ENSEA

Introduction à l'apprentissage machine

24/03/2016
Geoffroy.Peeters@ircam.fr
UMR SMTS IRCAM CNRS UPMC

Apprentissage machine

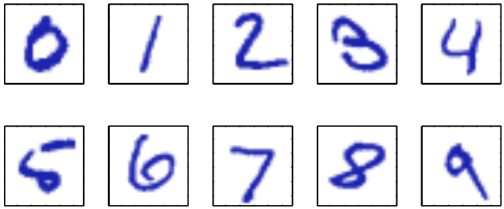
- un des champs d'étude de l'intelligence artificielle
- la discipline scientifique concernée par le développement, l'analyse et l'implémentation de méthodes automatisables qui permettent à une machine (au sens large) d'évoluer grâce à un processus d'apprentissage
- permet de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classiques

source : http ://sira-corp.com/new/MachineLearning

Exemple : Reconnaissance de caractère.

Comment reconnaître des caractères manuscrits ?

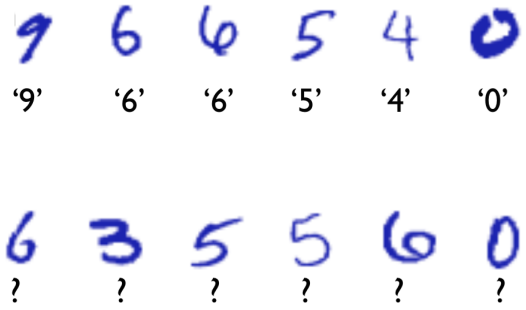
- par énumération de règles
  - si intensité pixel à la position ... alors c'est un "3"
  - long et fastidieux, diffiçile de couvrir tous les cas
- en demandant à la machine d'apprendre
  - lui laisser faire des essais et apprendre de ses erreurs
  - → apprentissage machine (machine-learning)



source : Hugo Larochelle

Comment ça marche ?

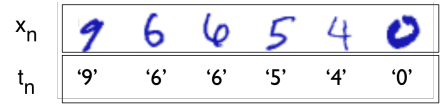
- On donne à l'algorithme des données d'**entrainement**
- l'algorithme d'apprentissage machine **apprend** un **modèle** capable de généraliser à de nouvelles données

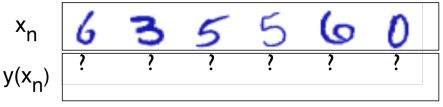


source : Hugo Larochelle

Notations

- On appel **ensemble d'entraînement** :
  - $\mathbb{D}_{train} = \{(x_1, t_1), \cdots, (x_N, t_N)\}$
    - x_n une **observation**
      - entrée du système
    - t_n la **cible** correspondante
      - sortie du système
- L'apprentissage machine fournit un **modèle** $y(x)$ qui prédit t en fonction de x :
  - $y(x_n) = \hat{t}$
- L'objectif est de trouver un modèle tel que $y(x_n) = \hat{t}_n \simeq t_n$
- On mesure la qualité de l'apprentissage (la qualité du modèle) sur un **ensemble de test** :
  - $\mathbb{D}_{test} = \{(x_{N+1}, t_{N+1}), \cdots, (x_{N+M}, t_{N+M})\}$





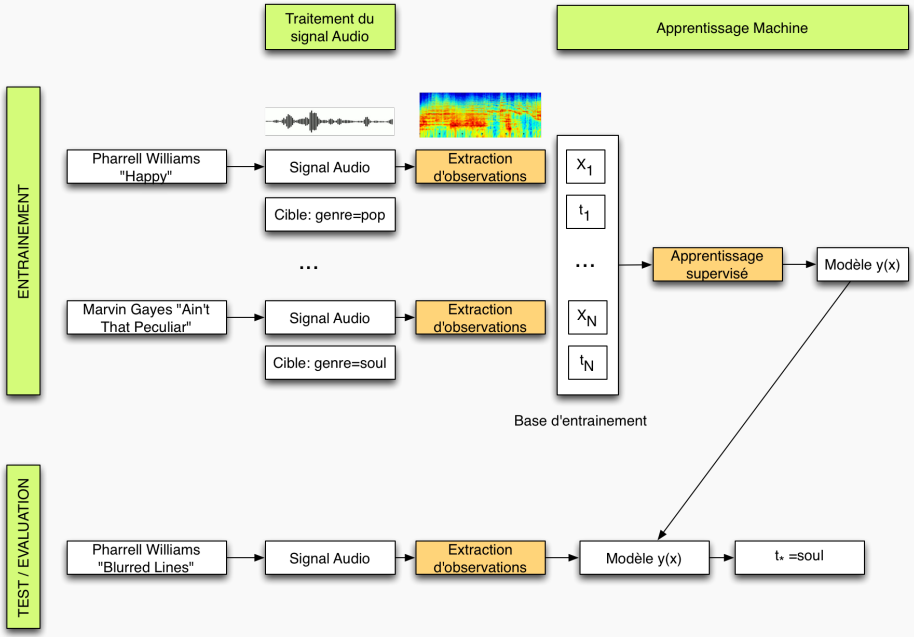
source : Hugo Larochelle

Apprentissage **supervisé**

Nous considérons un ensemble d'**observations** (entrées du système) $\{x_1, \cdots, x_N\}$

- Nous donnons également à la machine les **cibles** (sorties du système) souhaitées $\{t_1, \cdots, t_N\}$
- $\mathbb{D}_{train} = \{(x_1, t_1), \cdots, (x_N, t_N)\}$
- L'objectif de la machine est d'apprendre les cibles (sorties) correctes pour de nouvelles observations (entrées)

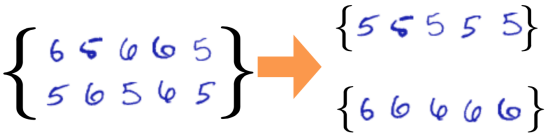
Exemple d'apprentissage supervisé en musique : reconnaissance du genre



Apprentissage **non-supervisé**

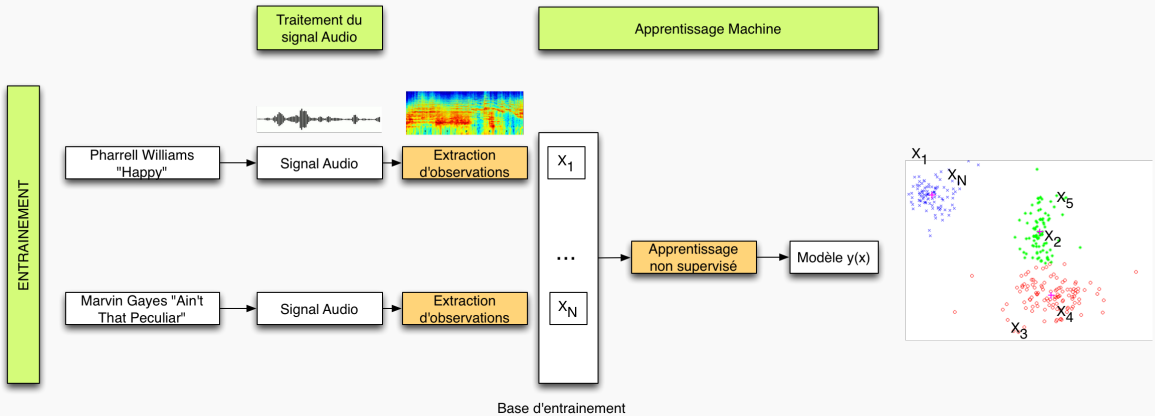
Nous considérons un ensemble d'**observations** (entrées du système) $\{x_1, \cdots, x_N\}$

- Nous ne donnons pas à la machine les cibles
- $\mathbb{D}_{train} = \{x_1, \cdots, x_N\}$
- L'objectif de la machine est de créer un modèle de x, un partitionnement (clustering) des données
  - Utilisation ? analyse de données, prise de décisions



source : Hugo Larochelle

Exemple d'apprentissage non-supervisé en musique : regroupement de morceaux par similarité de contenu



La **régression**

- La cible est un **nombre réel** : $t_n \in \mathbb{R}$
- Exemples :
  - Economie (prédiction de valeur en bourse) :
    - x= activité économique de la journée, t= la valeur d'une action demain
  - Audio (reconnaissance de tempo) :
    - x= le contenu spectral du signal, t= le tempo du morceau

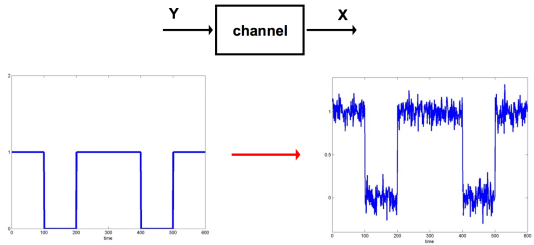
La **classification**

- La cible est un **indice de classe** : $t_n \in \{1, \cdots, C\}$
- Exemples :
  - Image (reconnaissance de caractères) :
    - x= vecteur d'intensité des pixels, t= l'identité du caractère
  - Audio (reconnaissance de parole) :
    - x : le contenu spectral du signal audio, t= le phonème prononcé

Système de communication.

Imaginons un système de communication dont l'entrée est Y et la sortie X.

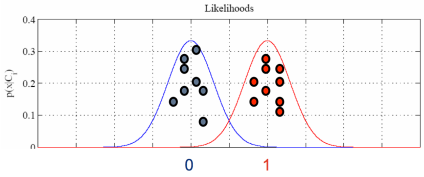
- on observe uniquement la sortie X
- on souhaite retrouver Y (non-observable) à partir de X (observable)
  - $\rightarrow$ on **infère** Y à partir de X



source : Arshia Cont

Solution 1 : Approche générative

- On **apprend** la fonction qui **génère**
  - les valeurs de X quand $Y = 0$: $P(X|Y = 0)$
  - les valeurs de X quand $Y = 1$: $P(X|Y = 1)$
- On en déduit les probabilités $P(Y = 0|X)$ et $P(Y = 1|X)$
- On décide que $Y = 0$ si $P(Y = 0|X) > P(Y = 1|X)$
- Ceci conduit à une **fonction de décision** $g(x)$
  - $g(x)$ est une conséquence des modèles génératifs



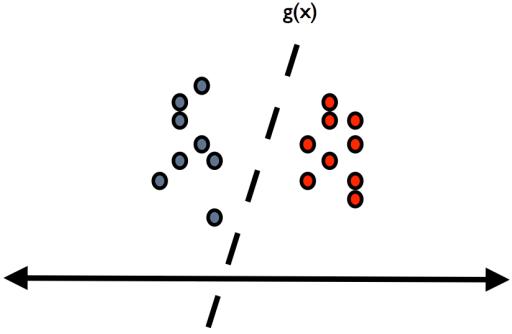
source : Arshia Cont

En résumé :

- nous partons de l'hypothèse qu'il existe une famille de modèles paramétriques permettant de générer X connaissant Y
  - Exemple : apprentissage Bayesien, modèle de Markov caché, réseaux de neurones artificiels

Solution 2 : Approche discriminante

- On apprend directement **la fonction de décision** $g(x)$qui sépare le mieux
  - les valeurs de X correspondant à $Y = 0$ et
  - les valeurs de X correspondant à $Y = 1$
- On ne considère pas la manière dont X est généré à partir de Y !!!



source : Arshia Cont

En résumé

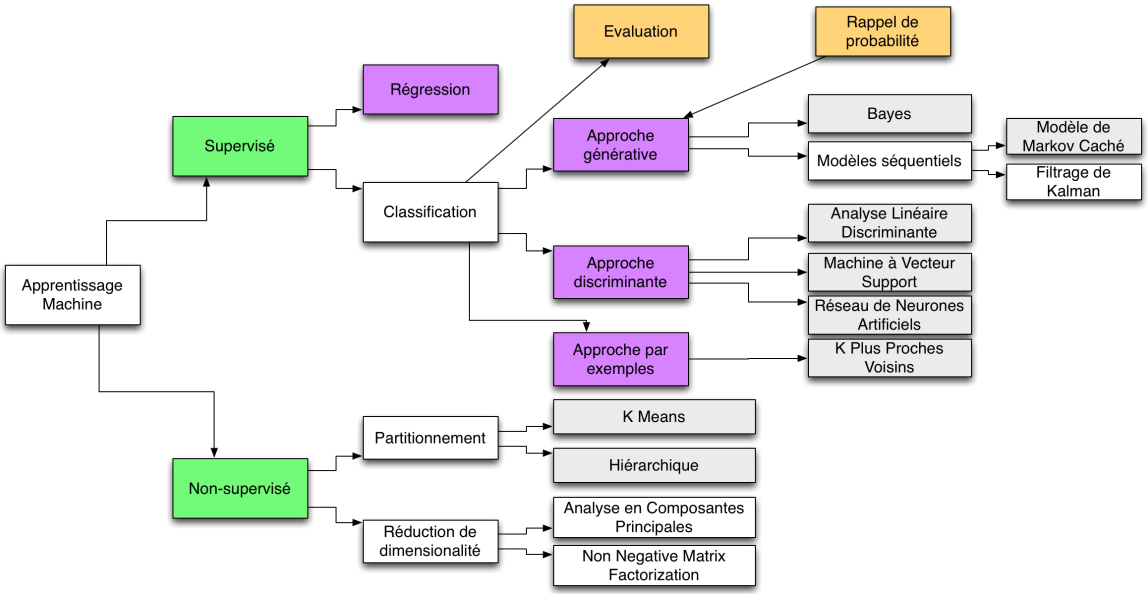
- nous n'avons pas d'hypothèse sur le modèle sous-jacent à X mais nous étudions comment séparer ses valeurs
  - Exemple : analyse linéaire discriminante, machine à vecteur support (SVM)

Solution 3 : Approche par exemplification

- On possède une série d'exemples de couples assignant une observation X à une cible Y : $\mathbb{D}_{train} = \{(x_1, y_1), \cdots, (x_N, t_N)\}$
  - pour une nouvelle observation $x^\star$, on cherche les observations X de la base d'entrainement les plus proches de $x^\star$,
  - on assigne à $x^\star$ le y correspondant aux X les plus proches
  - Exemple : K-plus-proche-voisin

Deux grands types d'apprentissage

- **Supervisé** → deux grandes cibles
  - **Régression**
  - **Classification** → trois grandes approches
    - Approche **générative** :
      - nous partons de l'hypothèse qu'il existe une famille de modèles paramétriques permettant de générer X connaissant Y
      - Exemple : apprentissage Bayesien, modèle de Markov caché
    - Approche **discriminante** :
      - nous n'avons pas d'hypothèse sur le modèle sous-jacent à X mais nous étudions comment les séparer
      - Exemple : analyse linéaire discriminante, machine à vecteur support (SVM), réseaux de neurones artificiels (ANN, Deep Learning)
    - Approche **par exemplification** :
      - K-plus-proche-voisin
- **Non-supervisé**

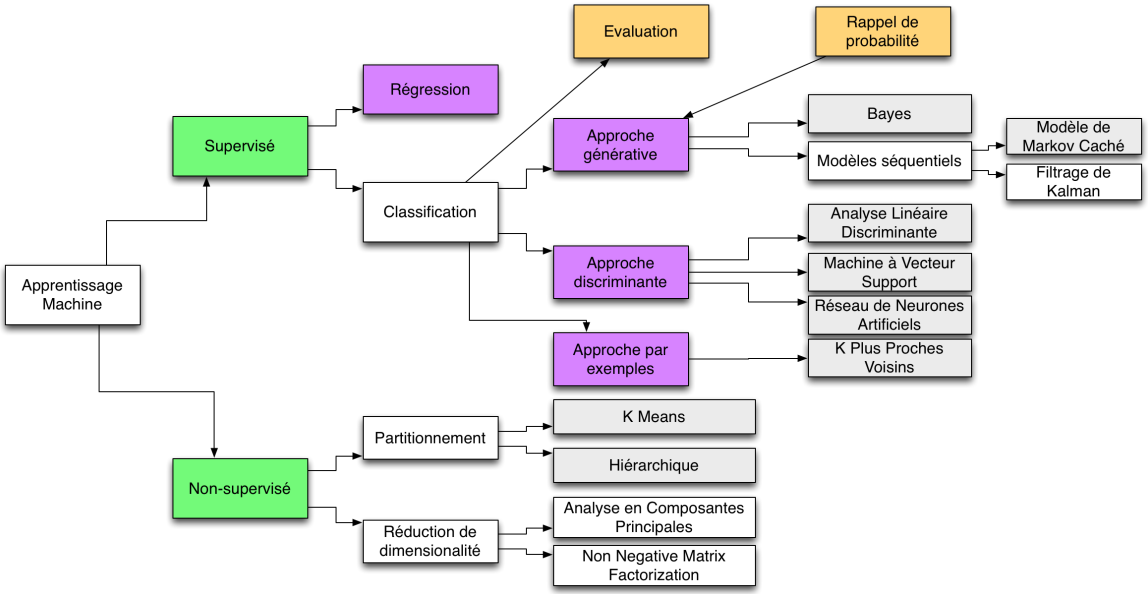

Apprentissage Machine
Supervisé
Régression
Classification
Evaluation
Rappel de probabilité
Approche générative
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Approche discriminante
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
Approche par exemples
K Plus Proches Voisins
Non-supervisé
Partitionnement
K Means
Hiérarchique
Réduction de dimensionalité
Analyse en Composantes Principales
Non Negative Matrix Factorization

Apprentissage

- Apprendre un modèle (génératif ou discriminant) à partir des observations X et des valeurs à prédire Y
- Le modèle doit permettre une bonne prédiction de Y en fonction des observations X

Généralisation

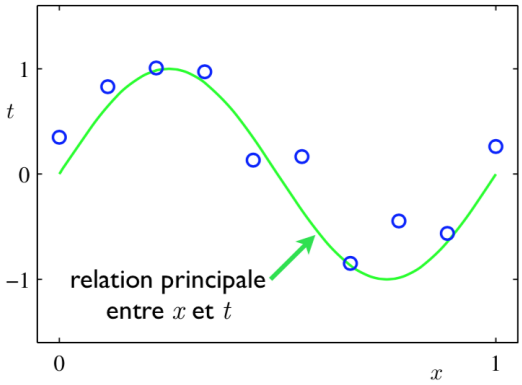
- Capacité du modèle à prédire correctement des valeurs $Y^\star$ en fonction de $X^\star$ en dehors de l'ensemble d'apprentissage
- Sur-apprentissage (over-fitting)
- En pratique on évalue les performances d'un modèle appris en séparant :
  - Ensemble d'entrainement (training-set) : $\{X, Y\}$
  - Ensemble de test (test-set) : $\{X^\star, Y^\star\}$


Apprentissage Machine
Supervisé
Non-supervisé
Régression
Classification
Partitionnement
Réduction de dimensionalité
Evaluation
Approche générative
Approche discriminante
Approche par exemples
Rappel de probabilité
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
K Plus Proches Voisins
K Means
Hiérarchique
Analyse en Composantes Principales
Non Negative Matrix Factorization

Régression

On cherche à prédire une cible t_n qui est un **nombre réel** : $t_n \in \mathbb{R}$

- Données :
  - entrée (observation) : x
  - sortie (cible) : $t \in \mathbb{R}$
- Objectif :
  - prédire t en fonction de x : $\hat{t} = y(x)$
  - y est notre modèle que nous devons apprendre à partir de l'ensemble d'entraînement $\mathbb{D}_{train} = \{(x_1, t_1), \cdots, (x_N, t_N)\}$

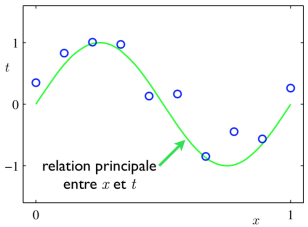


source : Hugo Larochelle

Forme du modèle y ?

- nous lui imposons une forme de polynôme d'ordre M de coefficients w_m

$$y(x, \boldsymbol{w}) = w_0 + w_1 x + ... + w_M x^M = \sum_{m=0}^{M} w_m x^m$$



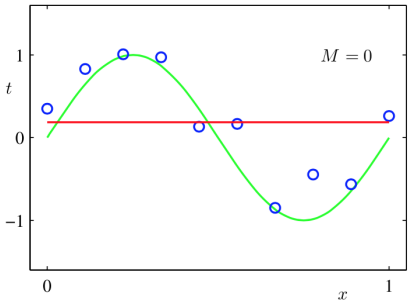
source : Hugo Larochelle

Estimation du modèle = estimation des coefficients w_m :

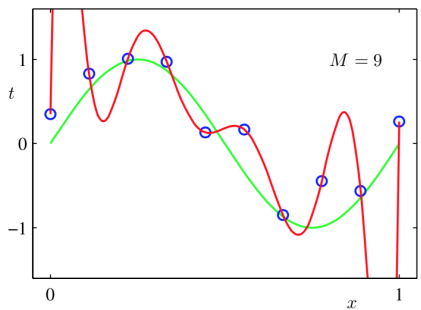
- on note $\boldsymbol{w}$ le vecteur des coefficients $[w_0, \cdots, w_M]$
- on cherche le vecteur $\boldsymbol{w}$ tel qu'il minimise l'erreur de prédiction sur l'ensemble d'entraînement $\mathbb{D}_{train}$
  - Erreur de prédiction sur une donnée t_n :
    - $\epsilon(\boldsymbol{w}, n) = (y(x_n, \boldsymbol{w}) - t_n)^2$
  - Erreur de prédiction totale (sur l'ensemble des données d'entraînement) :
    - $E(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^{N} \epsilon(\boldsymbol{w}, n)$

Comment choisir l'ordre du polynôme M ?

- si M est trop petit :
  - on modélise mal les données, grande perte sur l'ensemble d'entraînement
  - → **sous-apprentissage**
- si M est trop grand :
  - on apprend "par coeur" de l'ensemble d'entraînement
  - → **sur-apprentissage**

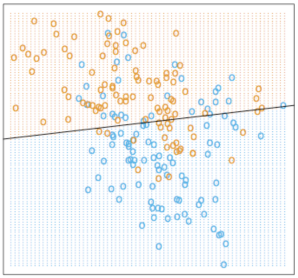


Sous-apprentissage (source : Hugo Larochelle)

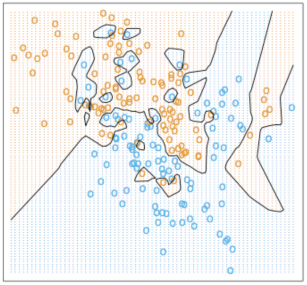


Dans le cas de la classification

- **Sous-apprentissage** :
  - grande perte sur l'ensemble d'entraînement
- **Sur-apprentissage** :
  - apprentissage "par coeur" de l'ensemble d'entraînement



Sous-apprentissage (source : Arshia Cont)



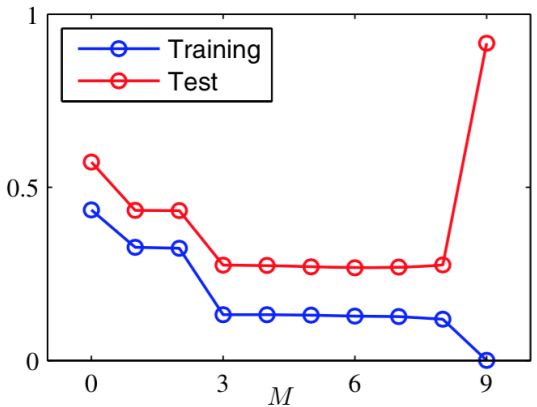
Sur-apprentissage (source : Arshia Cont)

Généralisation

- On cherche une valeur de M qui permet de retrouver la **tendance générale** de la relation entre x et t
  - sans apprendre le bruit
  - va permettre de généraliser à de nouvelles données

Capacité

- = Aptitude d'un modèle à apprendre "par coeur"
- plus M est grand,
  - plus le modèle a de capacité
- plus la capacité est grande,
  - plus la différence entre l'erreur d'entraînement et l'erreur de test augmente



source : Hugo Larochelle

Plus la quantité de données d'entraînement augmente, plus le modèle entraîné va bien généraliser

Régularisation

- Objectif :
  - utiliser un grand M avec peu de données
- Régularisation : on pénalise la somme des carrés des paramètres w_m
  - $E(\boldsymbol{w}) = \sum_{n=1}^{N}(y(x_n, \boldsymbol{w}) - t_n)^2 + \frac{\lambda}{2}||\boldsymbol{w}||^2$
    - dans lequel $||\boldsymbol{w}||^2 = w_0^2 + w_1^2 + \cdots + w_M^2$

Sélection de modèle

- Choix de M et choix de λ ?
- M et λ sont appelés des **hyper-paramètres**
- Comment déterminer les hyper-paramètres ? **sélection de modèles**
  - on rajoute une nouvelle base
    - base d'entraînement : $\mathbb{D}_{train}$
    - **base de validation** : $\mathbb{D}_{validation}$
    - base de test : $\mathbb{D}_{test}$

Régression polynomiale en D dimension

- D dimensions ?
  - le nombre de dimension de l'observation x
  - exemple :
    - $D = 1$: on considère l'intensité globale d'une image
    - $D = 100$: on considère l'intensité de chaque pixel d'une image de (10,10)
- Pour un polynôme d'ordre $M = 3$
  - on a $1 + D + D^2 + D^3$ paramètres à estimer
- pour $D = 1$ **dimensions** ($x = x_i$), on a 4 paramètres à estimer

$$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

- pour $D = 3$ **dimensions** ($\boldsymbol{x} = [x_i, x_j, x_k]$), on a 40 paramètres à estimer

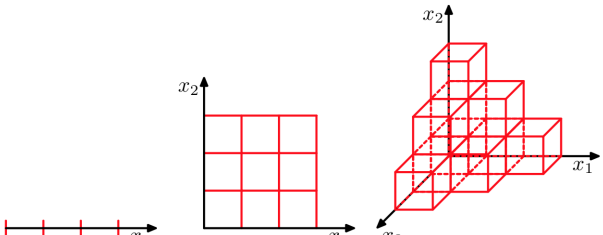
$$y(\boldsymbol{x}, \boldsymbol{w}) = w_0 + \sum_{i=1}^{D} w_i x_i + \sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij} x_i x_j + \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{k=1}^{D} w_{ijk} x_i x_j x_k$$

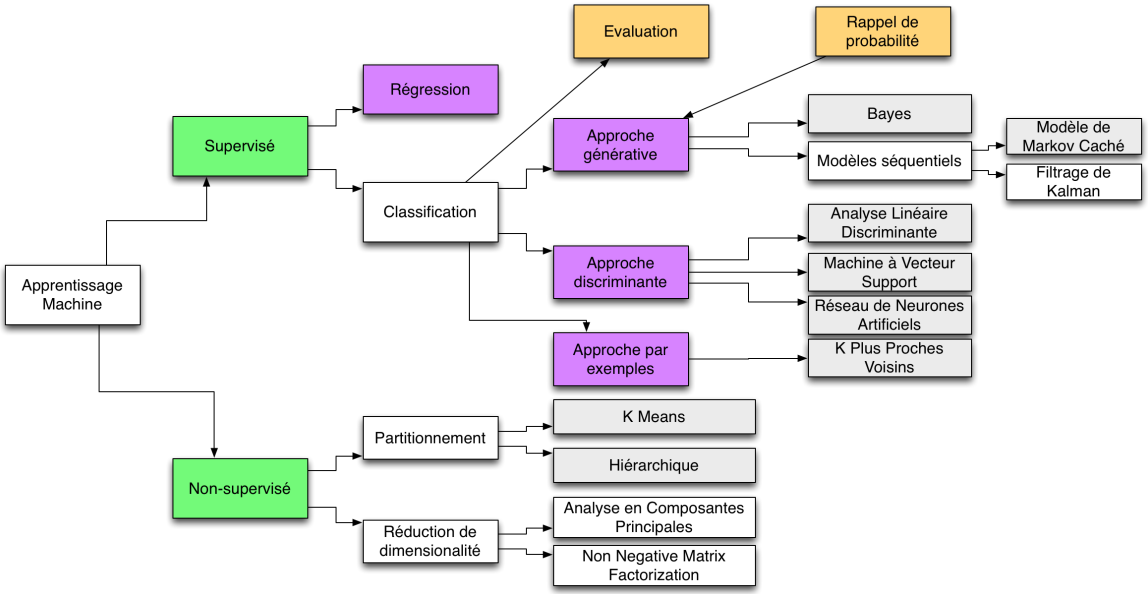
- pour D=100, on a 1.010.101 paramètres à estimer !!!

Malédiction de la dimension

- Pour un ordre $M = 3$ et D=100 dimensions, on a 1.010.101 paramètres à estimer!!!

- Pour pouvoir garantir qu'on va bien généraliser à une nouvelle entrée x, il faut avoir des entrées similaires à x dans l'ensemble d'entraînement
  - Au plus le nombre de dimension D augmente, au plus il devient difficile d'avoir des entrées similaires à x
- Preuve (interprétation géométrique) :
  - on divise également l'espace des observations en régions (hypercubes)
  - quand D augment, le nombre de régions augmente en $O(3^D)$
    - Il devient impossible de garantir qu'on aura bien un exemple dans chaque région!!!



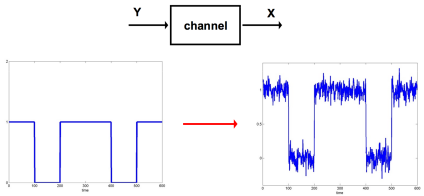

Apprentissage Machine
Supervisé
Régression
Classification
Evaluation
Rappel de probabilité
Approche générative
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Approche discriminante
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
Approche par exemples
K Plus Proches Voisins
Non-supervisé
Partitionnement
K Means
Hiérarchique
Réduction de dimensionalité
Analyse en Composantes Principales
Non Negative Matrix Factorization

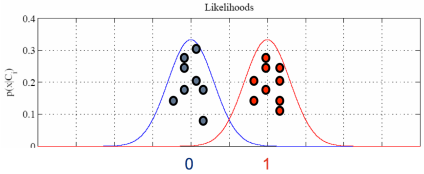
Approche générative

→ rappel probabilité

Décision Bayesienne

- On souhaite trouver la classe $Y = \{0, 1\}$ en fonction de x
- En l'absence d'information, les deux probabilités sont équiprobable :
  - $p(Y = 0) = p(Y = 1) = 0.5$
  - $p(Y)$ est la **probabilité a priori** (prior)
- On considère que X a été généré (modèle génératif) par Y : $p(X|Y)$
  - plus précisément on considère que X est une version bruitée de Y
    - $X = Y + \epsilon$
    - ou ϵ est un bruit de moyenne nul et de variance σ^2 : $\epsilon \sim \mathcal{N}(0, \sigma)$
  - on peut donc modéliser X comme
    - $P(X|Y = 0) \sim \mathcal{N}(0, \sigma)$
    - $P(X|Y = 1) \sim \mathcal{N}(1, \sigma)$
  - $P(X|Y)$ est la **vraisemblance** (likelihood)

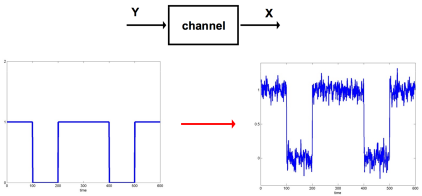


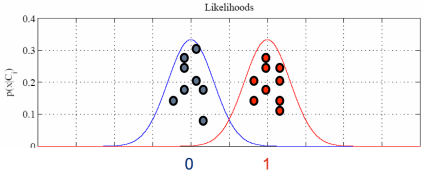


source : A. Cont

Comment choisir la meilleure classe $Y = 0$ ou 1 ?

- Méthode du **Maximum A Posteriori** (MAP)
  - on choisi la classe dont la probabilité a posteriori est maximale
  - $i^\star(x) = \arg\max_i \{p(Y = y_i|X)\}$
- Problème :
  - on ne connait pas $p(Y = y_i|x)$, mais on connait $P(X|Y = y_i)$
  - comment passer de l'un à l'autre → inférence Bayesienne





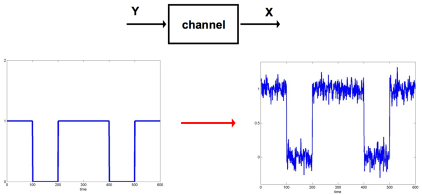
source : A. Cont

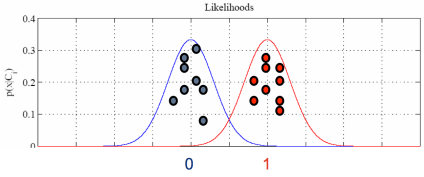
Inférence Bayesienne

- Permet de mettre à jour les informations à **priori** pour créer les informations à **posteriori** en fonction des informations que nous avons sur X (**vraisemblance**)
- Rappel :
  - $P(X,Y) = P(Y|X)P(X) = P(X|Y)P(Y)$
- Inférence Bayesienne :

$$p(Y = y_i|x) = p(Y = y_j) \cdot \frac{p(x|Y = y_i)}{p(x)}$$

$$\textbf{posterior} = \textbf{prior} \cdot \frac{\textbf{vraisemblance}}{\textbf{evidence}}$$



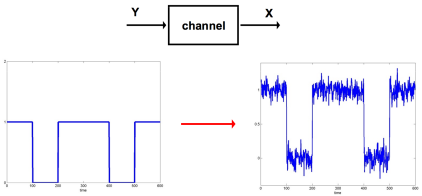


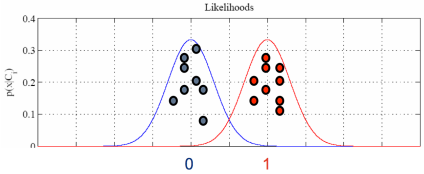
source : A. Cont

Comment choisir la meilleure classe $Y = 0$ ou 1 ?

- Méthode du **Maximum A Posteriori** (MAP) :
  - si on omet le dénominateur $p(x)$ (identique pour toutes les classes)
  - $i^\star(x) = \arg\max_i \{p(Y = y_i)p(x|Y = y_i)\}$
- Pour le calcul on considère les log-probabilités

$$i^\star(x) = \arg\max_i \{\log(p(Y = y_i)) + \log(p(x|Y = y_i))\}$$

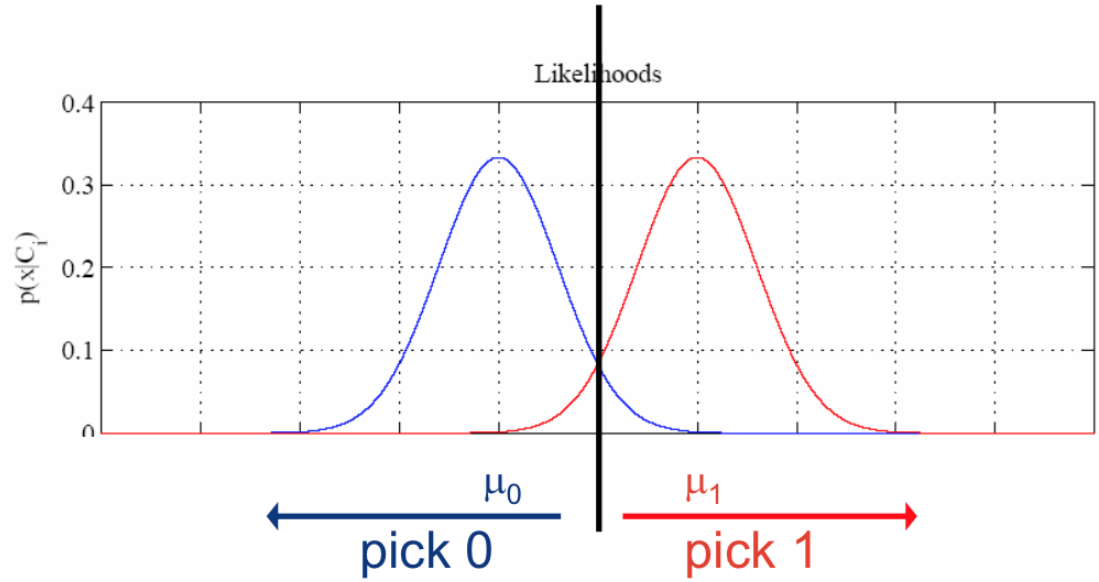




source : A. Cont

Inférence Bayesienne dans le cas Gaussien : Cas simple

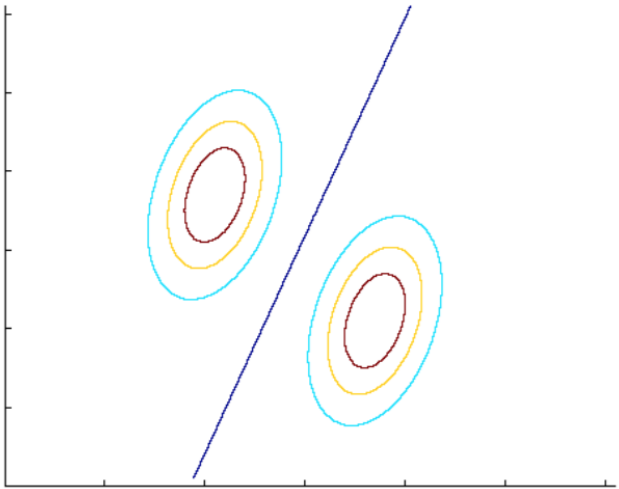
- Cas Simple ?
  - on considère toutes les classes **a priori** equi-probables :
    - $P(Y=0) = P(Y=1) = 0.5$
    - alors $i^\star(x) = \arg\max_i \{\log(p(x|Y=y_i))\}$
  - on considère $D=1$ et les variances égales ($\sigma = \sigma_0 = \sigma_1$) alors
    - $P(X|Y=0) \sim \mathcal{N}(\mu_0, \sigma)$
    - $P(X|Y=1) \sim \mathcal{N}(\mu_1, \sigma)$
- On peut montrer que
  - $i^\star(x) = \arg\min_i (w_i x + w_{i0})$
    - avec $w_i = -2\mu_i$
    - avec $w_{i0} = \mu_i^2$
- On peut montrer que les classes i et j sont **équi-probables** pour x tel que
  - $-2\mu_0 x + \mu_0^2 = -2\mu_1 x + \mu_1^2$
- Ces valeurs de x définissent une **frontière de décision** $g(x)$
  - $g(x) = \frac{\mu_1 + \mu 2}{2}$



source : Arshia Cont

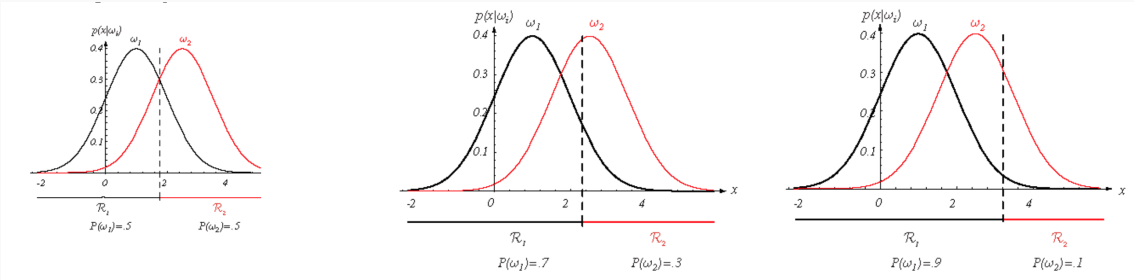
Inférence Bayesienne dans le cas Gaussien : Cas général

- Cas Général ?
  - on considère les classes **a priori** non equi-probables :
    - $P(Y=0) \neq P(Y=1)$
  - on considère $D>1$ et les matrices de co-variances égales ($\boldsymbol{\Sigma}=\boldsymbol{\Sigma}_0=\boldsymbol{\Sigma}_1$) alors
    - $P(X|Y=0) \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma})$ et
    - $P(X|Y=1) \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma})$
- On peut montrer que
  - $i^\star(x) = \arg\min_i \{\boldsymbol{w}_i^T \boldsymbol{x} + \omega_{i0}\}$
    - avec $\boldsymbol{w}_i = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i$
    - avec $\omega_{i0} = -\frac{1}{2}\boldsymbol{\mu}_i^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i + \log(P(Y=i))$
- On peut montrer que les classes i et j sont **equi-probables** pour $\boldsymbol{x}$
  - $\boldsymbol{w}_i^T\boldsymbol{x} + \omega_{i0} = \boldsymbol{w}_j^T\boldsymbol{x} + \omega_{j0}$
- Ces valeurs de $\boldsymbol{x}$ définissent la **frontière de décision**
  - $\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$
    - avec $\boldsymbol{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)$
    - avec $w_0 = -\frac{(\boldsymbol{\mu}_i+\boldsymbol{\mu}_j)^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_i-\boldsymbol{\mu}_j)}{2} + \log\frac{P(Y=i)}{P(Y=j)}$



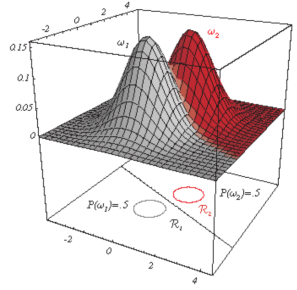
source : Arshia Cont

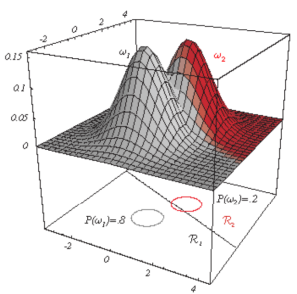
Illustration de l'influence de la probabilité a priori quand $D = 1$

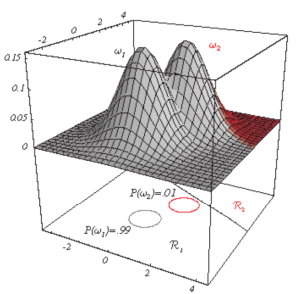


source : Arshia Cont

Illustration de l'influence de la probabilité a priori quand $D = 2$

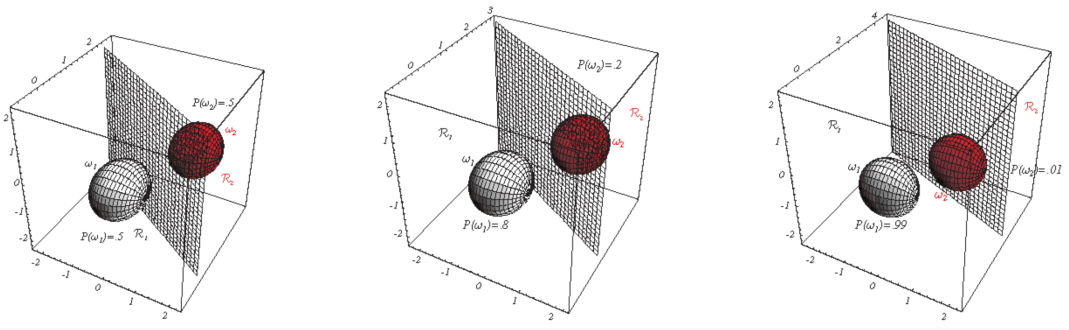






source : Arshia Cont

Illustration de l'influence de la probabilité a priori quand $D = 3$



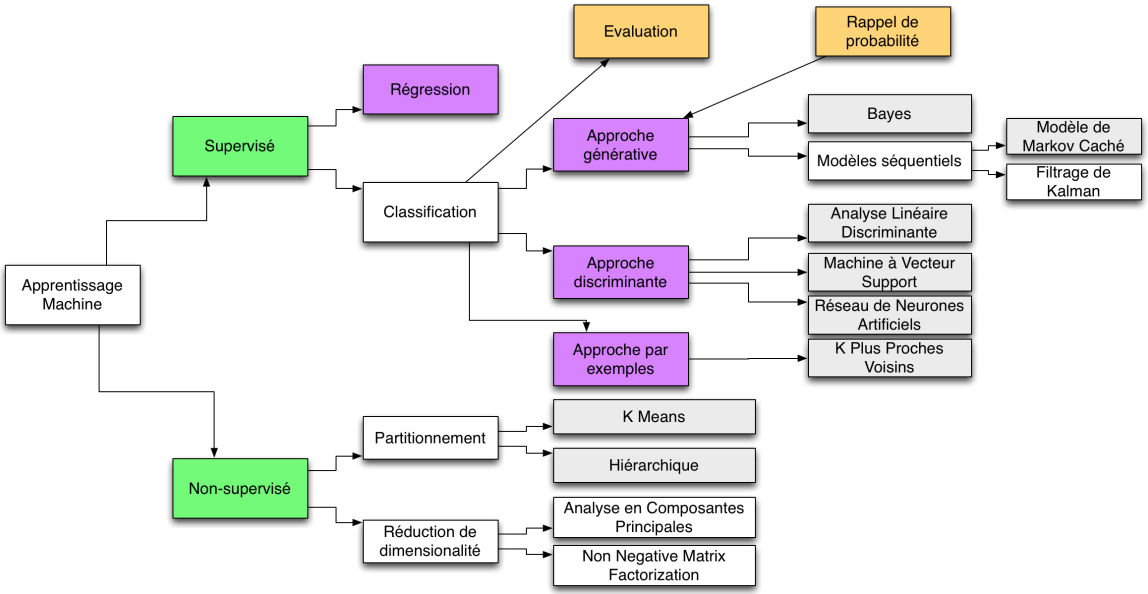
source : Arshia Cont

Conclusion

- Dans l'approche générative, nous apprenons les modèles $P(X|Y)$ ayant engendré X en fonction de Y
- Nous utilisons l'information a priori sur les classes Y : $P(Y)$
- Nous combinons les deux pour obtenir une probabilité a posteriori $P(Y|X)$ en utilisant la loi de Bayes
- La conséquence de cette modélisation est une frontière de décision entre classes $\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$

Approche discriminante

- Nous apprenons directement la frontière $\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$ qui permet le mieux de séparer les classes Y
- Nous ne faisons pas d'hypothèse sur le modèle ayant engendré X


Apprentissage Machine
Supervisé
Non-supervisé
Régression
Classification
Partitionnement
Réduction de dimensionalité
Evaluation
Approche générative
Approche discriminante
Approche par exemples
Rappel de probabilité
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
K Plus Proches Voisins
K Means
Hiérarchique
Analyse en Composantes Principales
Non Negative Matrix Factorization

Approche générative- Modèles séquentiels

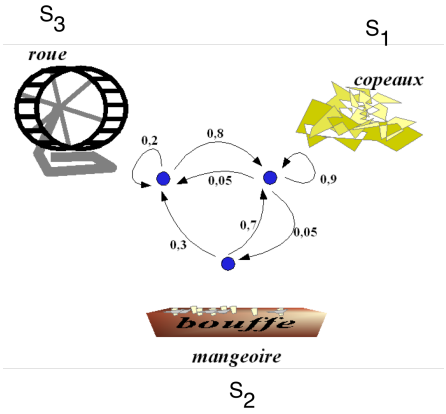
Modèle de Markov

- Andreï A. Markov (1856-1922) : un mathématicien Russe
- Chaîne de Markov :
  - un processus stochastique à **temps discret** t pouvant être dans des **états discrets** $S \in [1, \cdots, I]$
  - S_t la valeur de l'état à l'instant t
- Chaîne de Markov d'ordre 1 :
  - la prédiction de l'état actuel ne dépend que de l'instant précédent.
    - $p(S_t|S_{t-1}, S_{t-2} \cdots S_0) = p(S_t|S_{t-1})$

Exemple : Doudou le hamster

- Doudou le hamster à 3 états dans sa journée :
  - il dort :
    - il est dans l'état S_1 (copeaux)
  - il mange :
    - il est dans l'état S_2 (mangeoire)
  - il fait du sport :
    - il est dans l'état S_3 (roue)
- On peut représenter la succession de ces états par une matrice de transition entre états

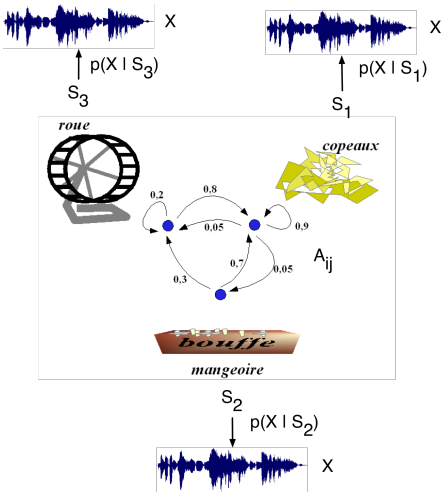
$$T_{ij} = P(S_{t+1} = j, S_t = i) = \begin{pmatrix} 0.9 & 0.05 & 0.05 \\ 0.7 & 0 & 0.3 \\ 0.8 & 0 & 0.2 \end{pmatrix}$$



Modèle de Markov d'une journée de Doudou le hamster

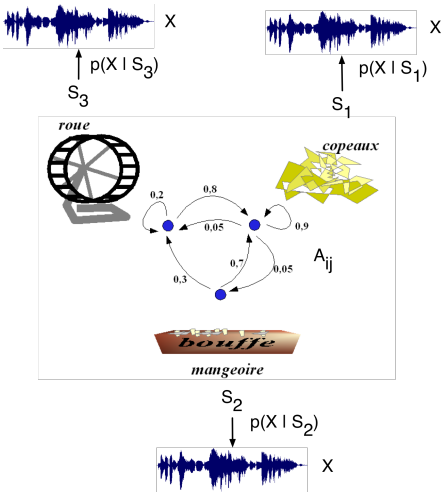
Modèle de Markov **caché** ?

- Dans un modèle de Marlov caché on observe pas directement les états S_i, ils sont "cachés"
- on observe une une émission de ces états S_i
  - exemple : on observe le bruit X que fait Doudou le hamster
- Pour chaque état, nous pouvons cependant définir la
  - probabilité d'émission de X par S_i : $p(X|S_i)$



Définition d'un modèle de Markov caché

- Définition des **états** S_i
- Définition des **observations** X
- **Probabilités d'émission** de X par l'état S_i :
  - $A_i(X) = P(X_t = X | S_t = i)$
- **Probabilités de transiter** de S_i au temps t vers S_j au temps $t+1$:
  - $T_{ij} = P(S_{t+1} = j | S_t = i)$
- La **probabilité qu'initialement** $(t = 0)$ le modèle se trouve dans l'état S_j :
  - $\pi_j = P(s_1 = j)$
- On note $\{\lambda\}$ l'ensemble de ces éléments

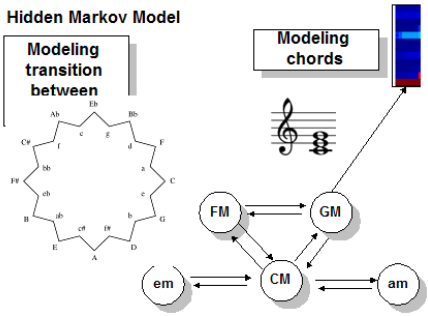


Le modèle de Markov caché permet de résoudre les trois problèmes suivants :

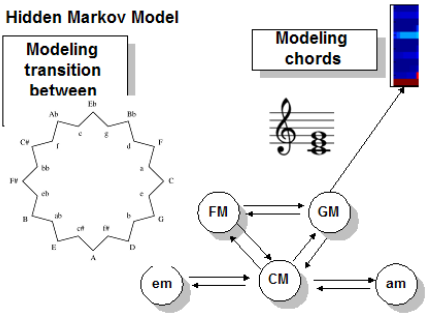
- **Décodage de la séquence d'états** :
  - Etant donné une suite d'observation X_t et un modèle $\{\lambda\}$, quelle est la suite d'état S_i correspondant : $S^\star = \arg\max_S P(S|X,\lambda)$
    - Exemple : si on observe la séquence de son X de Doudou et étant donné son modèle dormir/manger/exercice $\{\lambda\}$, quel est la séquence d'activités S de Doudou ?
- **Evaluation** :
  - Etant donné une suite d'observation X et un modèle $\{\lambda\}$, quelle est la probabilité que ce modèle ait généré X : $P(X|\{\lambda\})$
    - Exemple : comment déterminer si une séquence d'observation du son X correspond au modèle $\{\lambda_1\}$ dormir/manger/exercice de Doudou le hamster, ou au modèle $\{\lambda_2\}$ dormir/manger/travailler de Bill le salarié
- **Entrainement** :
  - Etant donné une/des suites d'observations $\{X_t\}$, trouver le modèle $\lambda^\star$ qui maximize la vraisemblance des observations : $\lambda^\star = \arg\max_\lambda P(X|\lambda)$
    - Exemple : comment déterminer les paramètres $\{\lambda\}$ du modèle de Doudou le hamster ?

Exemple : La reconnaissance d'accords

- **Objectif**
  - On veut estimer la suite d'accords $\{C_M, C\#_M, \cdots, C_m, \cdots\}$ d'un morceau de musique à partir de l'observation de son signal audio
- Définition du modèle
  - **Etats** $S_i=$
    - les différents accords $\{C_M, C\#_M, \cdots, C_m, \cdots\}$
  - **Observations** $X=$
    - un descripteur audio appelé Chroma ou Pitch Class Profile
  - **Probabilités d'émission** de X par l'état $S_i=$
    - probabilité d'observer ces hauteurs de chroma pour cet accord



- **Probabilités de transiter** de S_i au temps t vers S_j au temps $t+1$:
  - la matrice de transition entre accords respecte la théorie musicale
  - certains accords s'enchainent mieux (G_M vers C_M = consonance) que d'autres (G_M vers $C\#_M$ = dissonance), cercle des quintes, relatifs majeur-mineur, ...
- **Solution**
  - On estime la suite d'accords par **décodage** d'un modèle de Markov Caché



Exemple : La reconnaissance de parole(1)

Un système de reconnaissance de parole est composé de quatre grandes parties :

- 1) **Modèle de langage**
- 2) **Phonétiseur**
- 3) **Pré traitement audio**
- 4) **Modèle acoustique**

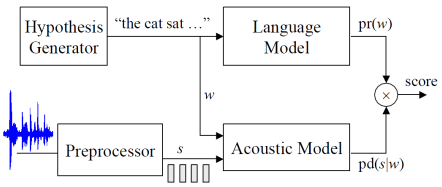
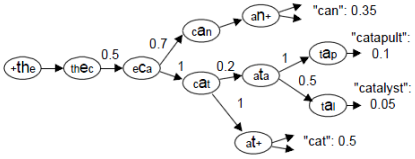


Schéma général d'un système de reconnaissance de parole (source : Mike Brookes)

Exemple : La reconnaissance de parole(2)

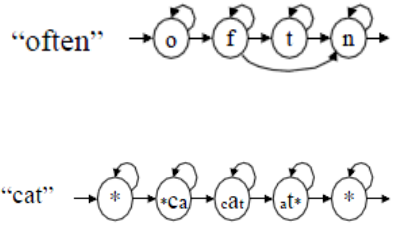
- 1) **Modèle de langage** :
  - Représente la probabilité d'une séquence de mots (dépend du vocabulaire et de la grammaire d'une langue, indépendant du signal audio)



Modèle de langage (source : Mike Brookes)

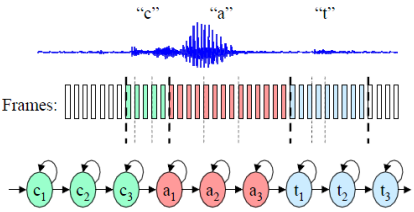
Exemple : La reconnaissance de parole(3)

- 2) **Phonétiseur** :
  - Transforme les mots en séquence de
    - **phonèmes**
      - Phonème : plus petite unité distinctive que l'on puisse isoler : cote (/k ?t/) et côte (/kot/)
      - Pour une même langue, selon l'accent ,il existe plusieurs prononciations d'un même mot (petit, p'tit) donc plusieurs suite de phonèmes possibles
      - 37 phonèmes en français
    - **tri-phones**
      - 37^3 tri-phone en français



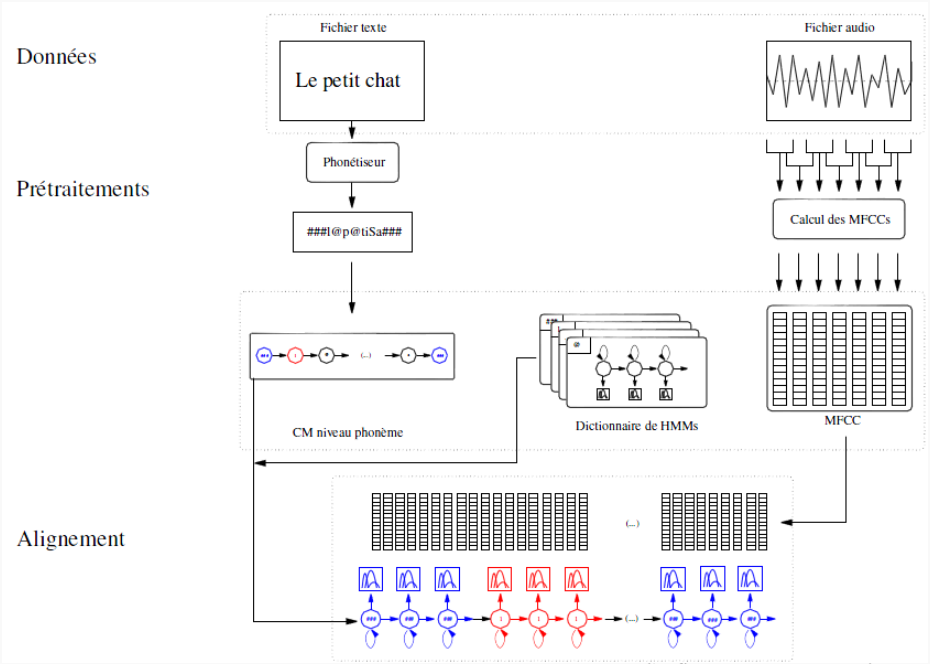
Exemple : La reconnaissance de parole(4)

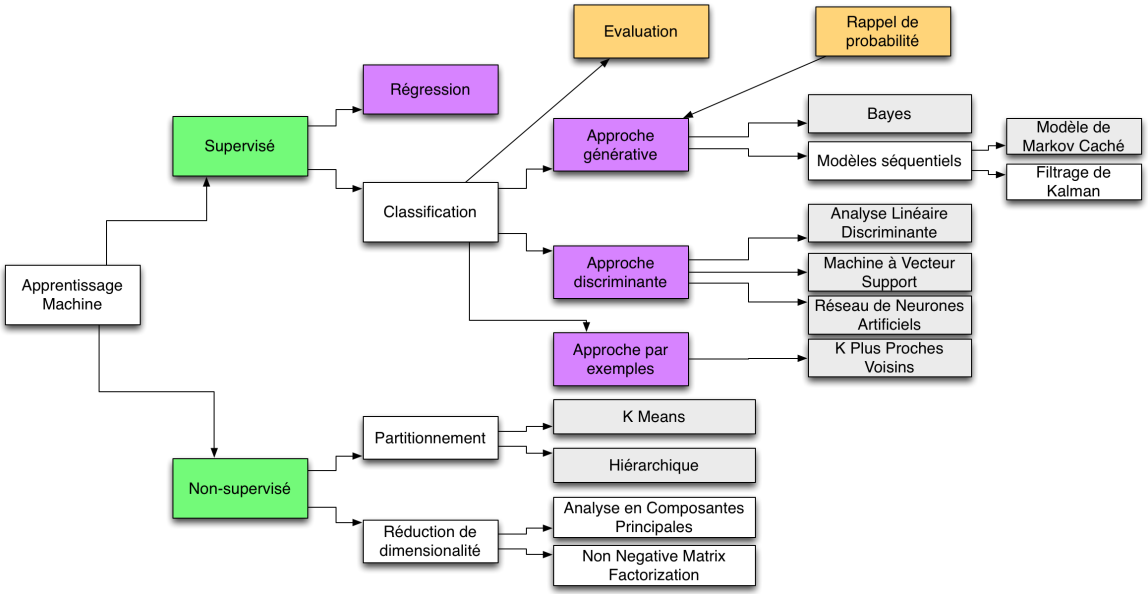
- 3) **Pré traitement audio** :
  - extrait des observations X pertinentes du signal audio
    - Généralement : MFCC + Δ MFCC + $\Delta\Delta$ MFCC (L=25ms, 40 bandes, 3 $\star$ 39 coefficients)
- 4) **Modèle acoustique** :
  - définit un modèle de Markov caché permettant la jonction entre un phonème / tri-phone et les différentes occurrences acoustiques (différents locuteurs)



Représentation d'un mot en séquence de phonème et représentation acoustique des phonèmes (source : Mike Brookes)

Exemple : Alignement de parole à un texte

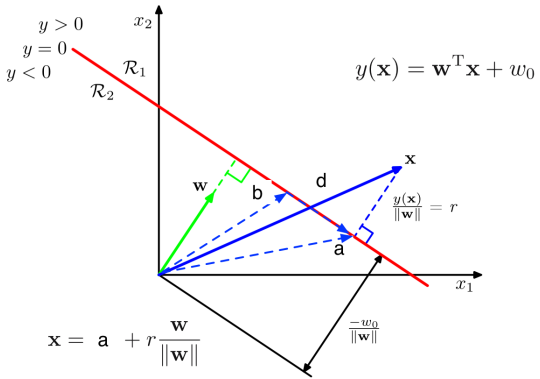



Apprentissage Machine
Supervisé
Non-supervisé
Régression
Classification
Partitionnement
Réduction de dimensionalité
Evaluation
Approche générative
Approche discriminante
Approche par exemples
Rappel de probabilité
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
K Plus Proches Voisins
K Means
Hiérarchique
Analyse en Composantes Principales
Non Negative Matrix Factorization

Approche discriminante

Frontière/surface de décision

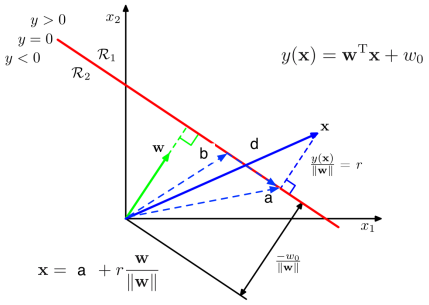
- La conséquence de l'approche générative est une frontière de décision séparant les deux classes
- Nous l'appelons $y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$
- $y(\boldsymbol{x})$ divise l'espace en deux sous-espaces
  - $y(\boldsymbol{x}) = 0$: points sur la frontière
  - $y(\boldsymbol{x}) > 0$: points dans la direction de $\boldsymbol{w}$
  - $y(\boldsymbol{x}) < 0$: points dans la direction opposée de $\boldsymbol{w}$
- Distance du point $\boldsymbol{x}$ à la frontière : $\frac{y(x)}{||\boldsymbol{w}||}$
- Pour la classification, choix de
  - la classe 0 si $y(x) < 0$
  - la classe 1 si $y(x) > 0$



source : Hugo Larochelle

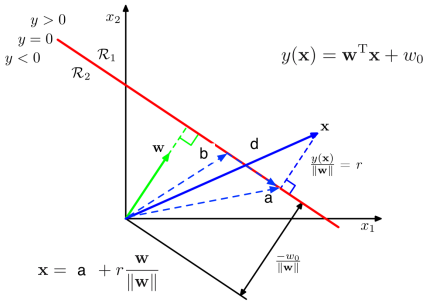
Frontière/surface de décision

- Equation : $y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$
- Surface de décision : ligne rouge
- La surface de décision est $\perp$ à $\boldsymbol{w}$
- **Preuve** :
  - $\boldsymbol{a}$ est sur la frontière donc $y(\boldsymbol{a}) = 0$
  - on décompose $\boldsymbol{a}$ comme $\boldsymbol{a} = \boldsymbol{b} + \boldsymbol{d}$
  - $y(\boldsymbol{a}) = 0$
    - $= y(\boldsymbol{b} + \boldsymbol{d})$
    - $= \boldsymbol{w}^T(\boldsymbol{b} + \boldsymbol{d}) + w_0$
    - $= \boldsymbol{w}^T\boldsymbol{b} + w_0 + \boldsymbol{w}^T\boldsymbol{d}$
    - $= y(\boldsymbol{b}) + \boldsymbol{w}^T\boldsymbol{d}$,
    - $\boldsymbol{b}$ est sur la frontière donc $y(\boldsymbol{b}) = 0$
    - $= \boldsymbol{w}^T\boldsymbol{d}$
    - donc $\boldsymbol{w}$ est $\perp$ à $\boldsymbol{d}$ qui lui est sur la frontière



Frontière/surface de décision

- Equation : $y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$
- La distance entre $\boldsymbol{x}$ et la frontière de décision est appelée r
- r est égale à $r = \frac{y(\boldsymbol{x})}{||\boldsymbol{w}||}$
- **Preuve** :
  - on décompose $\boldsymbol{x}$ comme $\boldsymbol{x} = \boldsymbol{a} + r\frac{\boldsymbol{w}}{||\boldsymbol{w}||}$
  - $y(\boldsymbol{x})$
    - $= y(\boldsymbol{a} + r\frac{\boldsymbol{w}}{||\boldsymbol{w}||})$
    - $= \boldsymbol{w}^T(\boldsymbol{a} + r\frac{\boldsymbol{w}}{||\boldsymbol{w}||}) + w_0$
    - $= \boldsymbol{w}^T\boldsymbol{a} + w_0 + r\frac{\boldsymbol{w}^T\boldsymbol{w}}{||\boldsymbol{w}||}$
    - $= y(\boldsymbol{a}) + r||\boldsymbol{w}||$
    - $\boldsymbol{a}$ est sur la frontière donc $y(\boldsymbol{a}) = 0$
    - $= r||\boldsymbol{w}||$

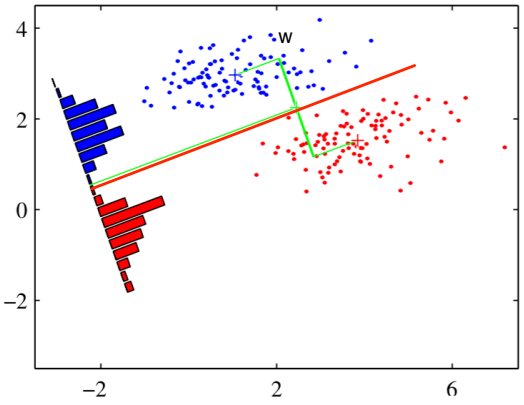


Comment choisir cette frontière pour séparer le mieux les classes ?

- Analyse Linéaire Discriminante (ALD)
- Machine à Vecteur Support (SVM)

Analyse Linéaire Discriminante (ALD)

- $y(\boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{x}$ est le résultat de la projection de $\boldsymbol{x}$ sur l'hyper-plan $\boldsymbol{w}$
- l'ALD cherche la projection qui
  - **maximise la séparation des moyennes** m_i des données projetées
    - $m_i = \frac{1}{N_i}\sum_{n\in C_i}\boldsymbol{w}^T\boldsymbol{x}_n$
  - **minimise les variances intra-classes** s_i^2 des entrées projetées
    - $s_i^2 = \sum_{n\in C_i}(\boldsymbol{w}^T\boldsymbol{x}_n - m_i)^2$



source : Hugo Larochelle

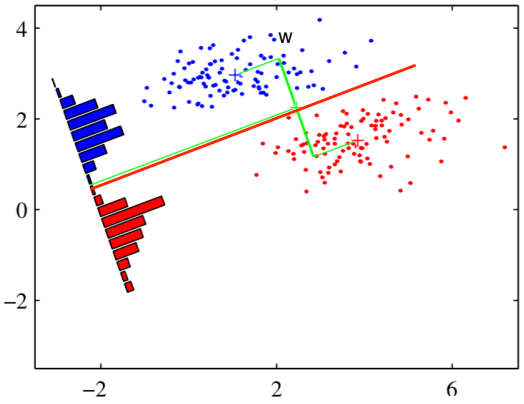
Analyse Linéaire Discriminante (ALD)

- Au total on maximise

$$J(\boldsymbol{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \frac{\textbf{variance inter-classes}}{\textbf{variance intra-classe}}$$

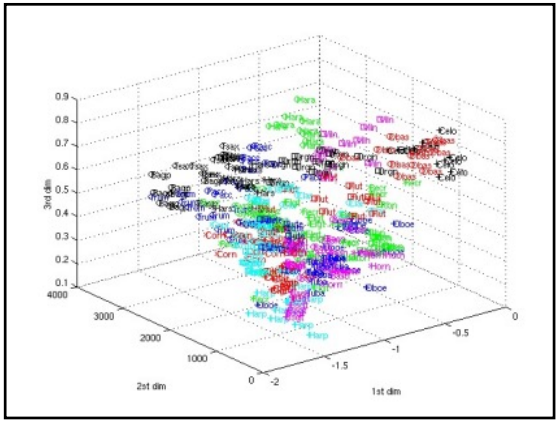
- La solution est $\boldsymbol{w} \propto S_w^{-1}(m_1 - m_2)$
- avec S_w la matrice de co-variance intra-classe

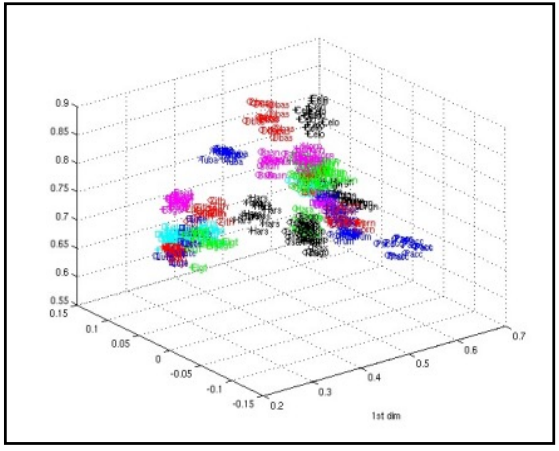
$$S_w = \sum_{n \in C_i} (x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_i} (x_n - m_2)(x_n - m_2)^T$$

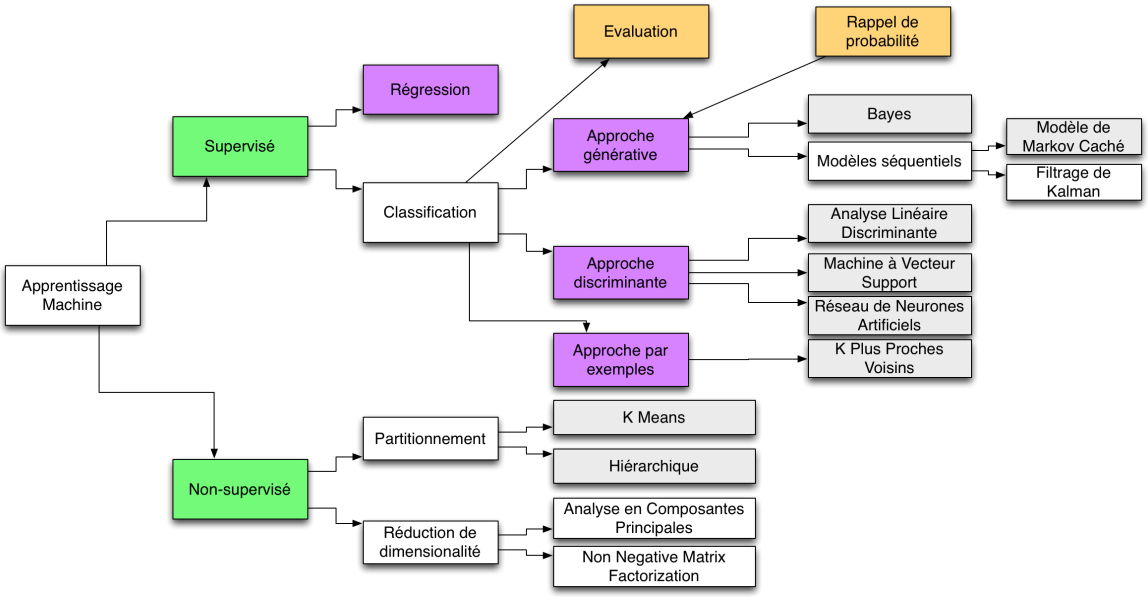


source : Hugo Larochelle

Exemple : application de l'ALD pour la reconnaissance des instruments

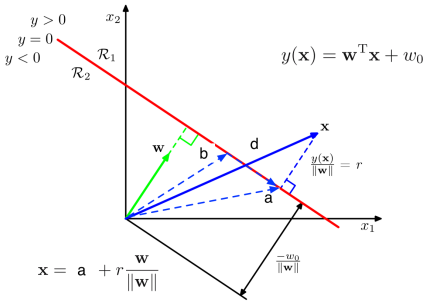





Apprentissage Machine
Supervisé
Non-supervisé
Régression
Classification
Partitionnement
Réduction de dimensionalité
Evaluation
Approche générative
Approche discriminante
Approche par exemples
Rappel de probabilité
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
K Plus Proches Voisins
K Means
Hiérarchique
Analyse en Composantes Principales
Non Negative Matrix Factorization

Machine à Vecteurs Supports (SVM)

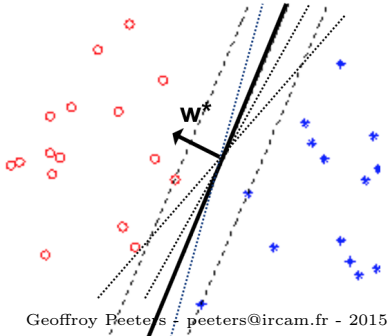
- Rappel :
  - La distance du point $\boldsymbol{x}$ à la frontière est $\frac{y(\boldsymbol{x})}{||\boldsymbol{w}||}$
- On appel **marge** λ la distance du point le plus proche à la frontière
  - $\lambda = \min_i \frac{|\boldsymbol{w}^T \boldsymbol{x}_i + w_0|}{||\boldsymbol{w}||}$



source : Ugo Larochelle

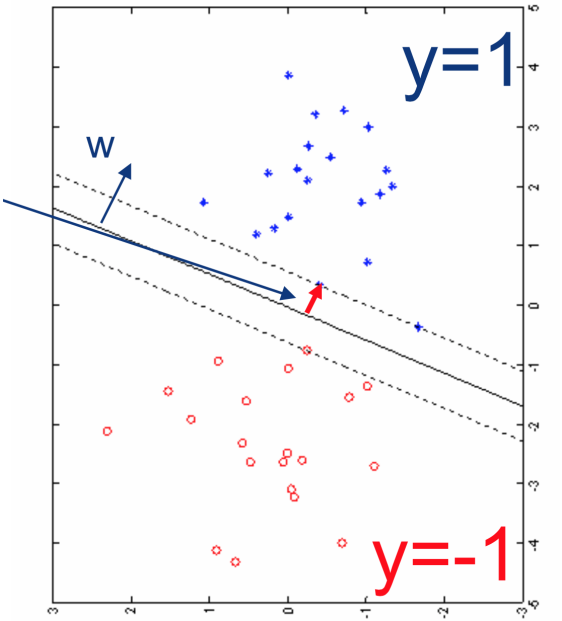
Maximisation d'une marge

- Parmi l'ensemble des hyper-plans $\boldsymbol{w}$ séparant les classes sans erreur
  - on cherche celui pour qui la marge λ (le sas de sécurité) est la plus grande
    - → celui pour lequel la généralisation sera la meilleure



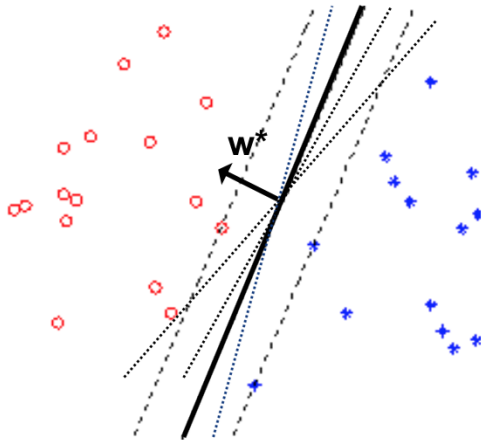
Maximisation d'une marge

- Il n'y aura pas d'erreur de classification si
  - $y_i \cdot (\boldsymbol{w}^T \boldsymbol{x}_i + w_0) > 0 \quad \forall i$
- Normalisation
  - pour éviter l'indétermination
  - on impose que la valeur de y au point le plus proche soit 1
    - $\min_i |\boldsymbol{w}^T \boldsymbol{x}_i + w_0| = 1$
    - donc $\gamma = \frac{1}{||\boldsymbol{w}||}$
- Maximiser la marge λ revient à
  - minimiser $||\boldsymbol{w}||$
- Il faut donc
  - $\min_{\boldsymbol{w}, w_0} ||\boldsymbol{w}||^2$ sous la contrainte $y_i \cdot (\boldsymbol{w}^T \boldsymbol{x}_i + w_0) \geqslant 1 \quad \forall i$



Maximisation d'une marge

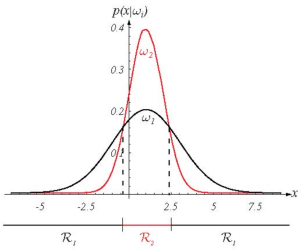
- Parmi l'ensemble des hyper-plans $\boldsymbol{w}$ séparant les classes sans erreur
  - on cherche celui pour qui la marge λ (le sas de sécurité) est la plus grande
    - $\rightarrow$ celui pour lequel la généralisation sera la meilleure
  - celui qui a la complexité la plus réduite
    - on favorise les $\boldsymbol{w}$ avec beaucoup de coefficients à zéro



source : Arshia COnt

Cas de séparations non-linéaires

- Comment fait-on dans le cas suivant ?
  - Pas de séparation linéaire
- Il faudrait utiliser un modèle pour la frontière de décision d'ordre plus élevé
  - $\boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} + \boldsymbol{W}^T\boldsymbol{x} + \boldsymbol{w}_0 = 0$
- Besoin d'un polynôme d'ordre très élevé de façon général
  - Beaucoup de paramètres
  - Complexité très grande



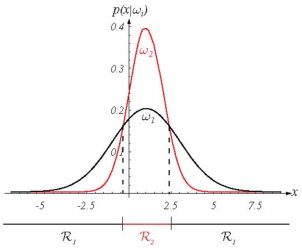
source : Arshia Cont

Cas de séparations non-linéaires

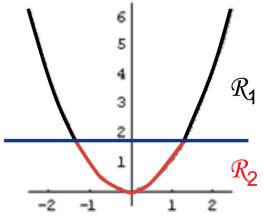
- Solution :
  - projeter les données dans un espace de dimension supérieur de manière à ce que dans cet espace les données soient linéairement séparables
    - $\phi : \mathcal{X} \rightarrow \mathcal{Z}$
    - $dim(\mathcal{Z}) > dim(\mathcal{X})$
  - Apprendre une frontière **linéaire** dans $\mathcal{Z}$ est équivalent à apprendre une frontière **non-linéaire** dans $\mathcal{X}$
- Exemple :

$$\phi : \mathbb{R} \rightarrow \mathbb{R}^2$$
$$x \rightarrow (x, x^2)$$

- ϕ est appelé un **noyau** (kernel)



source : Arshia Cont



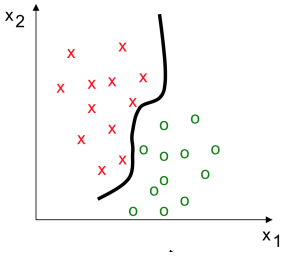
source : Arshia Cont

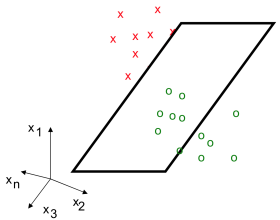
Cas de séparations non-linéaires

- Solution :
  - projeter les données dans un espace de dimension supérieur de manière à ce que dans cet espace les données soient linéairement séparables
    - $\phi : \mathcal{X} \rightarrow \mathcal{Z}$
    - $dim(\mathcal{Z}) > dim(\mathcal{X})$
  - Apprendre une frontière **linéaire** dans $\mathcal{Z}$ est équivalent à apprendre une frontière **non-linéaire** dans $\mathcal{X}$
- Exemple :

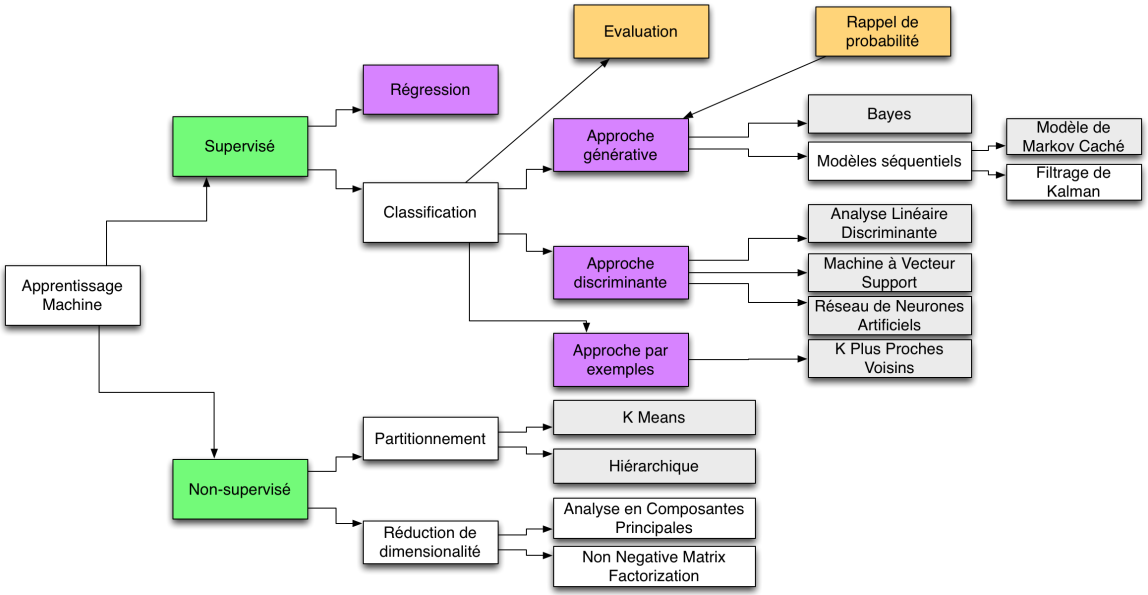
$$\phi : \mathbb{R} \rightarrow \mathbb{R}^2$$
$$x \rightarrow (x, x^2)$$

- ϕ est appelé un **noyau** (kernel)





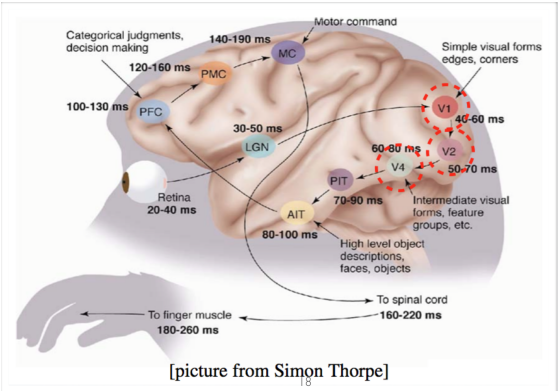
source : Arshia Cont


Apprentissage Machine
Supervisé
Non-supervisé
Régression
Classification
Partitionnement
Réduction de dimensionalité
Evaluation
Approche générative
Approche discriminante
Approche par exemples
Rappel de probabilité
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
K Plus Proches Voisins
K Means
Hiérarchique
Analyse en Composantes Principales
Non Negative Matrix Factorization

Réseaux de neurones artificiels

Cerveau et neurones

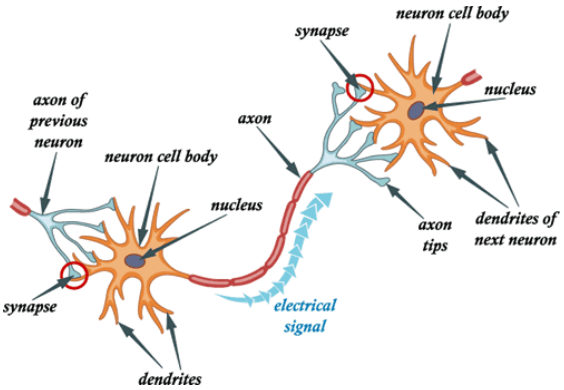
- Les réseaux de neurones artificiels tentent de reproduire la manière dont le cerveau traite l'information
- Dans le cerveau l'information est traité par un réseau complexe de neurones inter-connectés
- Les neurones des différentes régions du cerveau sont spécialisés dans des traitements spécifiques



source : Hugo Larochelle

Cerveau et neurones

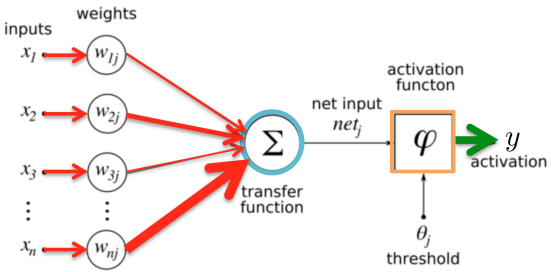
- Il y a environ 10^{10} à 10^{11} neurones dans notre cerveau
- Entre ces neurones circulent un signal électrique
- Tous les neurones sont connectés entre eux
- Chaque neurone
  - reçoit l'information à travers des **dendrites**
  - transforme l'information dans le corpus de sa cellule (**soma**)
  - retourne un signal à travers un cable appelé **axon**
- le point de connection entre ce cable (axon) et les dendrites des autres neurones sont appelés **synapses**



source : http://biology.stackexchange.com/questions/25967/nerves-neurons-axons-and-dendrites-by-example

Réseau de neurones artificiels

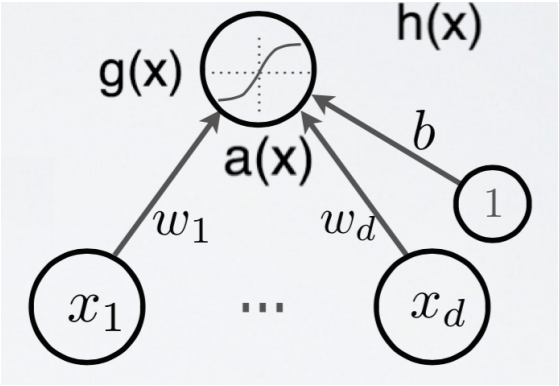
- Un Réseau de Neurone Artificiel (Artificial Neural Network - ANN) reproduit l'interconnection entre les différents neurones
- Chaque neurone est représenté par une fonction
  - prenant en **entrée** le signal des autres neurones
    - pondération spécifique à chacun d'entre eux
  - effectue une **transformation** de la somme des signaux résultants
  - **retourne** le signal vers l'étage de neurones suivant



source : Ph. Esling

Réseau de neurones artificiels

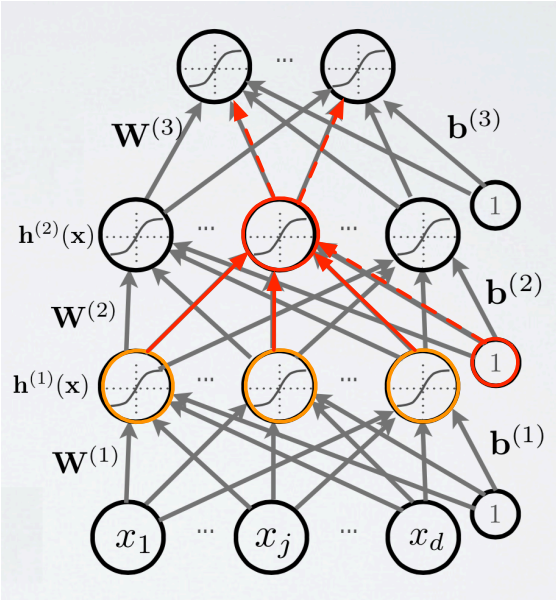
- Chaque neurone est représenté mathématiquement par
- 1 **Pre-activation** d'un neurone (activation des entrées)
  - $a(\boldsymbol{x}) = \sum_i w_i x_i + b = \boldsymbol{w}^T \boldsymbol{x} + b$
    - $\boldsymbol{w}$ sont les poids des connections (détermine quels neurones précédents apportent une information)
    - b est le biais du neurone
- 2 **Activation** du neurone
  - $h(\boldsymbol{x}) = g(a(\boldsymbol{x})) = g(\boldsymbol{w}^T \boldsymbol{x} + b)$
    - g est la fonction d'activation (généralement une fonction non-linéaire)



source : Hugo Larochelle

Réseau de neurones multi-couches

- Les neurones artificiels sont organisés en couches (layer)
  - Multi-Layer-Perceptron (MLP)



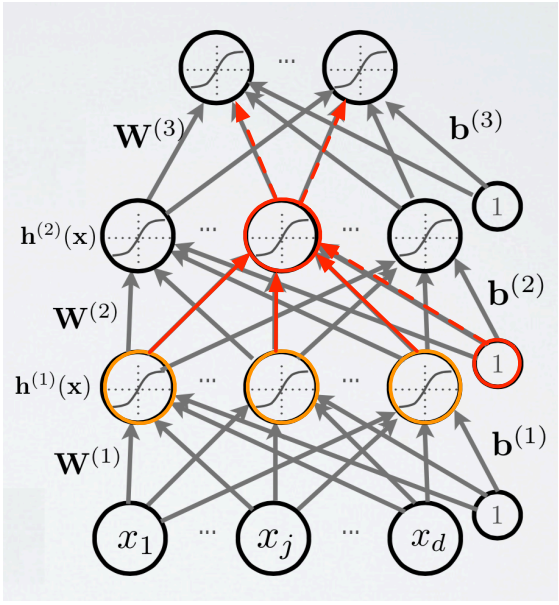
source : Hugo Larochelle

Réseau de neurones multi-couches

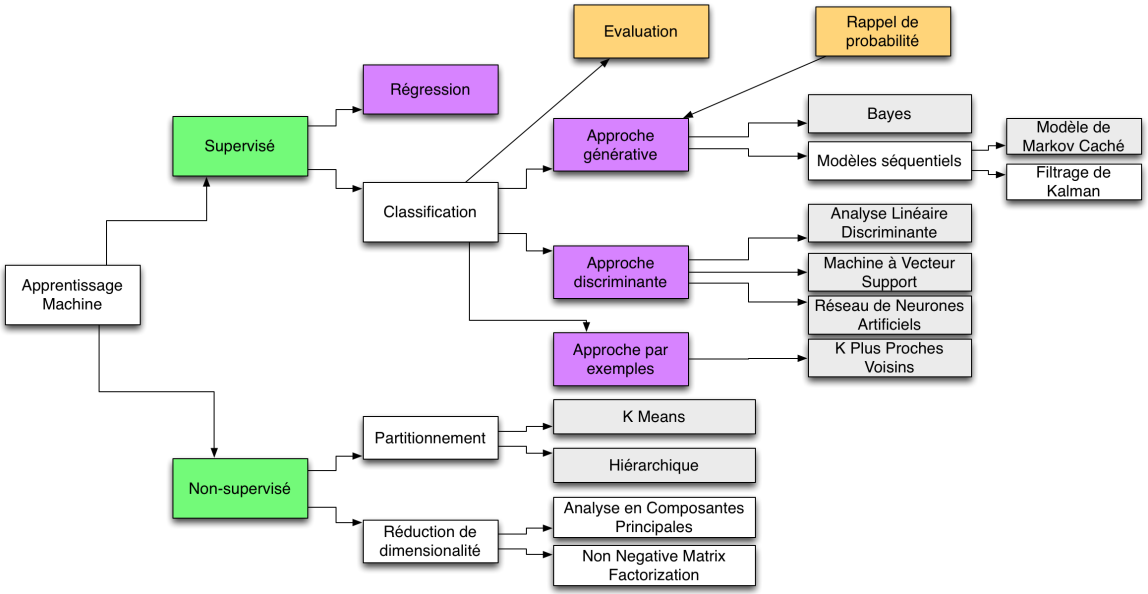
- 1 **Pre-activation** d'un neurone à l'étage k
  - $\boldsymbol{a}^{(k)}(x) = \boldsymbol{w}^{(k)}\underline{h}^{(k-1)}(\boldsymbol{x}) + \underline{b}^{(k)}$
- 2 **Activation** d'un neurone à l'étage k
  - $\underline{h}^{(k)} = \underline{g}(\boldsymbol{a}^{(k)}(\boldsymbol{x}))$
- 3 Activation de l'**étage de sortie**
  - $\underline{h}^{(L+1)} = \underline{o}(\boldsymbol{a}^{(L+1)}(\boldsymbol{x})) = \underline{f}(\boldsymbol{x})$

Entraînement d'un Réseau de Neurones Artificiels

- beaucoup de paramètres à apprendre : $\boldsymbol{w}^{(k)}, \underline{b}^{(k)}, \cdots$
- on impose les valeurs à l'étage de sortie
- algorithme de **back propagation**
- descente de gradient

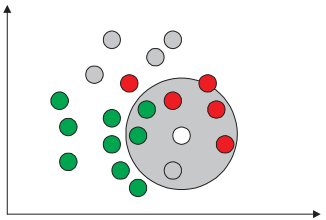


source : Hugo Larochelle


Apprentissage Machine
Supervisé
Régression
Classification
Evaluation
Rappel de probabilité
Approche générative
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Approche discriminante
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
Approche par exemples
K Plus Proches Voisins
Non-supervisé
Partitionnement
K Means
Hiérarchique
Réduction de dimensionalité
Analyse en Composantes Principales
Non Negative Matrix Factorization

Approche par exemplification

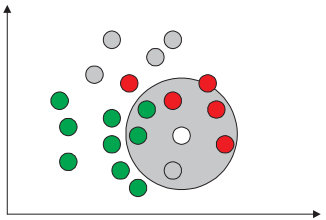
K Plus Proches Voisins



- Note : une observation $\boldsymbol{x}_n$ est un point dans un espace à D dimensions, appelé espace des descripteurs
- **Entraînement** :
  - exemple : $\boldsymbol{x}_n$ = la valeur des descripteurs audio à D dimensions
  - on remplit l'espace des descripteurs par l'ensemble des N "points" d'apprentissage : $\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$
  - à chaque point x_n est associé sa classe $t_n \in \{1, \cdots, C\}$
- **Evaluation** :
  - Soit $x^\star$ une observation nouvelle de classe $t^\star$ inconnue
  - On recherche dans l'espace des descripteurs les K points les plus proches de $x^\star$ selon une distance
    - généralement on utilise une distance euclidienne
  - On associe à $x^\star$ la classe majoritaire parmi celles assignées au K plus proche voisins $\{t_k\}$
    - $t^\star = \arg\max_{c \in \{1, \cdots, C\}} \sum_{k=1}^{K} \delta(c, t_k)$

K Plus Proches Voisins

- **Paramètres** :
  - on doit choisir le nombre K de plus proches voisins considérés
  - on doit choisir le type de distance utilisé (euclidienne ou autres)
    - si euclidien, cela suppose que les dimensions d sont d'échelles comparables ; sinon normalisation
- **Avantage** :
  - Il n'y a pas de modèle à apprendre !
- **Désavantage** :
  - demande le stockage et l'accès à toutes les données (le nombre de données peut être très très grand)
  - il faut calculer la distance entre $\boldsymbol{x}_m$ et tous les point $\boldsymbol{x}_k$
    - ce coût de calcul peut être très important

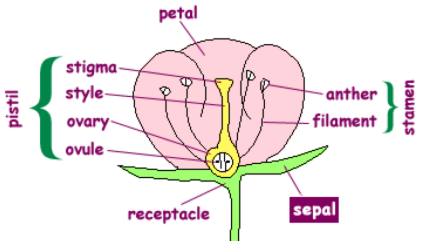


Algorithm des K plus proches voisins. Le point blanc est de classe inconnue. Si $K = 6$ on lui assignera la classe "rouge" car c'est la classe majoritaire parmi c'est 6 plus proches voisins.

Base IRIS

- $\mathbb{D}_{train,test} = \{(x_1, t_1), \cdots, (x_N, t_N)\}$

- Observations $\{x_1, \cdots, x_N\}$
  - 1. sepal length in cm
  - 2. sepal width in cm
  - 3. petal length in cm
  - 4. petal width in cm

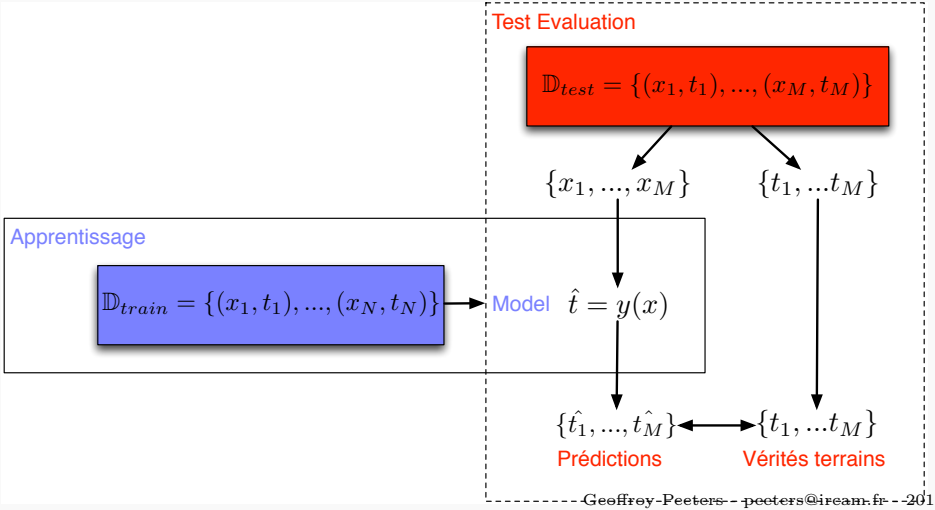


- Cibles $\{t_1, \cdots, t_N\}$
  - 1. Iris Setosa
  - 2. Iris Versicolour
  - 3. Iris Virginica

Evaluation d'un système de classification

Evaluation des performances d'un système de classification

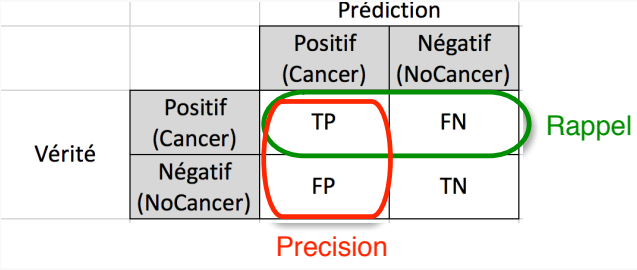
- L'ensemble de test $\mathbb{D}_{test}$ doit être différent de l'ensemble d'entraînement $\mathbb{D}_{train}$
  - On test la généralisation du modèle $y(x)$
- On compare les prédictions de classes $\hat{t}_m$ aux "vérités terrains" de classes t_m
  - Comment compare-t'on ?



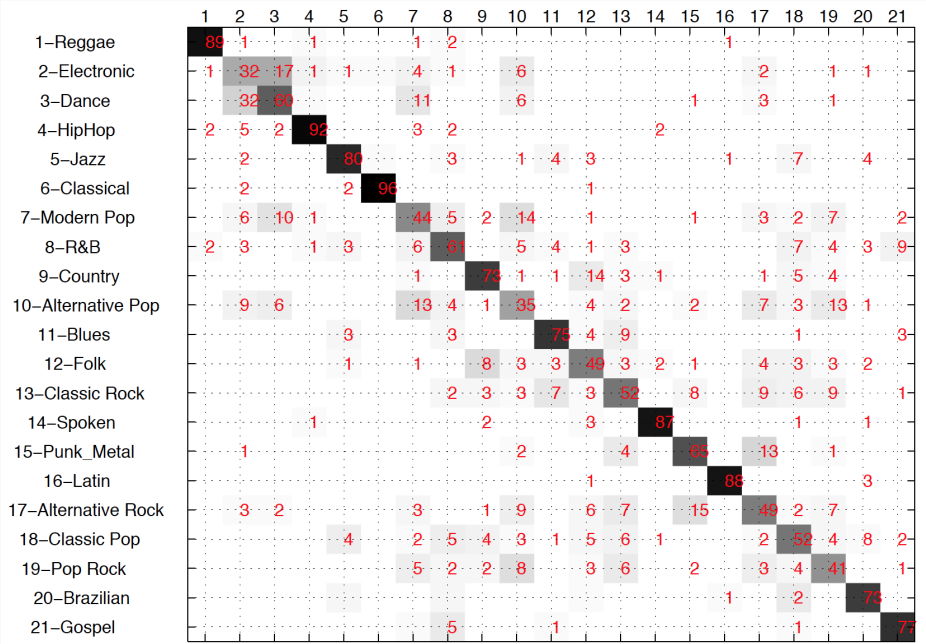
Calcul des TP, FP, TN, FN

- Pour deux classes (c_1=+, c_2=-)
- **True Positif** (TP) : # de données + détectées correctement (True) comme +
- **False Negatif** (FN) : # de données + détectées faussement (False) comme -
- **False Positif** (FP) : # de données - détectées faussement (False) comme +
- **True Negatif** (TN) : # de données - détectées correctement (True) comme -

Matrice de confusion



Exemple de matrice de confusion



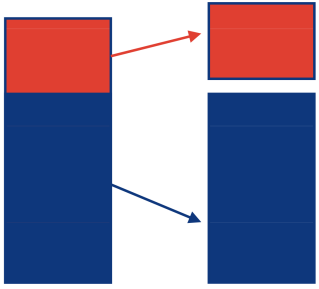
Calcul des indices résumés

$$
\begin{aligned}
Rappel(Recall) &= \text{Mesure la capacité à retrouver tous les } c_i \\
&= \frac{\#\text{ données détectées comme } c_i \text{ et étant réellement } c_i}{\#\text{ données étant réellement } c_i} \\
&= \frac{TP}{TP+FN} \\
Precision &= \text{Mesure la capacité à retrouver uniquement des } c_i \text{ (moteur de recherch} \\
&= \frac{\#\text{ données détectées comme } c_i \text{ et étant réellement } c_i}{\#\text{ données détectées comme } c_i \text{ (correctes ou fausses)}} \\
&= \frac{TP}{TP+FP} \\
F-measure &= \text{Prend en compte simultanément le Rappel et la Précision} \\
Accuracy &= \text{Mesure les performances globales indépendamment de la distribution} \\
&= \frac{TP+TN}{TP+FP+TN+FN}
\end{aligned}
$$

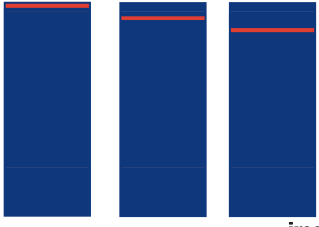
(1)

Séparer ensemble d'entraînement et de test

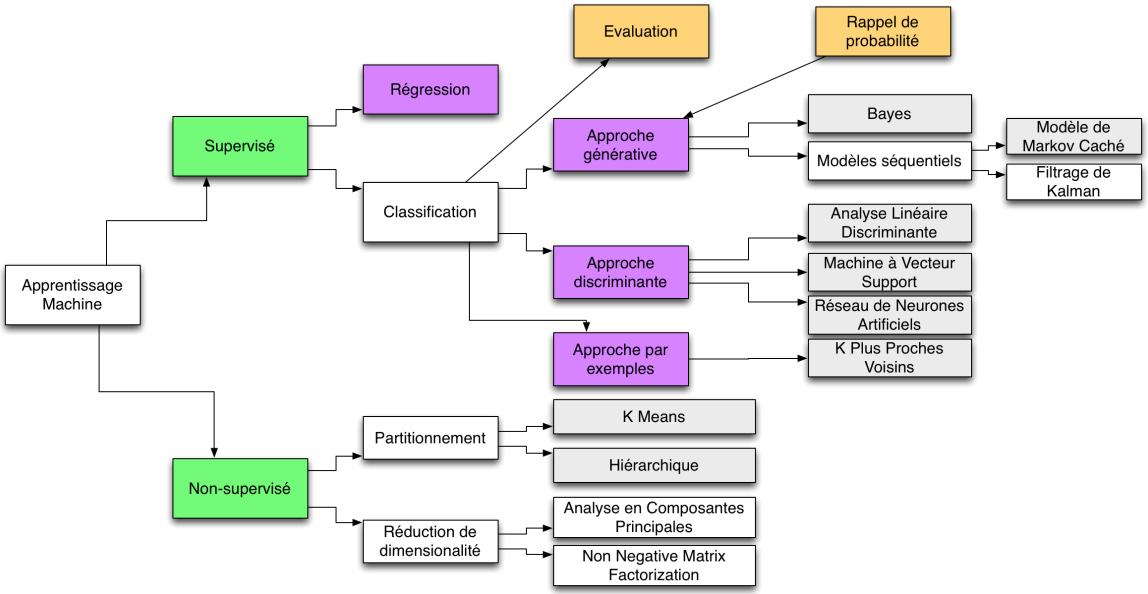
- 1. Deux ensembles séparées
- 2. Un ensemble unique
  - 2.1 **N-Fold Cross Validation** (validation à N-plis croisés)
    - $\mathbb{D}$ est divisé en N sous-ensemble $\mathbb{D}_n$
    - test : on choisit un sous-ensemble
    - entraînement : on utilise les $N-1$ autres
    - on réitère en choisissant un nouveau sous-ensemble de test parmi les N possibles
    - on calcul la moyenne des indices
  - 2.2. **Leave-one-out Cross-Validation** :
    - cas limite quand N=le nombre de données
    - test : on choisit une donnée
    - entraînement : on utilise toutes les $N-1$ autres données
    - on réitère ...
    - on calcul la moyenne des indices



source : Arshia Cont



source : Arshia Cont

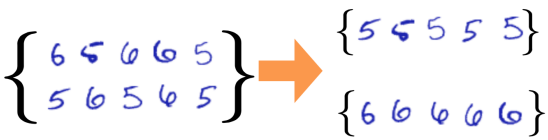

Apprentissage Machine
Supervisé
Non-supervisé
Régression
Classification
Partitionnement
Réduction de dimensionalité
Evaluation
Approche générative
Approche discriminante
Approche par exemples
Rappel de probabilité
Bayes
Modèles séquentiels
Modèle de Markov Caché
Filtrage de Kalman
Analyse Linéaire Discriminante
Machine à Vecteur Support
Réseau de Neurones Artificiels
K Plus Proches Voisins
K Means
Hiérarchique
Analyse en Composantes Principales
Non Negative Matrix Factorization

Apprentissage non-supervisé

Nous considérons un ensemble d'**observations** (entrées du système) $\{x_1, \cdots, x_N\}$

Apprentissage **non-supervisé**

- Nous donnons ne donnons pas à la machine les cibles
- $D = \{x_1, \cdots, x_N\}$
- L'objectif de la machine est de créer un modèle de x, un partitionnement (clustering) des données
  - peut être utilisé pour analyser les données, prendre des décisions



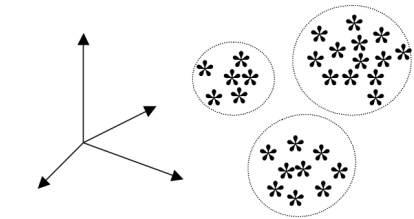
source : Hugo Larochelle

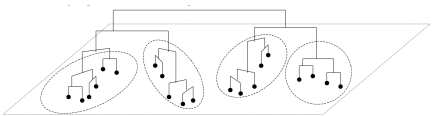
Apprentissage non-supervisé = Algorithme de clustering

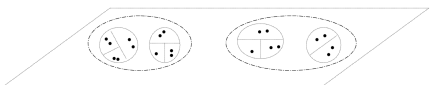
- Algorithme de clustering
  - divise un groupe hétérogène de données, en sous-groupes de manière que
    - les données considérées comme les plus similaires soient associées au sein d'un groupe homogène et qu'au contraire
    - les données considérées comme différentes se retrouvent dans d'autres groupes distincts
- **Objectif**
  - trouver la structure sous-jacente à un ensemble de données

Grandes classes d'algorithmes

- Méthodes **non-hiérarchiques**
  - on considère toutes les données simultanément
  - exemple : k-means
- Méthodes **hiérarchiques**
  - **ascendantes** :
    - on **agglomère** progressivement les données deux à deux
    - exemple : algorithme agglomératif hiérarchique
  - **descendantes** :
    - on **divise** à chaque étage les données en deux groupes
    - exemple : KD-tree



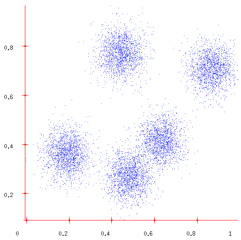


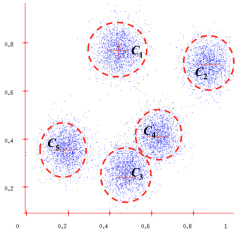


source : Christine Decaestecker, Marco Saerens

Algorithmes des K-Means (nuées dynamiques)

- Méthode non-hiérarchique
- On cherche à **organiser** l'ensemble de données $\mathbb{D} = \{x_1, \cdots, x_N\}$ en K **ensembles** (clusters)
- On cherche à **représenter** l'ensemble de données $\mathbb{D} = \{x_1, \cdots, x_N\}$ par les **prototypes** μ_k des K clusters



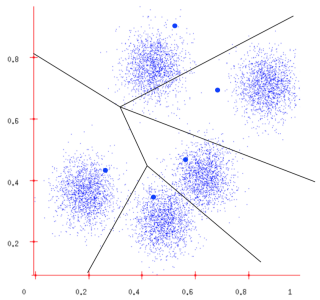


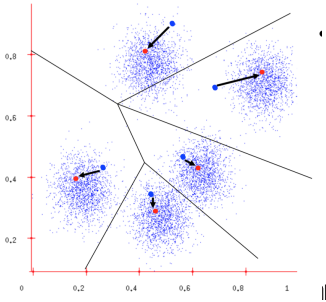
source : Philippe Esling

- **Assignation** :
  - r_{nk}=1 si donnée n appartient au cluster k
  - r_{nk}=0 sinon
- **Erreur commise** :
  - $J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2$
- **Objectif** : minimiser l'erreur
- **Comment** : algorithme Expectation-Maximization (EM)

Algorithme Expectation-Maximisation (EM)

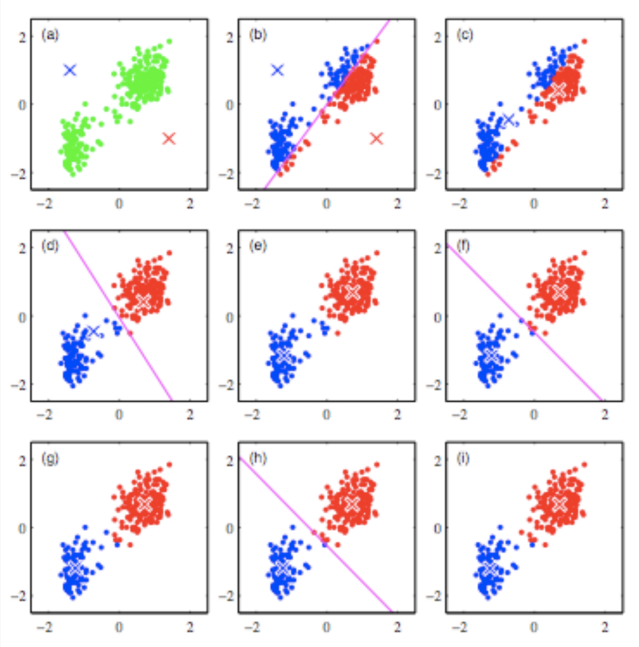
- **Initialisation** :
  - choisir K clusters (random, KD-tree)
  - calculer le centroide μ_k de chaque cluster à partir des objets attribués à k
  - $\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$
- **Etape E (Expectation) :**
  - attribuer chaque objet au cluster dont il est le plus proche (distance euclidienne)
  - $r_{nk} = 1$ si $k = \arg\min_j ||x_n - \mu_k||^2$,
  - $r_{nk} = 0$ sinon
- **Etape M (Maximization) :**
  - étant donné la nouvelle attribution des objets aux clusters, recalculer les centroides
  - $\mu_k = \frac{\sum_n r_{nk} x_n}{\sum_n r_{nk}}$
- **Itération :**
  - réitérer jusqu'à ce que l'erreur ne se réduise plus





source : Philippe Esling

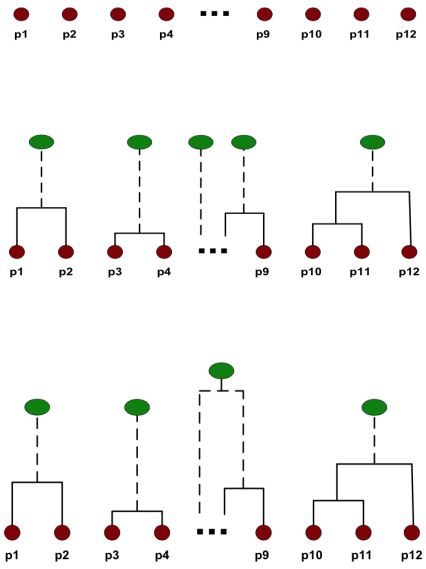
Algorithmes des K-Means



source : Archie Cont

Algorithme hiérarchique par agglomération

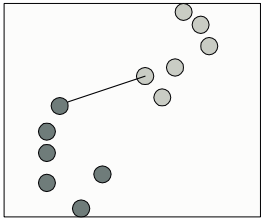
- Méthode hiérarchique (ascendante par agglomération)
- **Initialisation :**
  - chaque objet constitue un cluster
- **Itération :**
  - regroupement des deux objets les plus proches
    - **distance** entre objets
      - euclidienne, Minkowski, cosine
  - ou d'un objet à un cluster
  - ou des deux clusters les plus proches
    - **linkage** entre objets
- **Condition d'arrêt :**
  - on arrive au sommet de l'arbre, ou bien on a obtenu K clusters

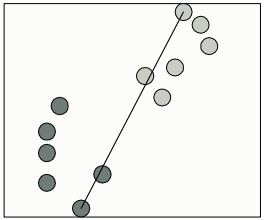


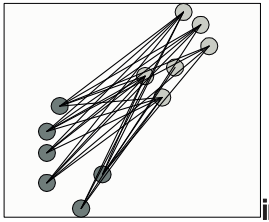
source : Philippe Esling

Linkage entre objets d'un cluster

- **Single :**
  - plus petite distance entre les objets dans les deux groupes — $d(r,s) = \min(dist(r_i, s_j))\ i \in r\ j \in s$
- **Complete :**
  - plus grande ...
  - $d(r,s) = \max(dist(r_i, s_j))\ i \in r\ j \in s$
- **Average :**
  - moyenne des distances entre toutes les paires d'objets entre les deux groupes
  - $d(r,s) = \text{mean}(dist(r_i, s_j))\ i \in r\ j \in s$
- **Centroid :**
  - distance entre les centroides des deux groupes
  - $d(r,s) = dist(\overline{r}, \overline{s})$
- **Ward :**
  - représente l'augmentation de l'inertie intra-groupe due à la réunion des groupes r et s
  - $d(r,s) = \frac{n_r n_s}{n_r + n_s} dist(\overline{r}, \overline{s})$

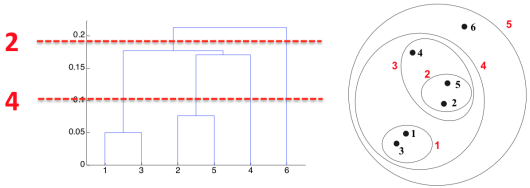






Dendrogramme

- Représente graphiquement sous forme d'arbre binaire les différentes connections entre objets/clusters.
- La hauteur de la connexion (distance Cophenetic) entre deux objets/clusters représente la distance entre les deux objets/ clusters connectés.



source : Philippe Esling