



Théorie et Algorithmes de l'Apprentissage Automatique (T3A)

Simon BERNARD
simon.bernard@univ-rouen.fr

- 11 séances d'1h30 de CM/TD/CC
- 11 séances d'1h30 de TP
 - Travail individuel (pas de binômes)
 - Sur les machines des salles de TP ou vos propres ordinateurs
 - Python
- 1 ou 2 examens écrits
- Rendus de TP + épreuve de TP

1. Fondements de l'apprentissage automatique
2. Régression linéaire
3. Classification linéaire
4. Support Vector Machine
5. SVM non linéaire et noyaux
6. Sélection de modèles



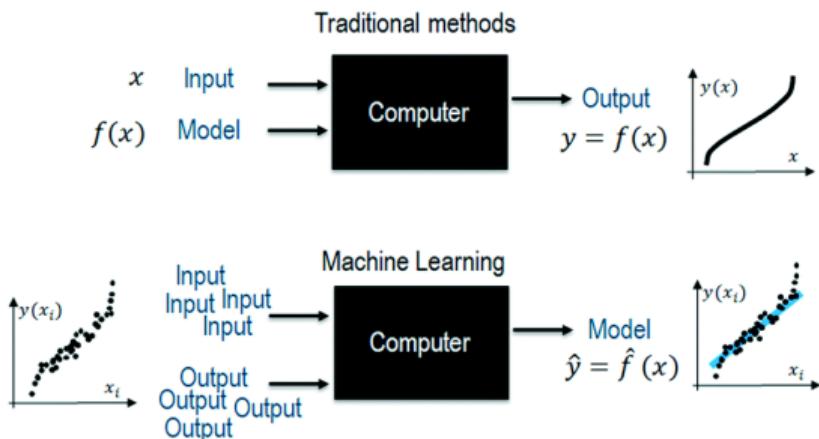
Théorie et Algorithmes de l'Apprentissage Automatique (T3A)

1 - Fondements de l'apprentissage automatique

Apprentissage automatique *(Machine Learning)*

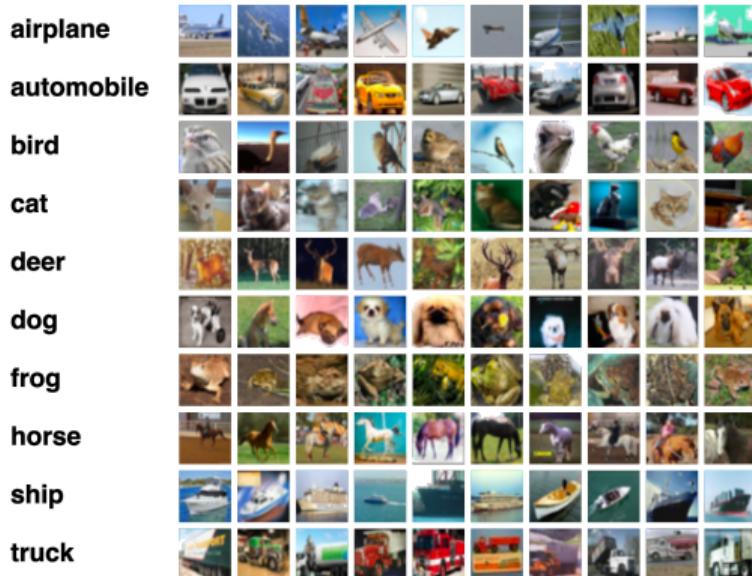
Apprentissage automatique

Rendre une machine capable d'accomplir des tâches complexes sans avoir été explicitement programmée dans ce but mais en se basant sur l'expérience (données exemples)



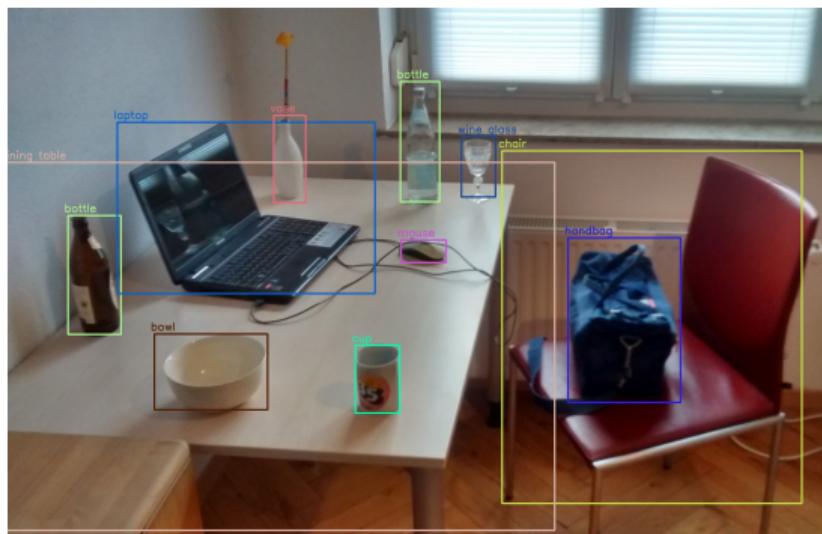
Traitements automatiques des images

Classification d'images



Traitements automatiques des images

Détection d'objets

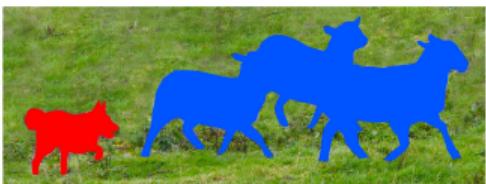


Traitement automatique des images

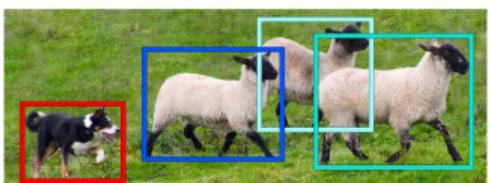
Segmentation d'images ou d'instances



Image Recognition



Semantic Segmentation



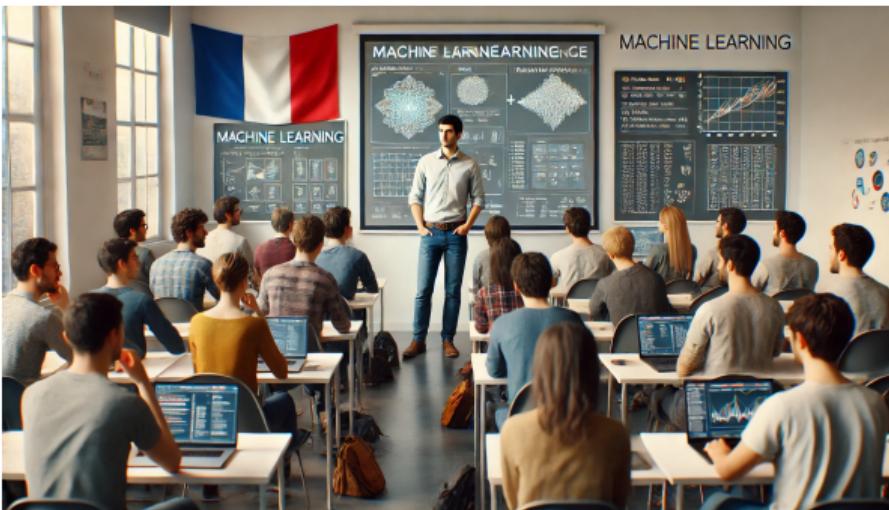
Object Detection



Instance Segmentation

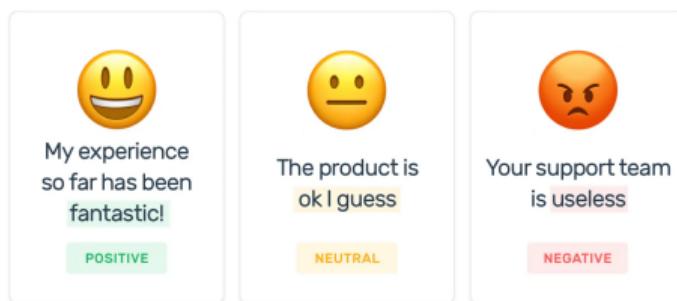
Traitements automatiques des images

Génération d'images



Traitement automatique du langage (NLP)

Classification de textes



When Sebastian Thrun PERSON started at Google ORG in 2007 DATE, few people outside of the company took him seriously. "I can tell you very senior CEOs of major American NORP car companies would shake my hand and turn away because I wasn't worth talking to," said Thrun PERSON, now the co-founder and CEO of online higher education startup Udacity, in an interview with Recode ORG earlier this week DATE.

A little less than a decade later DATE, dozens of self-driving startups have cropped up while automakers around the world clamor, wallet in hand, to secure their place in the fast-moving world of fully automated transportation.

Traitement automatique du langage (NLP)

Génération de texte

The first recorded travels by Europeans to China and back date from this time. The most famous traveler of the period was the Venetian Marco Polo, whose account of his trip to "Cambaluc," the capital of the Great Khan, and of life there astounded the people of Europe. The account of his travels, Il milione (or, The Million, known in English as the Travels of Marco Polo), appeared about the year 1299. Some argue over the accuracy of Marco Polo's accounts due to the lack of mentioning the Great Wall of China, tea houses, which would have been a prominent sight since Europeans had yet to adopt a tea culture, as well the practice of foot binding by the women in capital of the Great Khan. Some suggest that Marco Polo acquired much of his knowledge through contact with Persian traders since many of the places he named were in Persian.

How did some suspect that Polo learned about China instead of by actually visiting it?

Answer: through contact with Persian traders



"girl in pink dress is jumping in air."



"black and white dog jumps over bar."



"young girl in pink shirt is swinging on swing."



"man in blue wetsuit is surfing on wave."



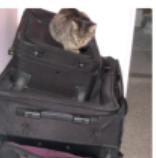
"little girl is eating piece of cake."



"baseball player is throwing ball in game."

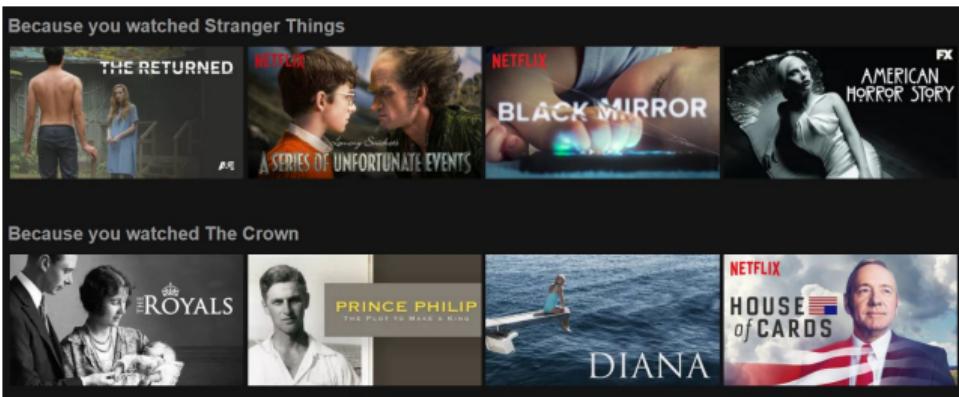
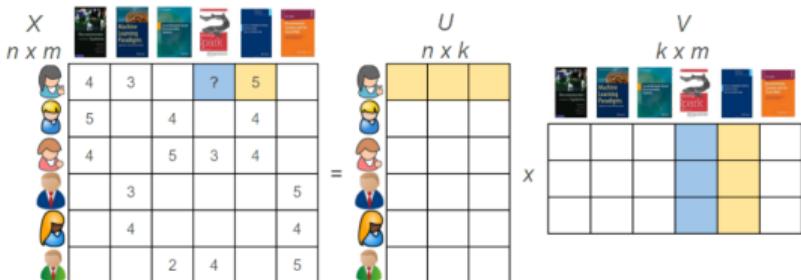


"woman is holding bunch of bananas."

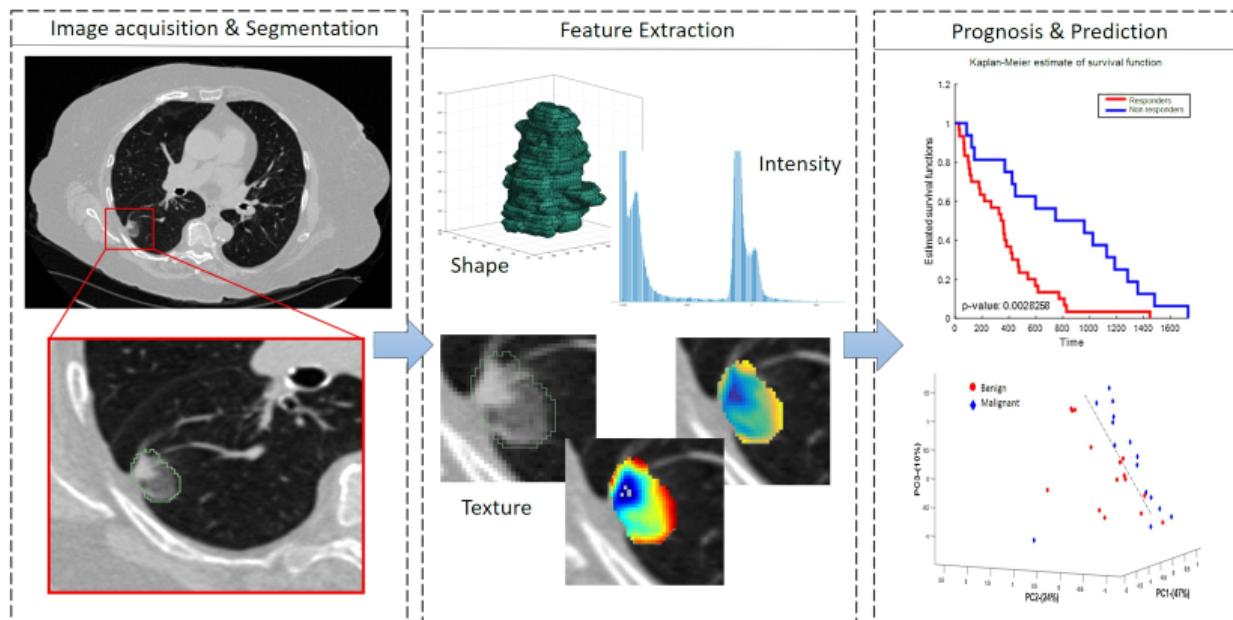


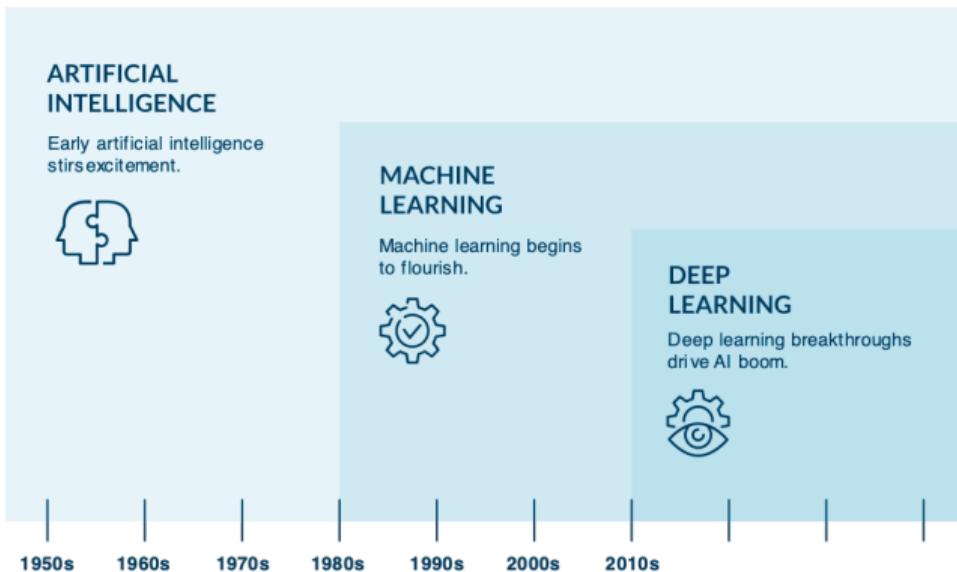
"black cat is sitting on top of suitcase."

Recommandation

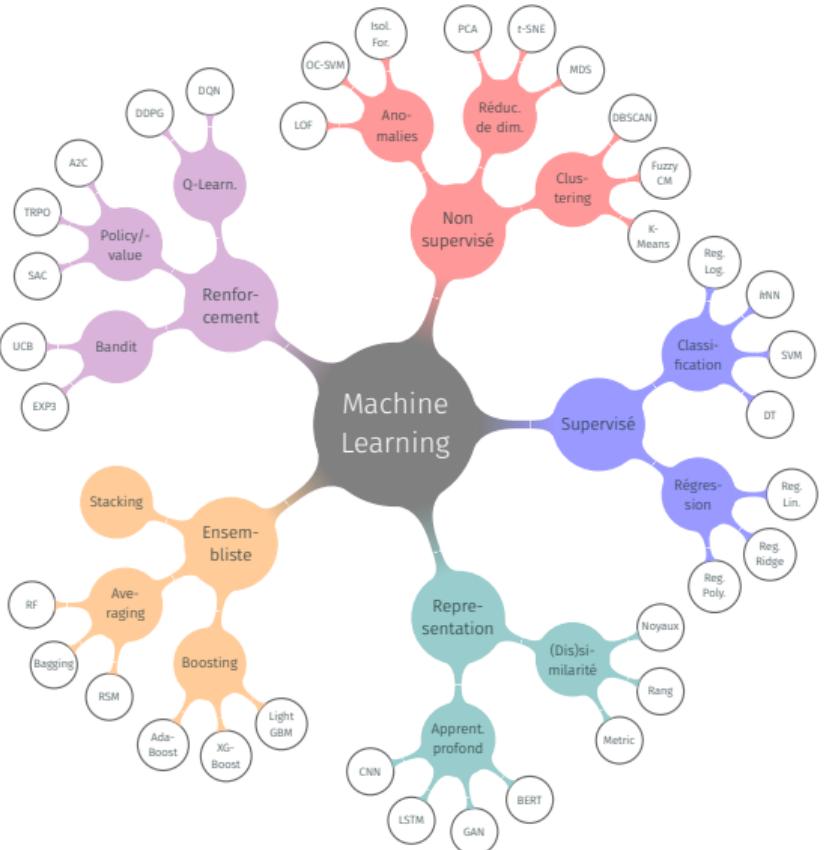


Aide au diagnostic médical





L'IA d'aujourd'hui est basé sur l'apprentissage profond (une technique de ML)



Apprentissage non supervisé

- *Clustering* : Organiser des objets en groupes homogènes
- Estimation de distribution : Estimer la distribution des données (e.g. pour la génération)
- Réduction de dimension : Plonger les données dans un espace de petite dimension

Apprentissage supervisé (focus du cours)

- régression
- Classification

Apprentissage supervisé

- L'apprentissage se fait à partir de données exemples
- Ces données représentent des "objets" du problème (ex. : des patients, des films, etc.)
- Une donnée (**instance**) est décrite par d valeurs (**caractéristiques**)

$$\mathbf{x} = \{x^{(1)}, x^{(2)}, \dots, x^{(d)}\}, \mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$$

- Les caractéristiques forment une description *pertinente* des objets
- Une **ensemble d'apprentissage** est un ensemble de n données exemples

$$\mathbf{X} = \{\mathbf{x}_i \in \mathcal{X}, i = 1, \dots, n\}$$

- Les données sont associées à une variable cible Y que l'on souhaite prédire
- \mathcal{X} et \mathcal{Y} sont les domaines de définition des entrées/sorties
- On suppose qu'il existe une fonction f (inconnue) :

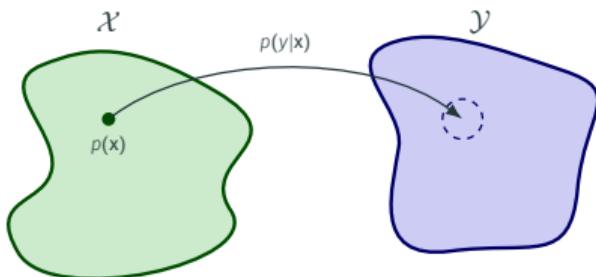
$$f : \mathcal{X} \rightarrow \mathcal{Y}$$

- L'apprentissage vise à estimer f avec une **fonction de prédiction** (ou **modèle**) h :

$$\hat{y} = h(x), \forall x \in \mathcal{X}$$

où \hat{y} est la **prédiction** de Y pour x

- \mathcal{X}, \mathcal{Y} munis d'une loi de probabilité jointe $p_{X,Y}$ (inconnue) :



- Pour obtenir h on dispose d'un ensemble de couples $(x_i, y_i) \sim p_{X,Y}$:

$$\mathcal{D} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1, \dots, n\}$$

où y_i est la vraie valeur de Y associée à x_i (*la vérité terrain*)

Deux types de tâches en fonction de la nature de \mathcal{Y} :

1. **Régression** : \mathcal{Y} est un ensemble continu de réels

$$\mathcal{Y} \subset \mathbb{R}$$

2. **Classification** : \mathcal{Y} est un ensemble discret de c classes

$$\mathcal{Y} = \{\lambda_1, \lambda_2, \dots, \lambda_c\}, c \geq 2$$

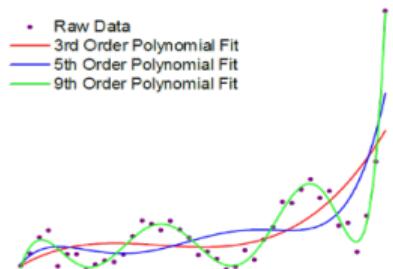
$\forall \mathbf{x}$, h prédit une classe ($\hat{y} \in \mathcal{Y}$) ou

$$\hat{y} = \{\hat{p}(\lambda_1|\mathbf{x}), \hat{p}(\lambda_2|\mathbf{x}), \dots, \hat{p}(\lambda_c|\mathbf{x})\}$$

- L'espace des h possibles est noté \mathcal{H} et appelé **l'espace des hypothèses**
- Bien souvent, h est défini dans \mathcal{H} à l'aide de paramètres $\theta \in \Theta$
- Dans ce cas, **trouver le meilleur h revient à trouver le meilleur θ**

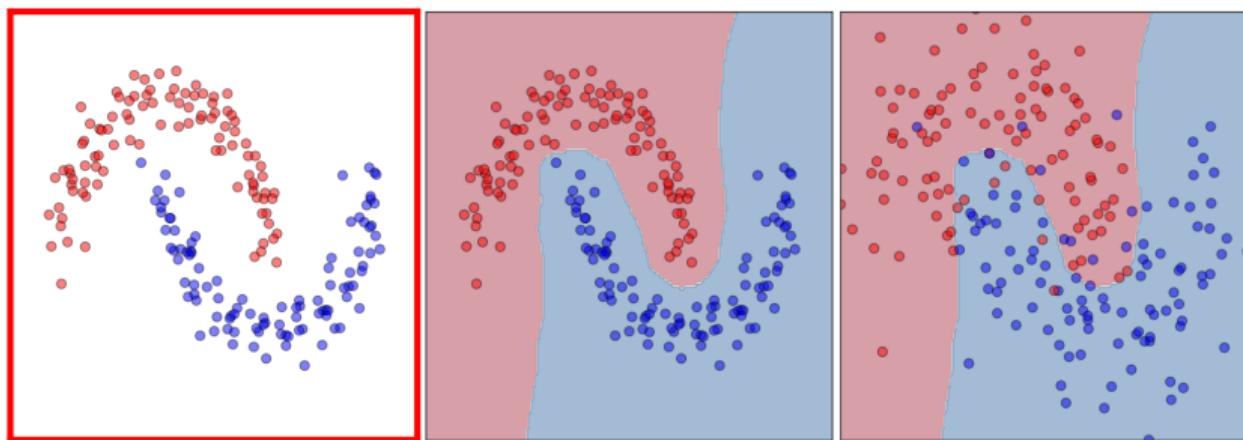
Exemple :

- \mathcal{H} : l'ensemble des fonctions polynomiales de degré p
- $\Theta \subset \mathbb{R}^{p+1}$: coefficients des polynômes de degré p



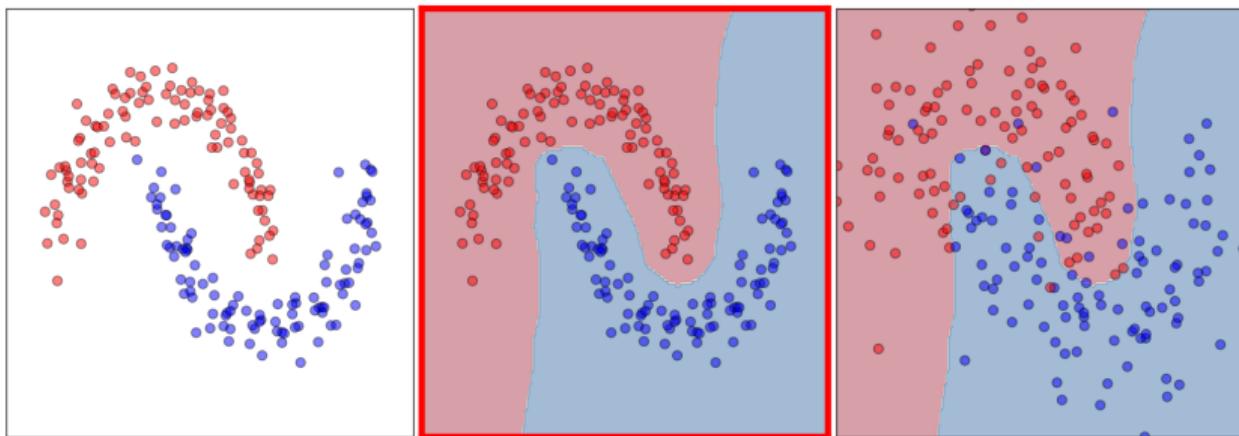
1. ensemble d'apprentissage, composé de n instances étiquetées (i.e. pour lesquelles on connaît la vraie classe) :

$$\mathcal{D} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1, \dots, n\}$$



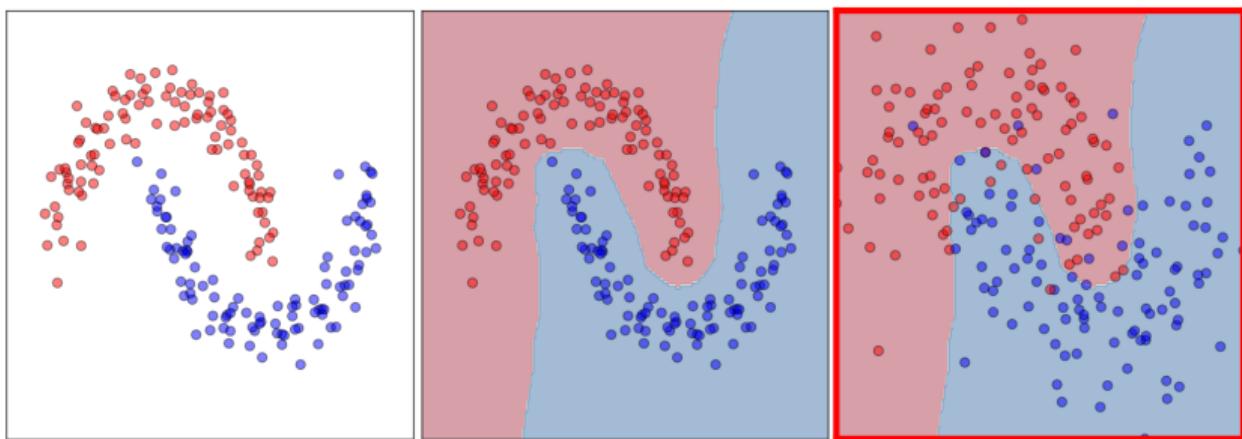
2. modèle h appris sur ces données. Ici $\mathcal{Y} = \{\text{rouge}, \text{bleu}\}$.

$$h : \mathcal{X} \rightarrow \mathcal{Y}$$



3. h prédit "rouge" ou "bleu" pour chaque nouvelle instance

$$\hat{y} = h(\mathbf{x}), \forall \mathbf{x} \notin \mathcal{D}$$



- Quelle méthode pour apprendre le meilleur h possible?

$$h^* = \arg \min_{h \in \mathcal{H}} P_{\mathcal{X}, \mathcal{Y}}(h(\mathbf{x}) \neq y)$$

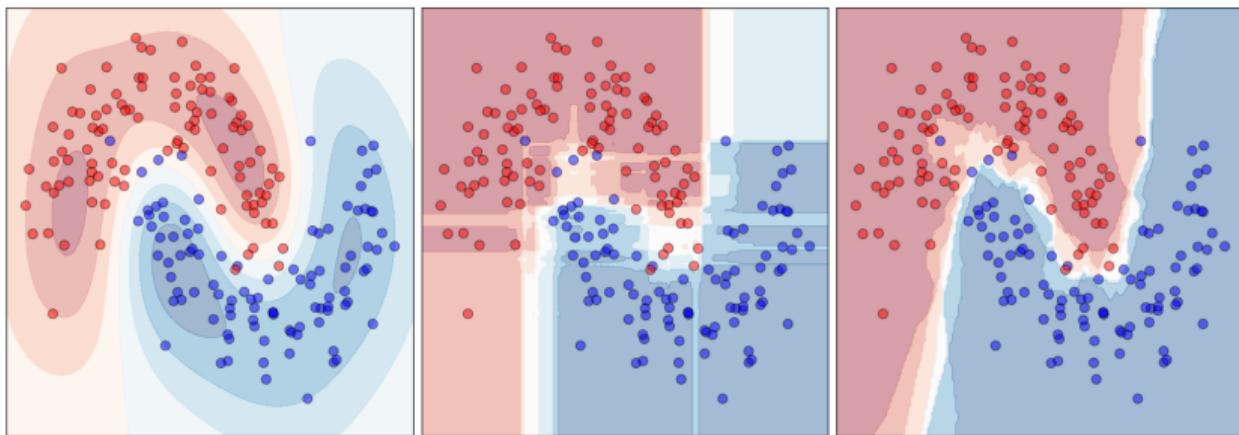


Figure 1 – SVM (RBF kernel), Forêts Aléatoires (100 arbres), K-Plus Proches Voisins (K=5)

Minimisation du risque empirique

- L'enjeu des méthodes d'apprentissage est de trouver le meilleur h possible
- Qu'est-ce qu'un h meilleur qu'un autre ?
- Notion de **fonction de perte (*loss function*)**

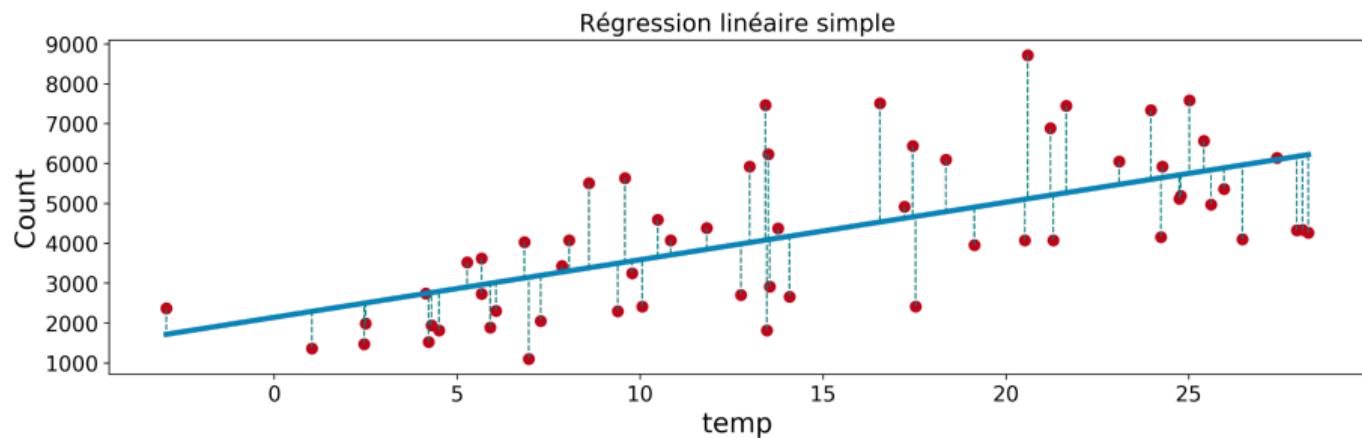
$$\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$$

tel que $\ell(y, h(x)) \geq 0$ mesure à quel point la prédiction $h(x)$ est proche de la vraie valeur y

Fonction de perte (*loss function*)

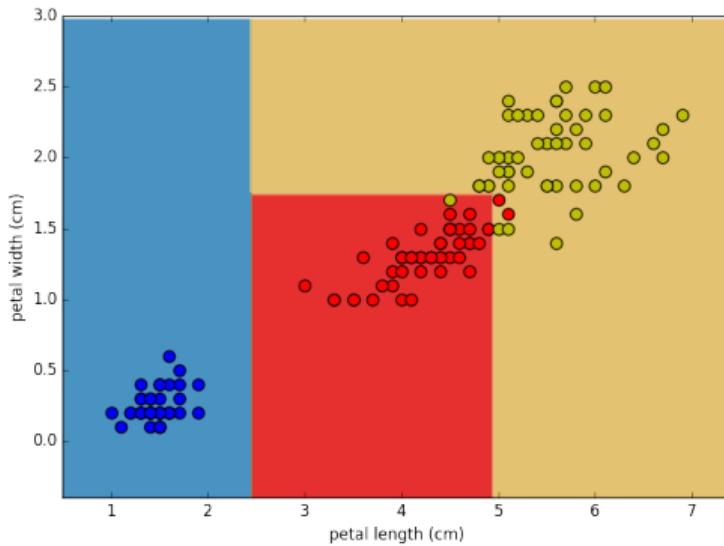
Exemple : *Quadratic loss* (ou moindres carrés) pour la régression

$$\ell(y, h(x)) = (y - h(x))^2$$



Exemple : 0-1 loss pour la classification

$$\ell(y, h(\mathbf{x})) = \begin{cases} 0 & \text{si } h(\mathbf{x}) = y \\ 1 & \text{sinon} \end{cases}$$



- La "qualité" globale d'un h est définie par le **risque réel** (ou **erreur en généralisation**) :

$$R(h) = E_{(x,y) \sim p_{X,Y}} [\ell(y, h(x))]$$

- Cela signifie que pour une distribution $p_{X,Y}$ donnée et pour un espace des hypothèses \mathcal{H} fixé, **le modèle optimal** $h^* \in \mathcal{H}$ est celui qui minimise le risque réel :

$$h^* = \arg \min_{h \in \mathcal{H}} R(h)$$

- On ne peut pas calculer $R(h)$ car on ne connaît pas $p_{X,Y}$
- Mais, on dispose d'un ensemble d'apprentissage pour calculer une estimation, **le risque empirique** :

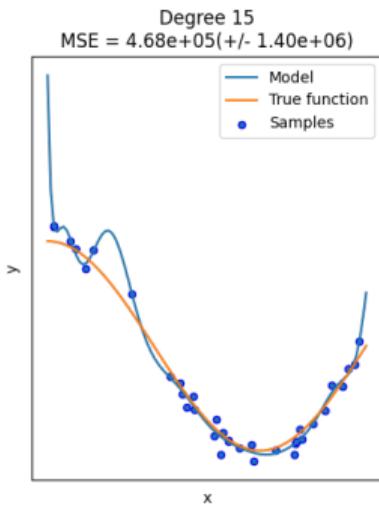
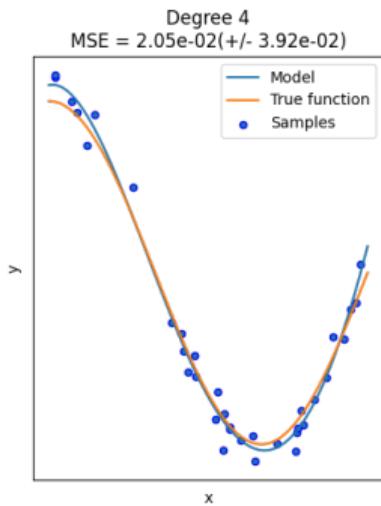
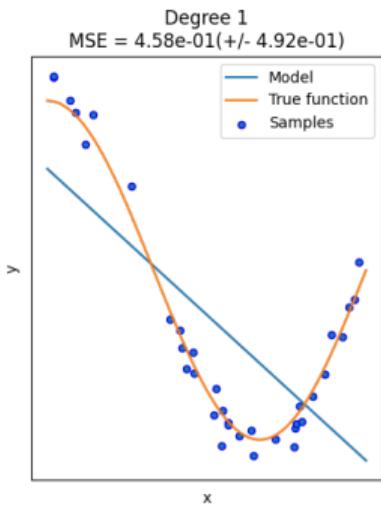
$$R_{\mathcal{D}}(h) = \frac{1}{n} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}} \ell(y_i, h(\mathbf{x}_i))$$

- Et donc minimiser le risque empirique pour trouver $h_{\mathcal{D}}^*$

$$h_{\mathcal{D}}^* = \arg \min_{h \in \mathcal{H}} R_{\mathcal{D}}(h)$$

Sous-apprentissage et sur-apprentissage

- Que se passe-t-il si on considère un \mathcal{H} tel que les h candidats sont trop "simples" ou trop "complexes" par rapport au problème ?



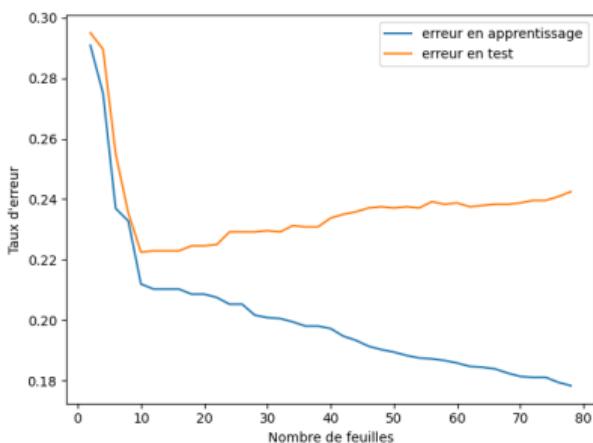
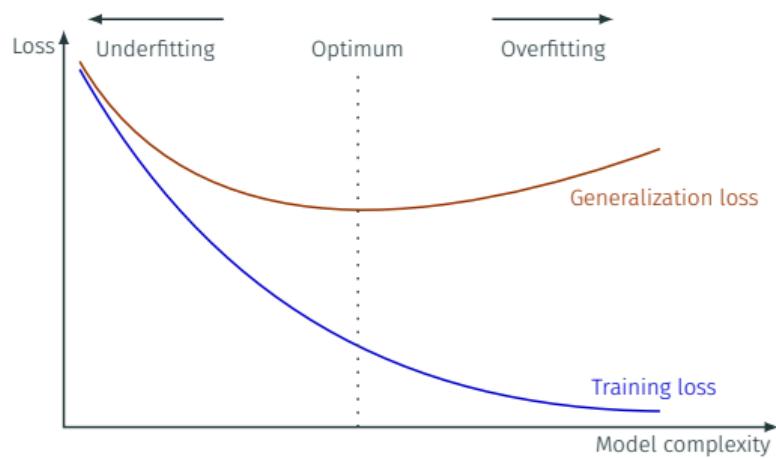
- Notons $\mathcal{Y}^{\mathcal{X}}$ l'ensemble de TOUS les $h : \mathcal{X} \rightarrow \mathcal{Y}$
- On appelle **modèle oracle** ou **modèle optimal de Bayes** le modèle h_B tel que :

$$h_B = \arg \min_{h \in \mathcal{Y}^{\mathcal{X}}} R(h)$$

En d'autres termes, le meilleur modèle h que l'on peut possiblement trouver

- La capacité d'un espace d'hypothèses \mathcal{H} induite par un algorithme d'apprentissage représente son aptitude à trouver un bon modèle $h \in \mathcal{H}$, quelle que soit la complexité de f
- En pratique, cette capacité est contrôlée par des hyper-paramètres :
 - Le degré des polynômes
 - Le nombre de voisins d'un kPPV
 - Le nombre d'itérations pour un apprentissage incrémental
 - Les paramètres de régularisation
 - La profondeur d'un arbre de décision

- Si la capacité de \mathcal{H} est trop faible :
 - $h_B \notin \mathcal{H}$ et $R(h) - R(h_B)$ est grand pour tout $h \in \mathcal{H}$
 - Les modèles h sous-apprennent les données
- Si la capacité de \mathcal{H} est trop élevée :
 - $h_B \in \mathcal{H}$ ou $R(h^*) - R(h_B)$ est petit
 - Mais $h_{\mathcal{D}}^*$ peut être arbitrairement bon sur les données d'apprentissage tel que
$$R(h_{\mathcal{D}}^*) \geq R(h_B) \geq R_{\mathcal{D}}(h_{\mathcal{D}}^*) \geq 0$$
 - Le risque empirique (erreur en apprentissage) est faible au détriment du risque réel (erreur en généralisation)
 - Le modèle $h_{\mathcal{D}}^*$ sur-apprend les données



- En cas de sur-apprentissage, on a

$$R(h_{\mathcal{D}}^*) \geq R_B \geq R_{\mathcal{D}}(h_{\mathcal{D}}^*) \geq 0$$

Ce qui signifie que $R_{\mathcal{D}}(h_{\mathcal{D}}^*)$ est un mauvais estimateur de $R(h_{\mathcal{D}}^*)$

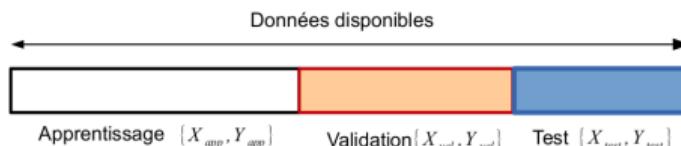
- On peut estimer l'erreur en généralisation en utilisant un ensemble de données indépendant de \mathcal{D} , i.e. un ensemble de test \mathcal{D}_{test} :

$$R_{\mathcal{D}_{test}}(h_{\mathcal{D}}^*) = \frac{1}{m} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{D}_{test}} L(y_i, h_{\mathcal{D}}^*(\mathbf{x}_i))$$

où m est le nombre de données dans \mathcal{D}_{test}

- **Règle d'or :** l'ensemble de test ne doit pas être utilisé QUE pour cela

Si n , le nombre de données d'apprentissage dans \mathcal{D} , est grand

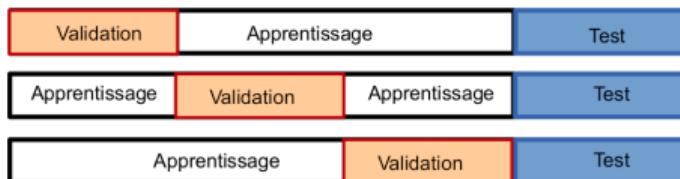


1. Découper aléatoirement $\mathcal{D} = \mathcal{D}_{app} \cup \mathcal{D}_{val} \cup \mathcal{D}_{test}$
2. Apprendre les modèles candidats sur \mathcal{D}_{app}
3. Évaluer leurs performances sur \mathcal{D}_{val}
4. Sélectionner le modèle qui donne la meilleure performance sur \mathcal{D}_{val}
5. Tester le modèle retenu sur \mathcal{D}_{test}

Remarque :

- \mathcal{D}_{test} n'est utilisé qu'une seule fois!
- Le modèle retenu est le meilleur sur \mathcal{D}_{val}

Si n est petit (ou pour fiabiliser)



1. Découper aléatoirement $\mathcal{D} = \mathcal{D}_{app} \cup \mathcal{D}_{test}$
2. Découper aléatoirement $\mathcal{D}_{app} = \mathcal{D}_1 \cup \dots \cup \mathcal{D}_K$ en K sous-ensembles disjoints (*fold*)
3. Pour $k = 1, \dots, K$
 - 3.1 Mettre de coté \mathcal{D}_k
 - 3.2 Apprendre un modèle sur les $K - 1$ *fold* restants
 - 3.3 Évaluer sa performance sur \mathcal{D}_k
4. Moyenner les K mesures de performances

Procédure appelée validation croisée (*K-fold cross-validation*), mais il existe d'autres techniques

Les étapes pour mener à bien une tâche d'apprentissage automatique :

1. Identification de la tâche (entrées, sorties, etc.)
2. Acquisition de données représentatives du problème
3. Préparation des données (extraction/sélection de caractéristiques, réduction de dimension, normalisation, etc.)
4. Choix d'un algorithme et/ou d'un espace des hypothèses
5. Entraînement/sélection de modèle (hyper-paramètres)
6. Validation/analyse des performances en test