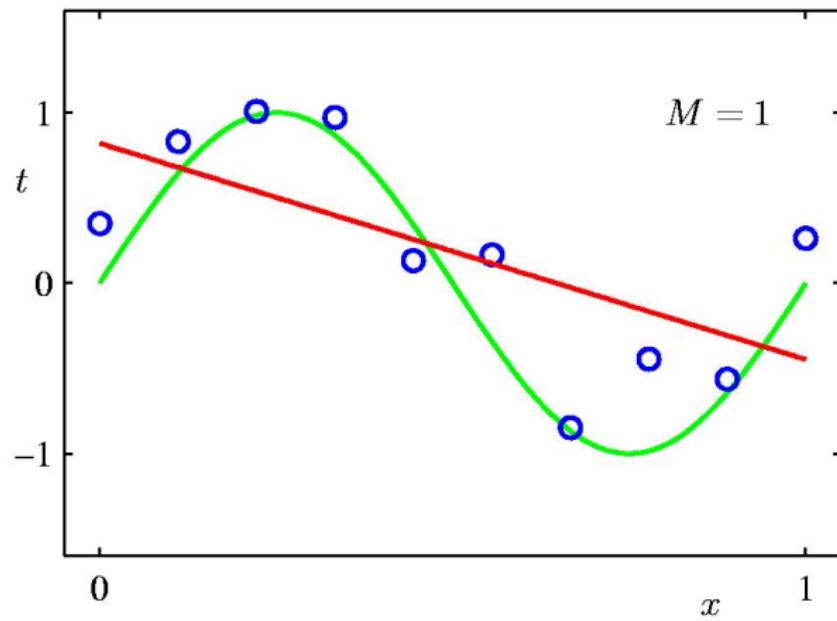
$$Err(w) = \frac{1}{2} \sum_{n=1}^N |f(x_n, w) - y_n|^2$$



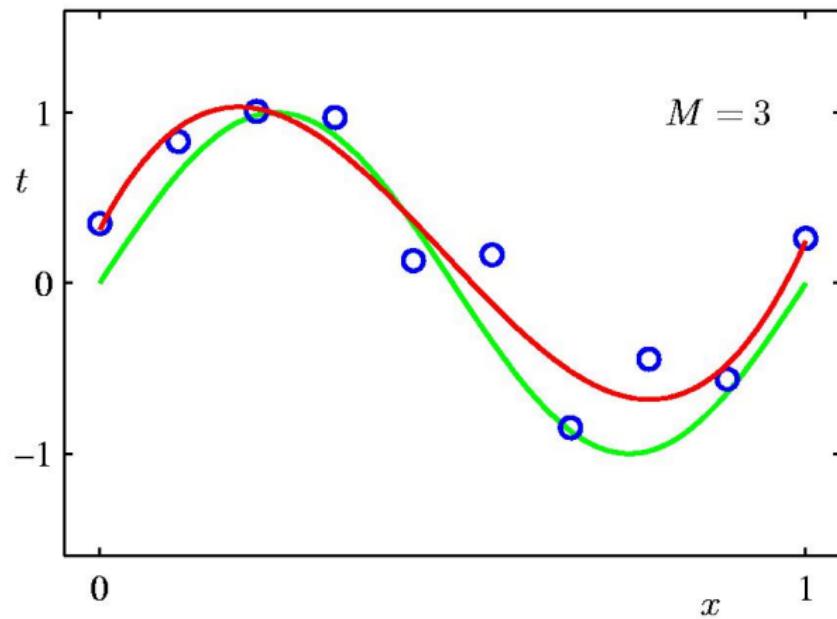
Polynôme d'ordre 0



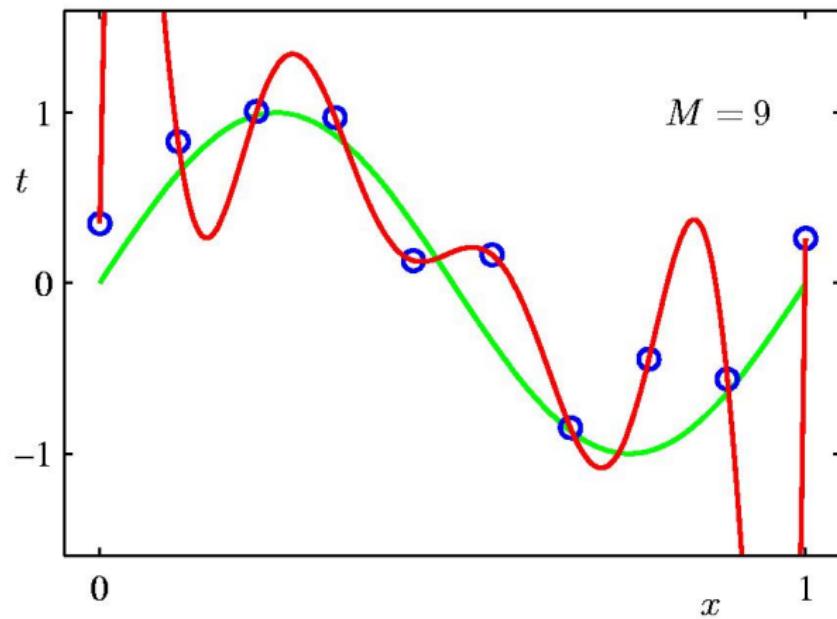
Polynôme d'ordre 1



Polynôme d'ordre 3

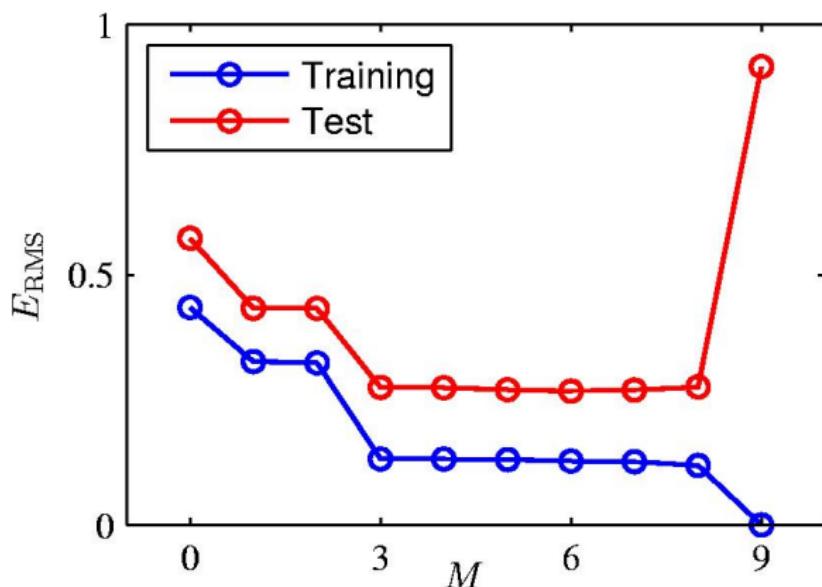


Polynôme d'ordre 9



Surapprentissage

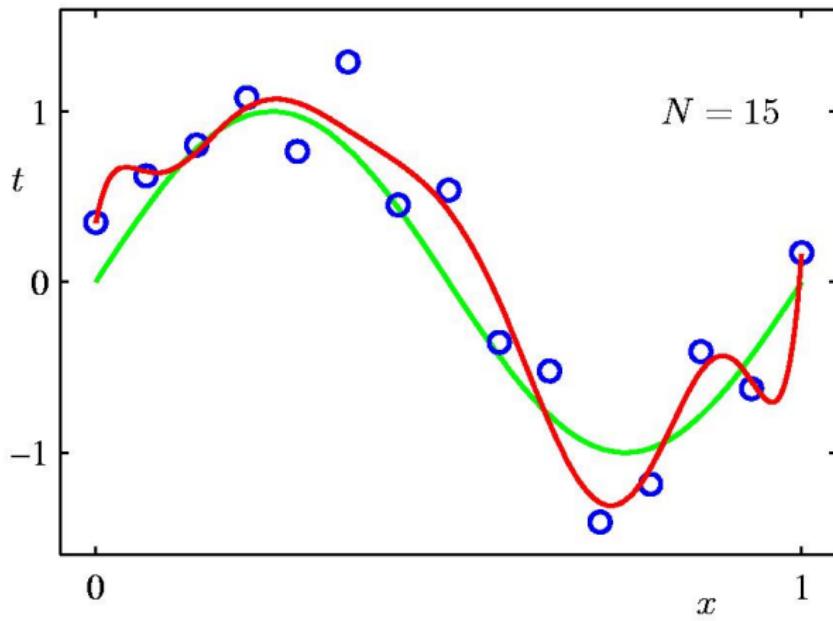
L'erreur sur l'ensemble d'apprentissage diminue en fonction du degré mais à partir d'un certain degré l'erreur sur l'ensemble de test remonte :



$$E_{RMS} = \sqrt{2\text{Err}(w^*)/M}$$

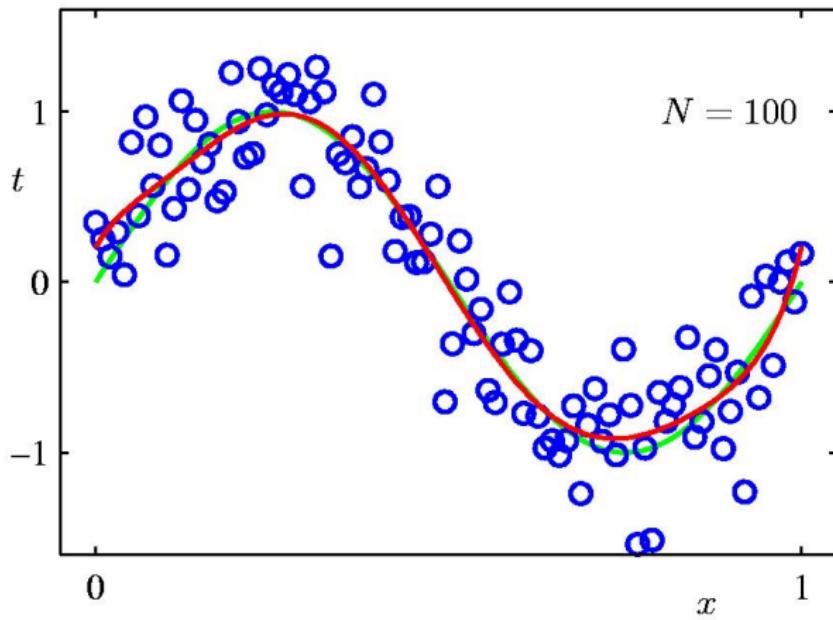
Influence de la taille de l'ensemble d'apprentissage

Polynôme de degré 9, 15 exemples d'apprentissage



Influence de la taille de l'ensemble d'apprentissage

Polynôme de degré 9, 100 exemples d'apprentissage



Solution 1 : Régularisation

Motivation

Pour éviter le surapprentissage, on ajoute un terme de régularisation dans le problème d'optimisation pour pénaliser "les solutions trop complexes" (cf. principe du rasoir d'Occam).

Le problème d'optimisation régularisé

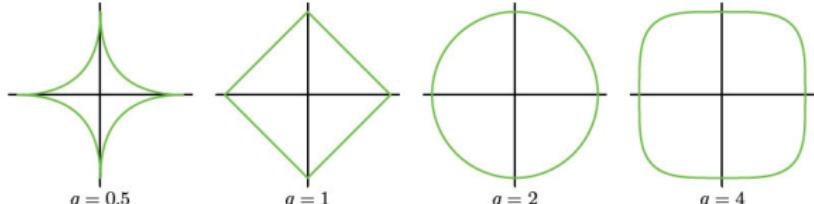
$$w^* = \arg \min_w L(w) + \lambda r(w).$$

r est une fonction de régularisation mesurant "la complexité" de la solution w . λ permet de régler le compromis entre attache aux données et "simplicité" de la solution.

La fonction de régularisation r

On choisit généralement une fonction de régularisation de la forme :

$$r(w) = \sum_{i=0}^d |w_i|^q$$



Régression d'arête (Ridge Regression)

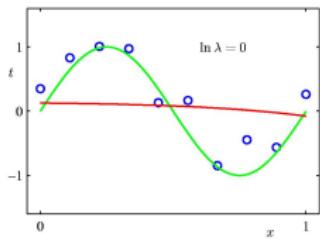
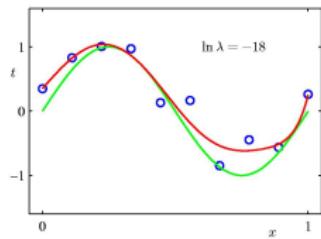
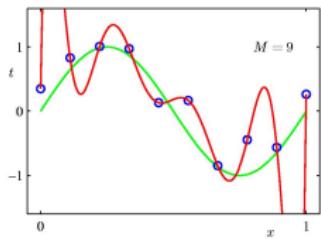
Pénalisation des coefficients trop larges

$$Err(w) = \frac{(1 - \lambda)}{2} \sum_{n=1}^N \|f(x_n, w) - y_n\|^2 + \frac{\lambda}{2} \|w\|^2.$$

Solution analytique

$$w_{\text{ridge}}^* = (\lambda I + X^\top X)^{-1} X^\top y.$$

Exemples



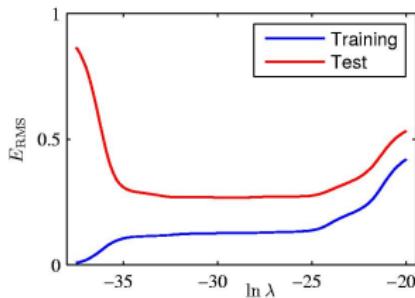
Régularisation

Pénalisation des coefficients trop larges

$$Err(w) = \frac{(1 - \lambda)}{2} \sum_{n=1}^N \|f(x_n, w) - y_n\|^2 + \frac{\lambda}{2} \|w\|^2.$$

Exemples

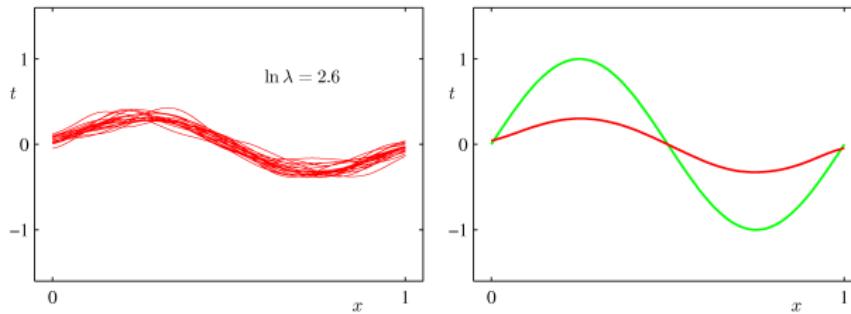
	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
w_0^*	0.35	0.35	0.13
w_1^*	232.37	4.74	-0.05
w_2^*	-5321.83	-0.77	-0.06
w_3^*	48568.31	-31.97	-0.05
w_4^*	-231639.30	-3.89	-0.03
w_5^*	640042.26	55.28	-0.02
w_6^*	-1061800.52	41.32	-0.01
w_7^*	1042400.18	-45.95	-0.00
w_8^*	-557682.99	-91.53	0.00
w_9^*	125201.43	72.68	0.01



Biais - Variance

Expérience

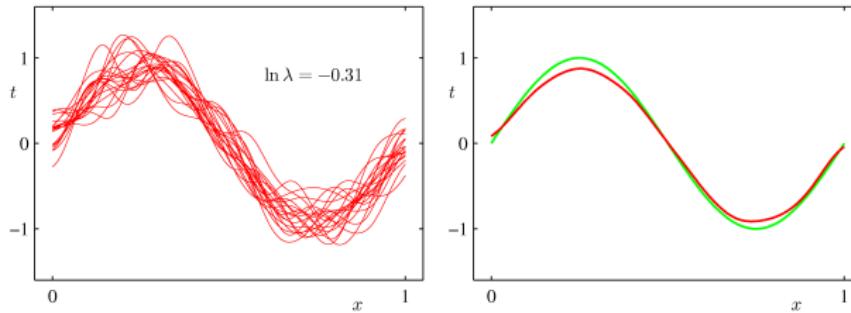
- On prend 25 ensembles de données différentes.
- On trace la régression pour chacun de ces ensembles.
- On calcule la moyenne sur tout les jeux de données que l'on compare avec la courbe originale (en vert).
- On change la valeur du paramètre du régularisation.



Solution 2 : Étude Biais - Variance

Expérience

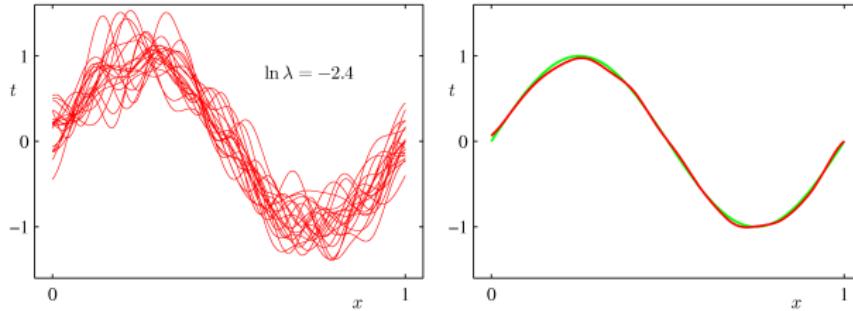
- On prend 25 ensembles de données différentes.
- On trace la régression pour chacun de ces ensembles.
- On calcule la moyenne sur tout les jeux de données que l'on compare avec la courbe originale (en vert).
- On change la valeur du paramètre du régularisation.



Biais - Variance

Expérience

- On prend 25 ensembles de données différentes.
- On trace la régression pour chacun de ces ensembles.
- On calcule la moyenne sur tout les jeux de données que l'on compare avec la courbe originale (en vert).
- On change la valeur du paramètre du régularisation.



Décomposition en Biais/Variance

Notations

On cherche à apprendre le modèle bruité : $y = F(x) + \nu$. On approche F par $f(x, \hat{w}) \in \mathcal{H}$.

Pour un ensemble X , on a $\hat{f}(x) = f(x, \hat{w})$ et $\bar{f}(x) = Ex[f(x, \hat{w})]$ la moyenne sur plusieurs ensembles de données.

Biais/Bruit/Variance

- On peut montrer que l'on a

$$Ex \left[(y_0 - \hat{f}(\mathbf{x}_0))^2 \right] = (y_0 - \bar{f}(\mathbf{x}_0))^2 + \underbrace{Ex \left[(\hat{f}(\mathbf{x}_0) - \bar{f}(\mathbf{x}_0))^2 \right]}_{\text{variance}}$$

- Avec

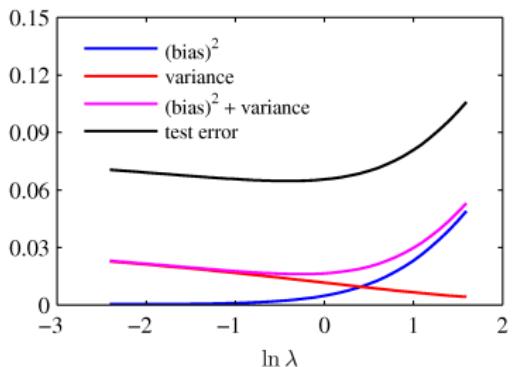
$$(y_0 - \bar{f}(\mathbf{x}_0)) = \underbrace{(y_0 - F(\mathbf{x}_0))}_{\text{noise}} + \underbrace{(F(\mathbf{x}_0) - \bar{f}(\mathbf{x}_0))}_{\text{biais}}$$

Décomposition en Biais/Variance

Décomposition

$$E[\text{l'erreur quadratique}] = \text{biais}^2 + \text{variance} + \text{noise}^2$$

- On ne peut rien faire pour réduire le bruit.
 - Le biais est minimisé par notre fonction de coût quadratique.
 - La variance est réduite par le terme de régularisation et/ou une validation croisé.



Idée de la validation croisé

Idée

On pourrait calculer l'erreur sur un échantillon indépendant n'ayant pas participé à l'estimation du modèle.

Les jeux de données

On peut découper l'ensemble des données trois jeux de données :

- **Train** : données permettant l'apprentissage,
- **Val** : données permettant de valider l'apprentissage et notamment régler les paramètres de la méthode d'apprentissage,
- **Test** : données de test permettant de mesurer les performances de la méthode.

Chacun de ces ensembles doit être statiquement représentatif des données que vous pouvez rencontrer.

Vous pouvez, par exemple, tirer au hasard de façon uniforme chacun des trois ensembles en respectant l'équilibre des classes.

La validation croisée

Principe

- Itérer l'estimation de l'erreur sur plusieurs échantillons de validation puis en calculer la moyenne.
- C'est indispensable pour réduire la variance et améliorer la précision lorsque la taille de l'échantillon initial est trop réduite pour en extraire un échantillon de validation ou test de taille suffisante.
- Plusieurs stratégies existent : k-fold, leave-one-out,...

k-fold cross-validation

- Découper aléatoirement l'échantillon d'apprentissage D en K parts de tailles approximativement égales.
- Répéter K fois l'opération qui consiste à mettre de côté l'une des parties, estimer le modèle sur les $K - 1$ parties restantes, calculer l'erreur sur chacune des observations n'ayant pas participé à l'estimation.
- Moyenner toutes ces erreurs pour aboutir à l'estimation par validation croisée.

Leave one out

On utilise la même stratégie que la k-fold cross-validation mais la partie mise de coté a une taille d'un exemple.

K-fold cross validation



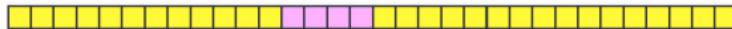
Apprend sur jaune, test sur rose → erreur₁



Apprend sur jaune, test sur rose → erreur₂



Apprend sur jaune, test sur rose → erreur₃



Apprend sur jaune, test sur rose → erreur₄



Apprend sur jaune, test sur rose → erreur₅



Apprend sur jaune, test sur rose → erreur₆



Apprend sur jaune, test sur rose → erreur₇

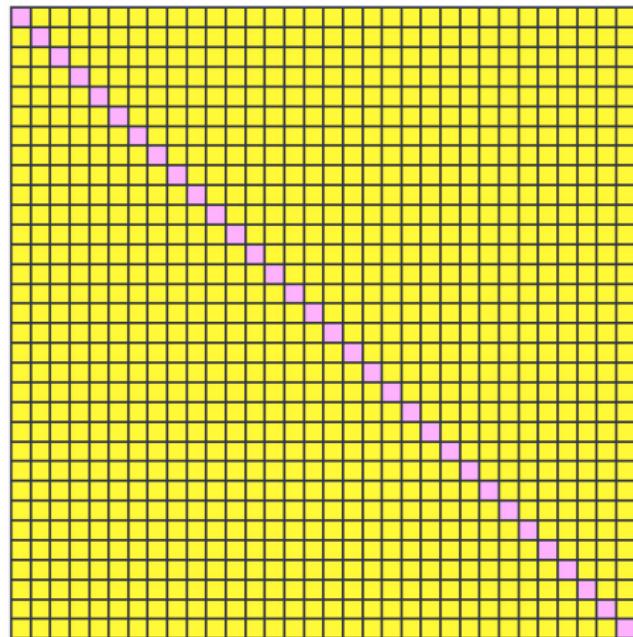


Apprend sur jaune, test sur rose → erreur₈

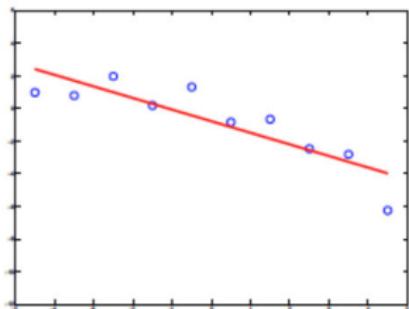
$$\text{erreur} = \sum \text{erreur}_i / k$$

Leave one out cross validation

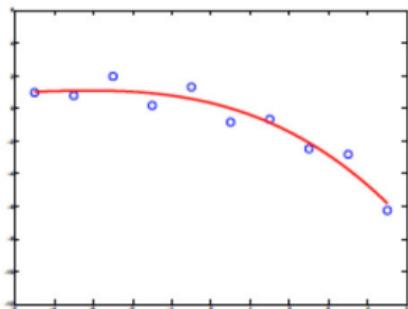
Les modèles sont appris sur toutes les données sauf le i -ème exemple.



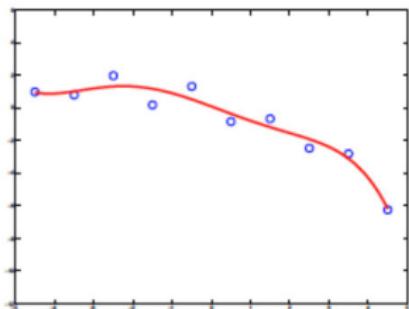
Exemple de cross-validation



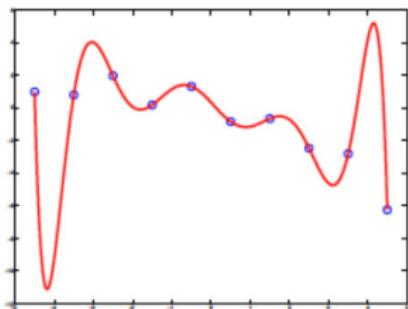
$$m = 1 : L = 1.4, \hat{L}_{cv} = 2.6$$



$$m = 3 : L = 0.4, \hat{L}_{cv} = 1.3$$



$$m = 5 : L = 0.3, \hat{L}_{cv} = 2.7$$



$$m = 10 : L = 0, \hat{L}_{cv} = 4 \times 10^4$$

Sélection de modèle par validation simple

Algorithme

- Sélection d'une famille de fonction avec un hyperparamètre θ .
- Divisé l'ensemble d'apprentissage D_n en deux :
 - $D^{\text{tr}} = \{x_1, \dots, x_{\text{tr}}\}$,
 - $D^{\text{val}} = \{x_{\text{tr}+1}, \dots, x_{\text{tr}+\text{val}}\}$,
 - On a $n = \text{tr} + \text{val}$.
- Pour toutes les valeurs de θ_m de l'hyperparamètre θ faire :
 - Sélectionner $f_{\theta_m}^*(D^{\text{tr}}) = \arg \min_{f \in \mathcal{F}_{\theta_m}} \hat{L}(f, D^{\text{tr}})$.
 - Estimer $L(f_{\theta_m}^*)$ avec $\hat{L}(f, D^{\text{val}}) = \frac{1}{\text{val}} \sum_{x_i \in D^{\text{val}}} \text{loss}(x_i, f_{\theta_m}^*)$.
- Choisir $\theta^* = \arg \min_{\theta_m} L(f_{\theta_m}^*)$.
- Retourner $f^* = \arg \min_{f \in \mathcal{F}_{\theta_m^*}} \hat{L}(f, D_n)$.

Sélection de modèle par cross validation croisée

Algorithme

- Sélection d'une famille de fonction avec un hyperparamètre θ .
- Divisé l'ensemble d'apprentissage D_n en K ensembles distincts de même taille D^1, \dots, D^K .
- Pour toutes les valeurs de θ_m de l'hyperparamètre θ faire :
 - Pour chaque ensemble D^k et $\bar{D}^k = D_n \setminus D^k$ faire :
 - Sélectionner $f_{\theta_m}^*(\bar{D}^{\text{tr}}) = \arg \min_{f \in \mathcal{F}_{\theta_m}} \hat{L}(f, \bar{D})$.
 - Estimer $L(f_{\theta_m}^*(\bar{D}^k))$ avec $\hat{L}(f, D^k) = \frac{1}{|D^k|} \sum_{x_i \in D^k} \text{loss}(x_i, f_{\theta_m}^*)$.
 - Choisir $L(f^*)$ avec $\frac{1}{K} \sum_k L(f_{\theta_m}^*(\bar{D}^k))$.
 - Choisir $\theta^* = \arg \min_{\theta_m} L(f_{\theta_m}^*)$.
 - Retourner $f^* = \arg \min_{f \in \mathcal{F}_{\theta_m^*}} \hat{L}(f, D_n)$.