Data Science

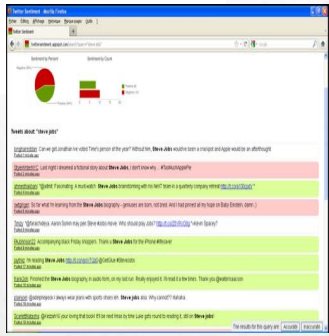
T. Artières, C. Capponi, H. Kadri, P. Milanesi

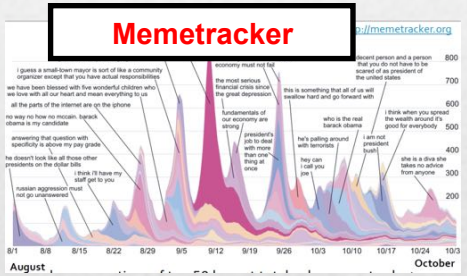
Thierry.Artieres@centrale-marseille.fr, thierry.artieres@lis.fr
Ecole Centrale Marseille
Equipe d'Apprentissage Automatique (QARMA)
LIS, Laboratoire d'Informatique et Systèmes

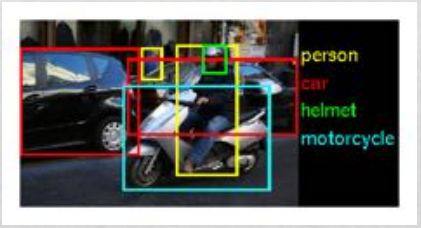
Objectifs

- Vous initier à l'apprentissage automatique
- Avoir une idée de :
  - Quand les machines peuvent-elles apprendre ?
  - Pourquoi les machines peuvent-elles apprendre ?
  - Comment les machines peuvent-elles apprendre ?

- Applications en texte
  - Recherche d'information (Bing, Google, etc)
  - Résumé automatique
  - Détection de sentiments /fake reviews …
  - Traduction automatique (Google Translate)
- Application en vision
  - Reconnaissance de chiffres/ caractères (Deep NNs on Mnist)
  - Classification d'objets dans les images (VOC challenge)
  - Reconnaissance de visages etc (Picasa)
  - Analyse de vidéos
- Application en parole
  - Reconnaissance du locuteur
  - Reconnaissance de parole
  - Dialogue « multilingue »
  - Reconnaissance musicale (*Shazam*)





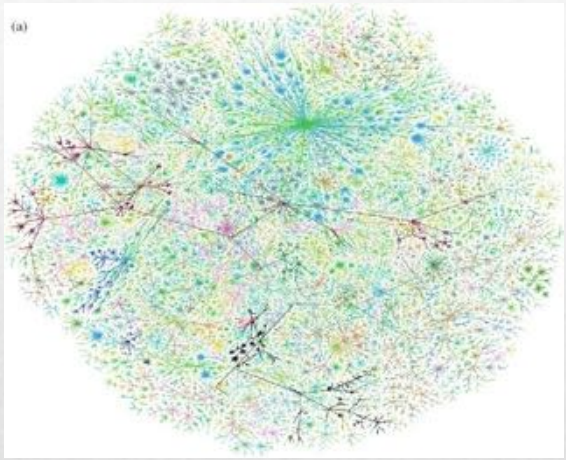


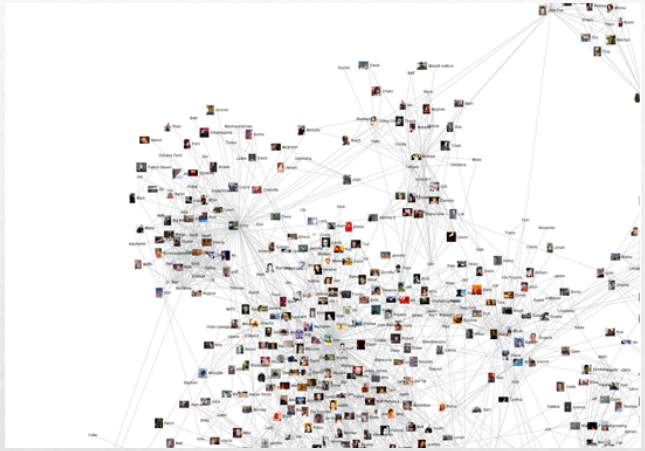
Application sur des données multimédia

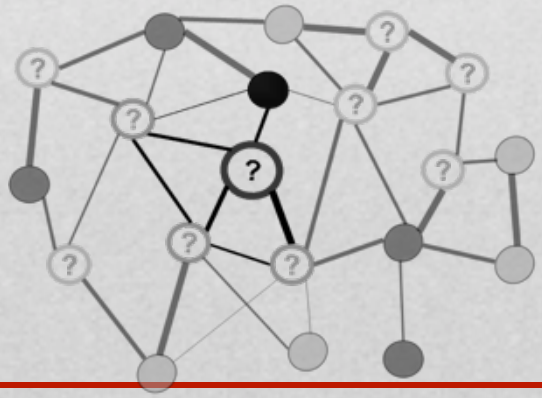
Applications orientées web
Systèmes de recommandation

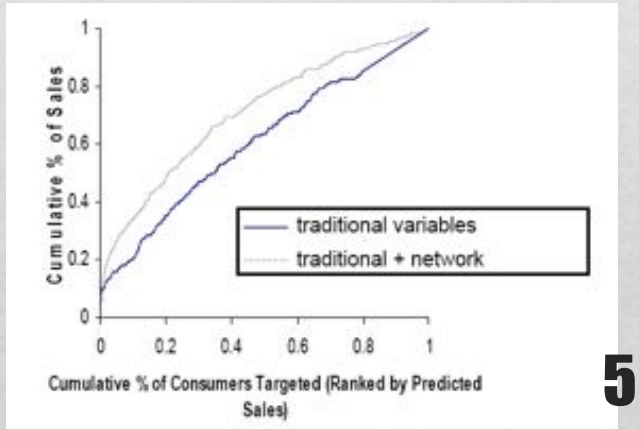
D'après [Ricci]

Applications orientées web
Réseaux sociaux et grands graphes









- Santé
  - Syndrome locked in
- Jeux et usages courants
  - Commande
  - État émotionnel

Brain Computer interfaces
Brain reading

- Kaggle
  - Classification dans un très grand nombre de classes
  - Titanic
  - Dog vs Cats

  …

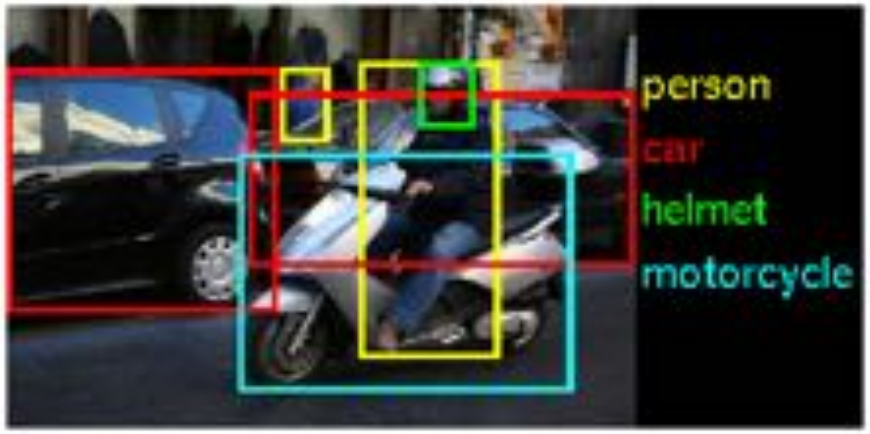
Pour s'amuser… mais pas seulement !

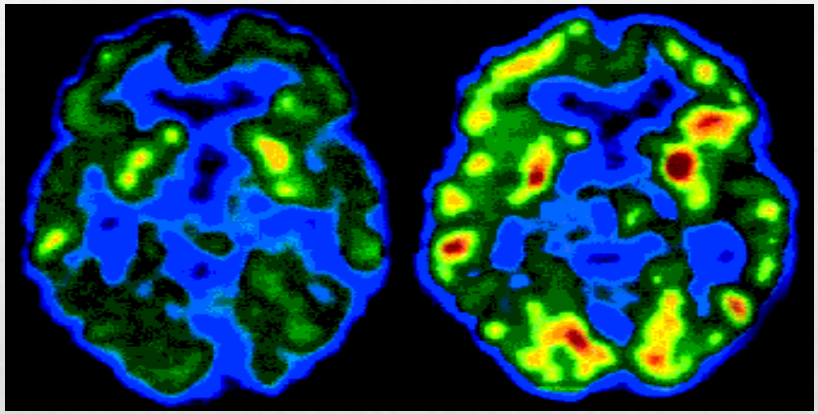
- Nombreux cours en ligne disponible
  - Machine Learning / Andrew Ng
  - Deep Learning / G. Hinton

- Plateformes MOOC (par ex. Coursera)
  - Practical Machine Learning
    - https://www.coursera.org/course/predmachlearn
  - The datascientist toolbox
    - https://www.coursera.org/course/datascitoolbox
  - …

Pour étudier

QUAND UTILISER L'APPRENTISSAGE À PARTIR D'EXEMPLES ?





Programmation classique : Phase d'analyse ?

- Lorsque l'on ne sait pas écrire le programme
- Lorsque le programme est susceptible de changer rapidement fréquemment (spam filtering)
- Lorsque cela permet de développer plus vite

Quand utiliser l'apprentissage pour écrire un programme ?

Les problèmes génériques

On cherche à apprendre une fonction $y = f(x)$

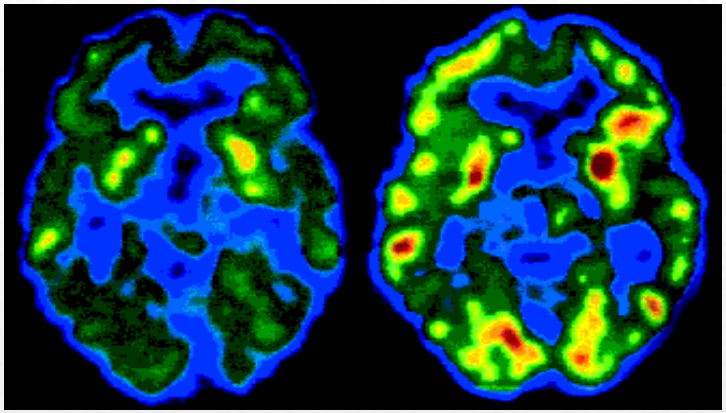
- où
  - x = forme observée
    - x peut être discret ou continu ou mixte
  - y = sortie associée
    - Réel, vecteur de réel (Régression)
    - Catégorie (classification)
    - …

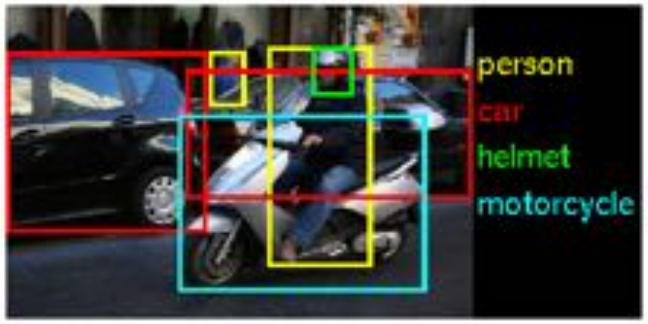
$x \in X^1 \times X^2 \times ... \times X^p$

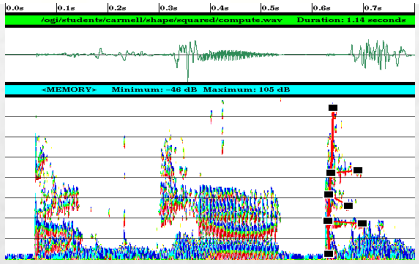
- A partir d'un échantillon fini (i.i.d.) d'exemples étiquetés $\{(x^i, y^i), i = 1..L\}$
  - Les observations sont des réalisations d'une loi jointe *p(x,y)*

- En général *f* est paramétrée par un jeu de paramètres W

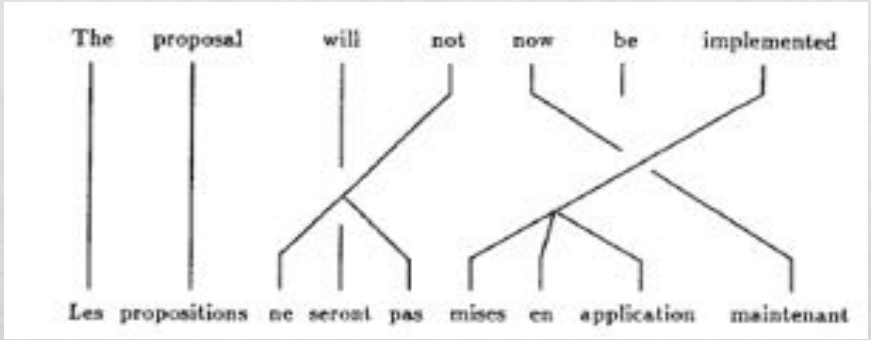
Apprentissage supervisé





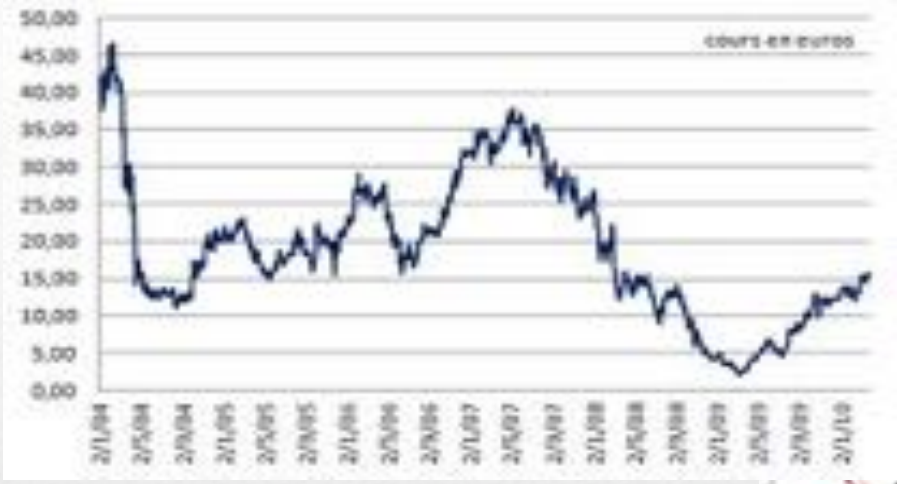


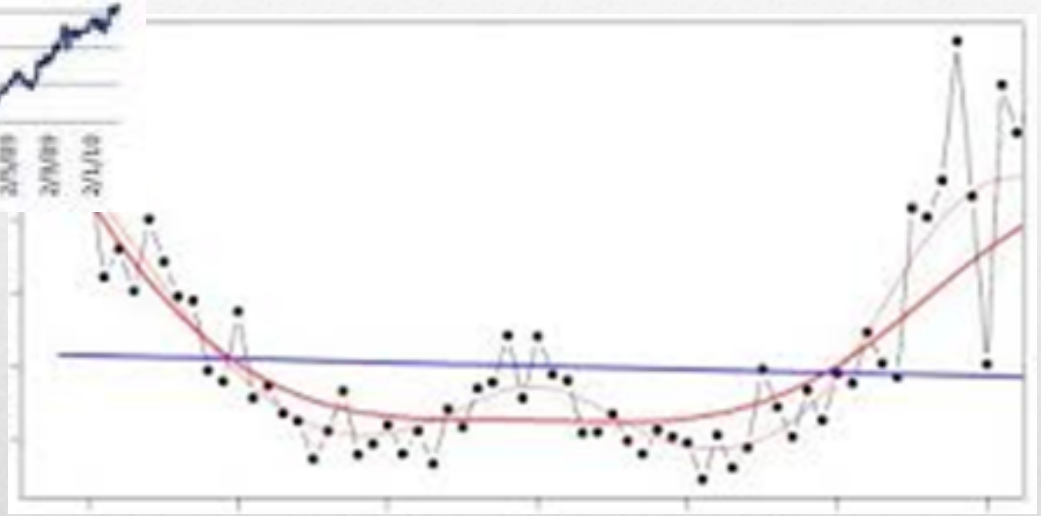
Titanic challenge



Discrimination

$y=F(x)$ où y est une classe





$y=F(x)$ où y est une valeur/un vecteur réel

Prédiction et régression

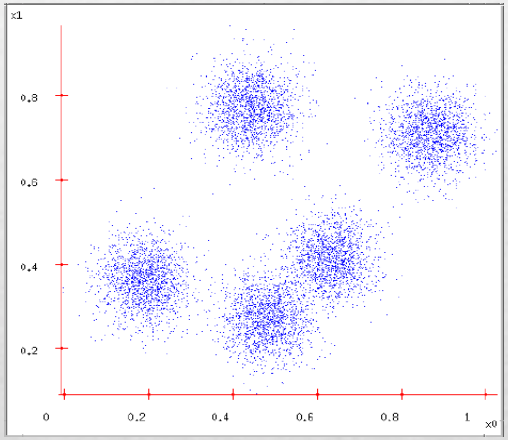
On cherche à apprendre une fonction $x \implies f(x)$

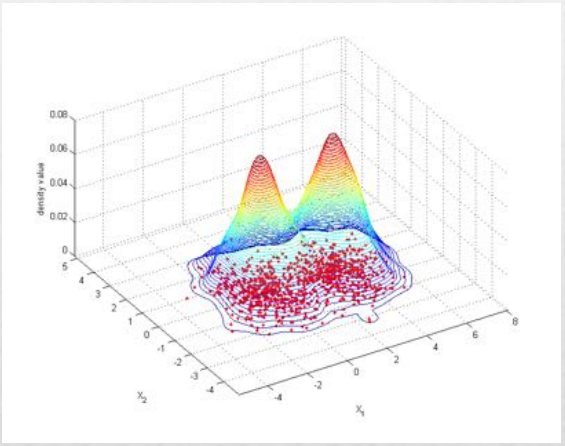
- où
  - x = forme observée

A partir d'un échantillon fini (i.i.d) d'exemples non étiquetés $\{(x^i), i = 1..U\}$

Apprentissage non supervisé

- Partitionnement / clustering
- Estimation de densité
- Découverte de facteurs cachés explicatifs





Applications du cadre non supervisé

On cherche à apprendre une fonction

$$y = f(x)$$

- où
  - x = forme observée
  - y = sortie associée
- A partir d'un échantillon fini (i.i.d.)
  - d'exemples étiquetés $\{(x^i, y^i), i = 1..L\}$
  - et non étiquetés $\{(x^i), i = L+1..L+U\}$

- En général *f* est paramétrée par un jeu de paramètres W
- Même applications que le supervisé

Apprentissage semi-supervisé

On cherche à apprendre une fonction $d = f(e)$

- Où
  - e = forme observée : état
  - d = sortie associée : décision

- A partir d'un échantillon de trajectoires où les actions sont décidées par l'algorithme

- Cadres d'applications : robotique, jeux, stratégie, décision séquentielle

Apprentissage par renforcement

En pratique

- Cadre statistique
  - Phénomène régi par des lois inconnues dont on observe des réalisations
  - A partir de ces observations on veut inférer des connaissances, conclusions, etc
- Les données
  - Représentent la principale connaissance sur le phénomène étudié
  - On distingue
    - Les données d'apprentissage
    - Les données de validation
    - Les données de test

1. Données : recueil, prétraitement
   - coût du recueil
   - représentativité des données : quantité, choix
   - qualité des prétraitements
2. Choix d'un modèle adapté
   - a priori (pendant l'apprentissage)
3. Apprentissage
   - ensemble d 'apprentissage, apprendre les paramètres du modèle choisi, estimation
4. Validation
   - critères de validation, ensemble test ou autre méthode
5. Utilisation

Les étapes de résolution d'un problème

- On met au point un système sur des données

- **On utilise le système sur d'autres données**
  - **Notion de capacité de généralisation**

Une difficulté majeure

Décision Bayésienne

Un cadre théorique pour la classification

- Cadre de la décision statistique
- Problème
  Discrimination : décider de la classe C d'une observation x
- Objectif
  Minimiser
  - Probabilité de décision erronée
  - Moyenne des coûts produits par des décisions erronées
- Hypothèse : on connaît toutes les probabilités

- Règle de décision $r : X \rightarrow \{C_1, C_2, ..., C_K\}$, $x \rightarrow r(x)$
- Règle Bayésienne $r(x) = \arg\max_{C_k} \{P(C_k / x)\}$
- Propriété
  - Minimise la probabilité d'erreur de classification

Théorie de la décision Bayesienne

- Implémentation de la règle Bayésienne par approximation des quantités nécessaires
  - Probabilités a posteriori / fonctions discriminantes
  - Densités de probabilités / modèles génératifs

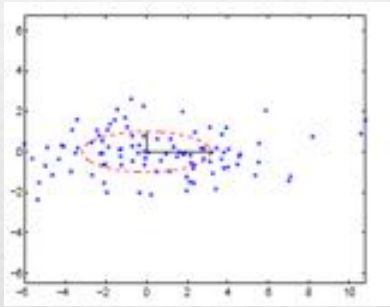
Lien entre théorie Bayésienne et reconnaissance des formes

CLASSIFIEUR GAUSSIEN

- Densité en dimension 1 $$p(x) = \frac{1}{\sqrt{2\Pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$
- Densité en dimension quelconque $$p(x) = \frac{1}{(2\Pi)^{p/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

- Matrice de covariance
  - Terme diagonal nul => variable constante !
  - Axes de variabilité // axes de coordonnées

    $\Rightarrow$Termes non diagonaux nuls
  - Nuage non // axes de coordonnées => Termes non diagonaux non nuls



- Loi gaussienne et distance

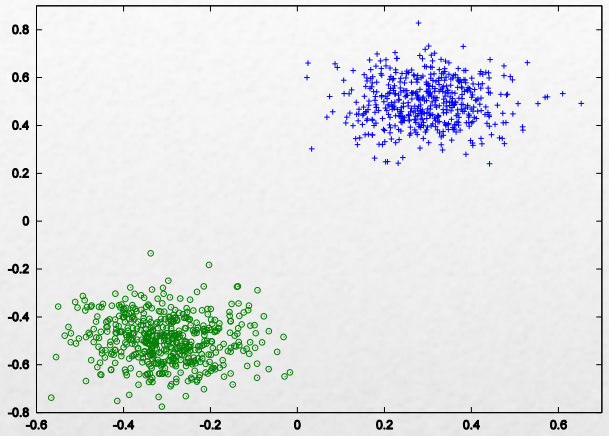
$$-logp(x) = \frac{1}{2} log|\Sigma| + \frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu) + Cste$$

$$-logp(x) \propto \|x-\mu\|^2 + C \text{ si } \sigma = Id$$

Loi Gaussienne

Classifieur Gaussien / Interprétation

- En supposant 2 classes gaussiennes de même Matrice de covariance (par exemple Identité)
- L'implémentation de la règle de décision Bayésienne correspond à
  - Une frontière de décision linéaire bissectrice du segment reliant les deux moyennes
  - Classer un point en fonction de la distance minimale à l'une des deux moyennes des classes



UN CLASSIFIEUR SIMPLE : LES K PLUS PROCHES VOISINS

- Principe
  - Pour classer un exemple x
    - On détermine dans la base d'apprentissage les K exemples qui sont les plus proches de x
    - On regarde parmi ces K voisins la classe majoritaire
    - On reconnait x de cette classe
- Intérêt
  - Modèle performant asymptotiquement
- Points faibles
  - Modèle performant asymptotiquement
  - Ce n'est pas un modèle à proprement parler
    - Pas de synthèse des données d'apprentissage

Les K ppv comme estimateurs de densité

- Pour un ensemble de données $D=\{x^i\}$, on fixe k

- Pour un point x quelconque, on considère une hypersphère centrée en x, on la fait grandir jusqu'à ce qu'elle englobe k points, soit V son volume

$$\Rightarrow \hat{p}(x) \equiv \frac{k}{NV} \approx p(x)$$

- k/N = proportion de points qui sont dans la sphère
  - En pratique $k \propto N^{1/2}$ donne une estimation raisonnable

- Si on a estimé la densité de la distribution *p(x)* par les k-ppv
  - Soit k_i le nombre de points de C_i parmi les k voisins
  - Soit V(x) le volume de la sphère

  ⇒On peut utiliser les estimateurs suivants :

$$\hat{p}(x/C_i) = \frac{k_i}{N_i V(x)} \quad et \quad \hat{p}(C_i) = \frac{N_i}{N}$$

- Implémentation de la règle de décision Bayésienne

$$x \in C_i \Leftrightarrow p(C_i/x) > p(C_j/x) \forall j \neq i$$

$$x \in C_i \Leftrightarrow k_i > k_j \forall j \neq i$$

- Soit e l 'erreur 1-ppv et e^* l'erreur Bayesienne (théorique), C le nombre de classes

- La moitié de l'information de discrimination disponible dans une population infinie d'exemples étiquetés est contenue dans le plus proche voisin

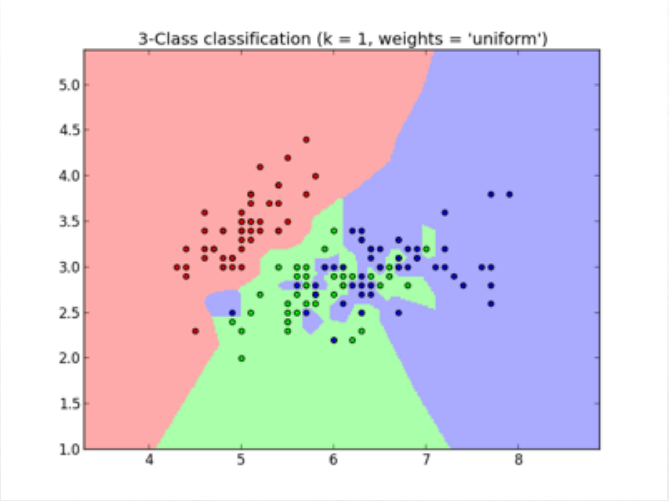
$$e^* \leq e \leq e^*(2 - \frac{C \times e^*}{C-1})$$

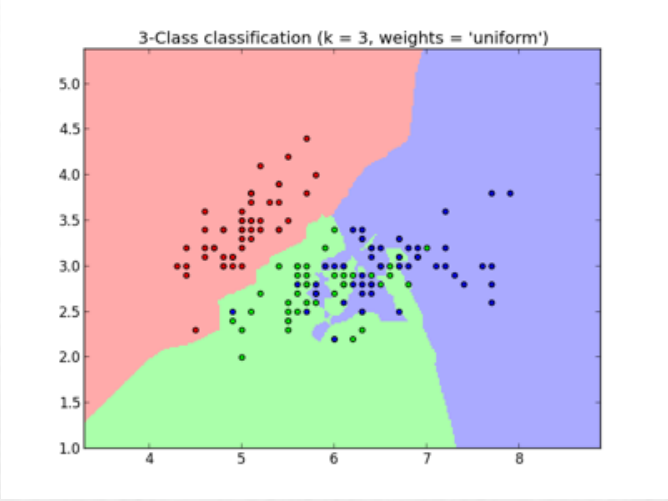
- Cas à 2 classes : soit e_k l 'erreur k-ppv:

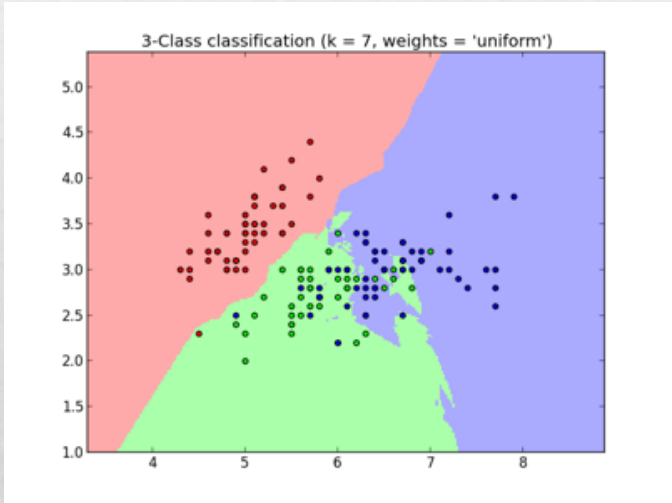
$$e^* < \ldots < e_k < e_{k-1} < \ldots < e_1 < 2e^*$$

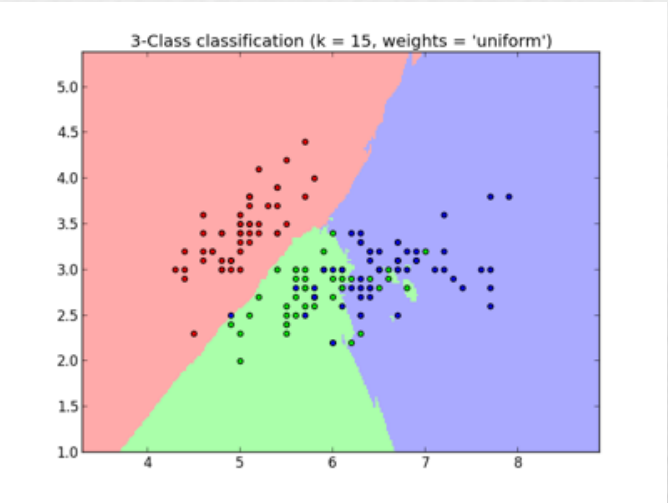
- Surfaces de décision cas 1-NN : classifieur linéaire par morceaux

Propriétés



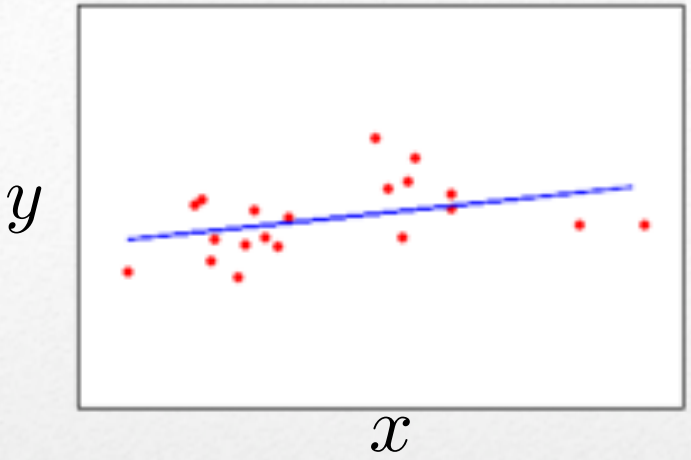






Nombre de voisins

Un premier modèle de régression : La régression linéaire



- On veut apprendre une fonction $y = F(x)$
- Modèle :
  - On considère que la fonction est linéaire => on utilise un modèle linéaire

$$F_w(x) = w_0 + w_1 x_1 + w_2 x_2 + ... + w_p x_p = \mathbf{w}^t x$$

- Apprentissage
  - On cherche les paramètres du modèle (les poids w) minimisant l 'erreur de prédiction

$$F(x) - F_w(x)$$

sur la base d'apprentissage $\{(x^i, y^i), i = 1..N\}$

- Ensemble d'apprentissage $\{(x^i, y^i), i = 1..N\}$

- Critère d'apprentissage
  - Sur un exemple : Erreur quadratique

$$l(f_{\mathbf{w}}(x), y) = \|y - f_{\mathbf{w}}(x)\|^2$$

  - Sur l'ensemble d'apprentissage : Erreur Quadratique (somme ou moyenne)

$$R_{emp}(\mathbf{w}) = \sum_{i=1}^{N} l\left(f_{\mathbf{w}}(x^i), y^i\right)$$

Critère d'apprentissage : Mean Squared Error (MSE)

- Hypothèse linéaire « légitime » ?
  - Et si ce n'est pas le cas ?
- Est-il bon d'avoir une erreur faible sur la base d'apprentissage ? i.e. la prédiction sur une nouvelle donnée sera-t-elle bonne ?
- Quelle est l'influence du nombre d'exemples ?

Questions

- Hypothèse linéaire légitime
  - Complexité d'une famille de modèles / VC Dim
  - Sélection de modèle
    - GridSearching, Cross Validation
    - Structural Risk Minimization
- Est-il bon d'avoir une erreur faible en apprentissage…?
  - Compromis Biais-Variance / Surapprentissage
  - Bornes de généralisation
- Influence du nombre d'exemples
  - Bornes de généralisation
  - Inégalités de concentration

Réponses

FORMALISATION DE L'APPRENTISSAGE COMME UN PROBLÈME D'OPTIMISATION

- Base d'apprentissage

$$\{(x^i, y^i), i = 1..N\}$$

- Critère d'apprentissage

$$R_{emp}(w) = \sum_{i=1}^{N} l\left(f_w(x^i), y^i\right)$$

- Fonction de perte : loss
  - Erreur quadratique
  - 0/1 loss
  - …

Le risque empirique

$$R_{emp}(w) = \sum_{i=1}^{N} l\left(f_w(x^i), y^i\right)$$

est une estimation du risque réel

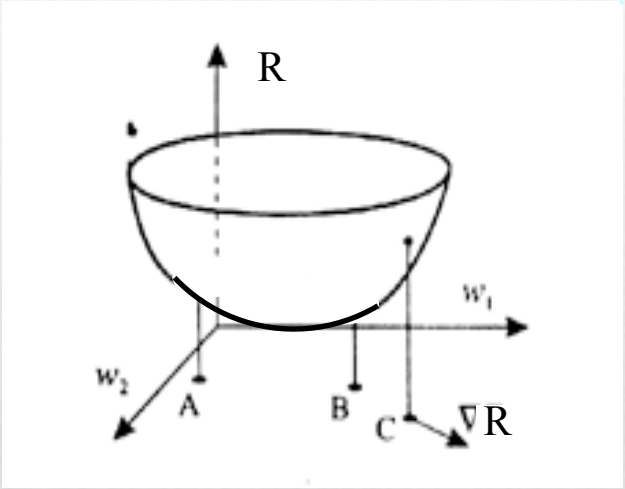
$$R(w) = \int_{(x,y)} l\left(f_w(x), y\right) p(x,y) dx dy$$

Risque réel et empirique

- Trouver le meilleur modèle sur la base d'apprentissage

$$R_{emp}(w) = \sum_{i=1}^{N} l\left(f_w(x^i), y^i\right)$$

= Optimisation d'une fonction de w



- Trouver le meilleur modèle est un peu différent

L'apprentissage comme un problème d'optimisation

EN PRATIQUE

- On apprend
  - Fonctions discriminantes
  - Lois de probabilités a posteriori
  - Densités de probabilités

  à partir d'une collection finie

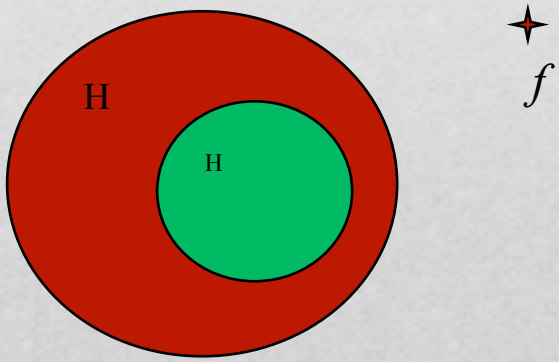
⇒ Décalage entre ce que l'on veut et ce que l'on obtient

- L'espace de recherche ne contient pas nécessairement la fonction que l'on cherche
- Décalage entre l'optimum (la fonction) pour la tâche et l'optimum pour la fonction de cout de l'apprentissage (échantillon fini, critère non adapté)
- L'optimisation itérative ne converge pas toujours vers l'optimum
- L'optimisation itérative dépend de l'initialisation

- Sensibilité du résultat de l'apprentissage
  - à la base d'apprentissage (taille et exemples)
  - Aux hyper-paramètres de l'algorithme d'optimisation
  - A l'initialisation de l'algorithme d'optimisation (si itératif)
  - A la structure du modèle (e.g. #degré de l'expansion polynimiale)

- Plus H (l'espace de fonctions dans lequel on cherche une fonction optimale) est grand
  - Plus la variance de l'estimateur est grande

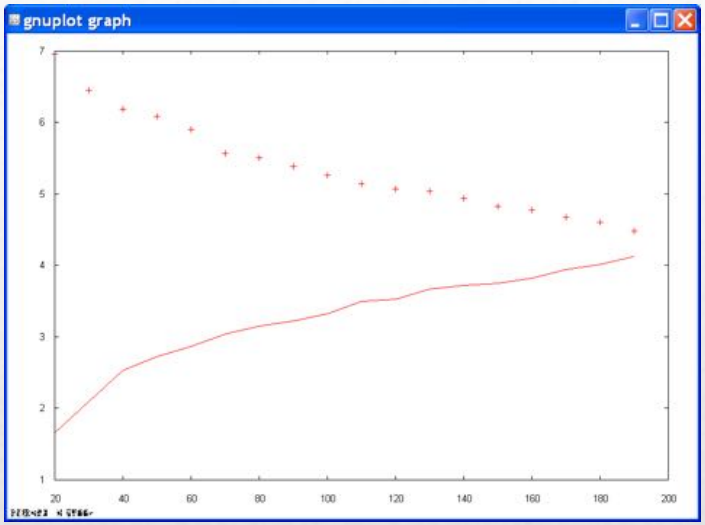
- Plus H est petit
  - Plus le biais est grand



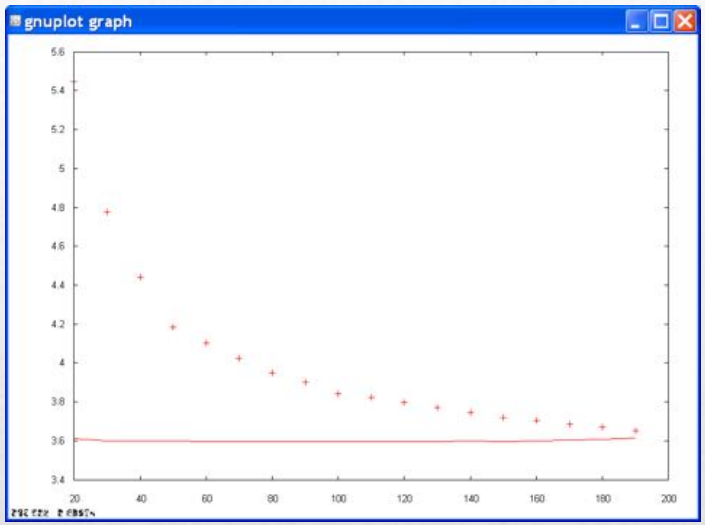
Exemple de comportement

Intervalles de confiance (95%)

Apprentissage



Test

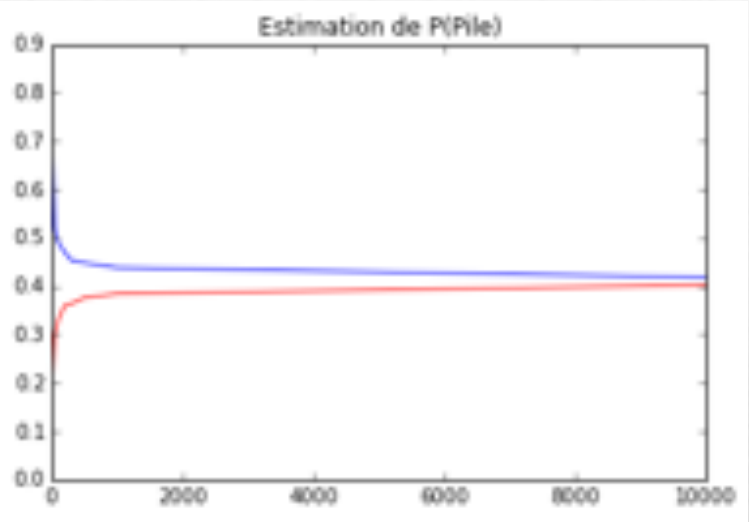


Erreurs de régression sur la base d'apprentissage et de test en fonction de la taille (#exemples apprentissage)

Données auto-mpg.datas (UCI) : 2 attributs

IC calculés à partir de Moyennes sur 100 expériences

Estimation de la probabilité de tomber sur pile d'une pièce biaisé



Estimation d'une statistique simple

ESTIMATION DE LA PERFORMANCE ET SELECTION DE MODELE

- Quel est le meilleur modèle de régression pour un jeu de données fixé ?
  - Structure du modèle
  - Paramètres du modèle

- Nécessité de bien estimer la performance
  - Pour une structure de modèle donnée
  - Pour le jeu de données disponible

⇒Découpage Train / Validation / Test

⇒Multiples découpages Train / Test (Cross Validation) pour :

⇒ estimation de la variance des résultats

⇒ Exploiter toutes les données en test

⇒Grid Search pour le réglage de ce qui ne s'optimise pas numériquement

Objectifs

Train / Validation / Test

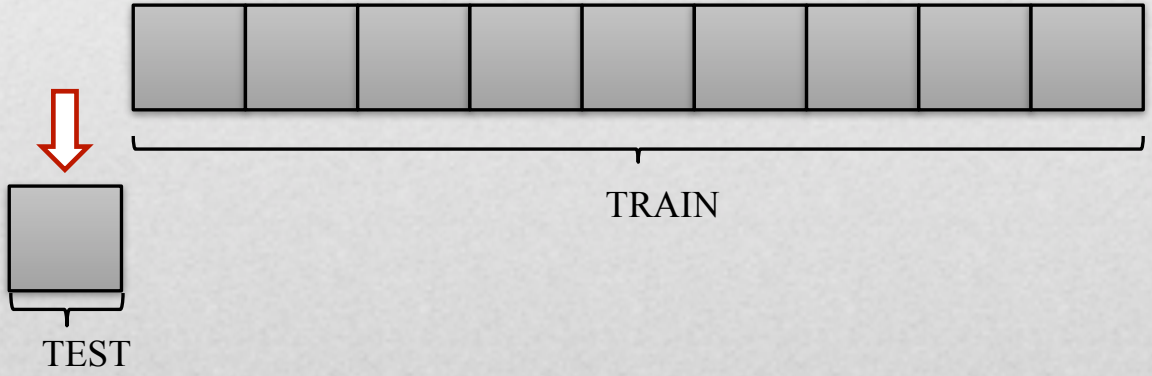
- Découpage des données en 3 parties
  - Training Set : pour l'optimisation des paramètres du modèle
  - Validation Set : pour le choix du meilleur modèle si on en teste plusieurs
  - Test Set : pour l'estimation de l'erreur en généralisation du meilleur modèle
- Mais
  - Limite la taille des données d'apprentissage : on apprend moins bien
  - Limite la taille des données d'évaluation : on évalue moins bien

Cross validation

Ensemble de données

⇒ N paires (Ensemble de TRAIN, Ensemble de TEST)

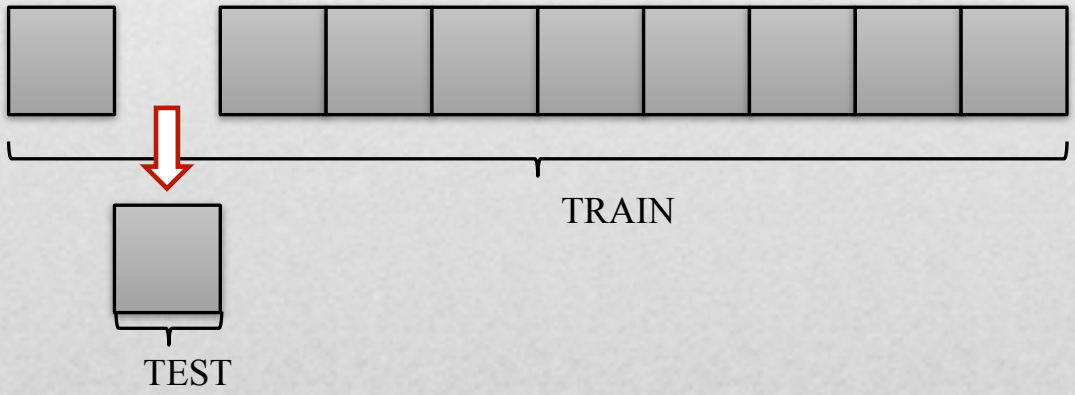
- En utilisant tour à tour chaque morceau comme ensemble de TEST
- Et le reste en TRAIN



Ensemble de données

⇒ N paires (Ensemble de TRAIN, Ensemble de TEST)

- En utilisant tour à tour chaque morceau comme ensemble de TEST
- Et le reste en TRAIN

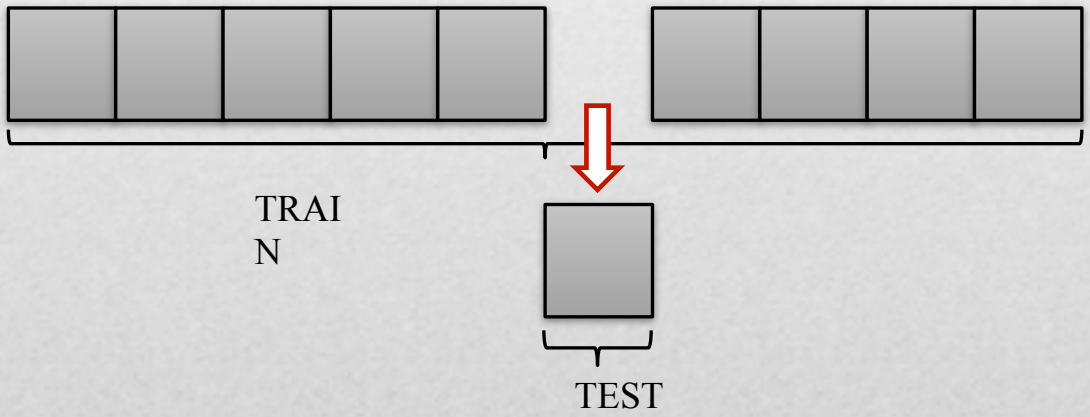


Cross validation

Ensemble de données

⇒ N paires (Ensemble de TRAIN, Ensemble de TEST)

- En utilisant tour à tour chaque morceau comme ensemble de TEST
- Et le reste en TRAIN



Cross validation

- Permet de récupérer une estimation de la variabilité de la performance en test
  - Hypothèse gaussienne sur la performance

  ⇒ Définition d'un intervalle de confiance à 95%

$$[\mu - Z_\alpha \sigma, \mu + Z_\alpha \sigma]$$

- Leave One Out
  - Cross Validation extrême
    - Autant de folds que d'exemples dans l'ensemble de données

  ⇒ Meilleure estimation de l'erreur

  ⇒ Très lourd
    - Autant d'apprentissage que d'exemples

Cross Validation et LOO

- Proche de la VC

- Tirage avec remise de B ensembles de Train de taille X (généralement la taille de l'échantillon), et test sur le restant

  ⇒Un exemple peut être présent plusieurs fois dans le TRAIN

- Idem pour l'estimation d'un intervalle de confiance sur la performance

Bootstrap

- Le modèle que l'on obtient par apprentissage résulte
  - Du choix a priori d'une famille de modèle
    - Et dans cette famille de certains choix « architecturaux » (e.g. degré d'une régression polynomiale)
  - De paramètres de l'algorithme d'optimisation
  - Des paramètres du modèle appris avec l'algorithme d'optimisation paramétré dans la famille défini par les choix architecturaux

- Seuls les derniers sont optimisables numériquement et automatiquement.
  - Pour les autres on optimise manuellement, par exemple en les testant exhaustivement…

⇒ On se définit un ensemble de paramètres avec pour chacun une liste de valeurs à tester.
⇒ On détermine une grille de jeux de paramètres à tester.
⇒ Pour chacun on calcule une estimation de la performance en généralisation
⇒ On sélectionne le meilleur modèle

GridSearching