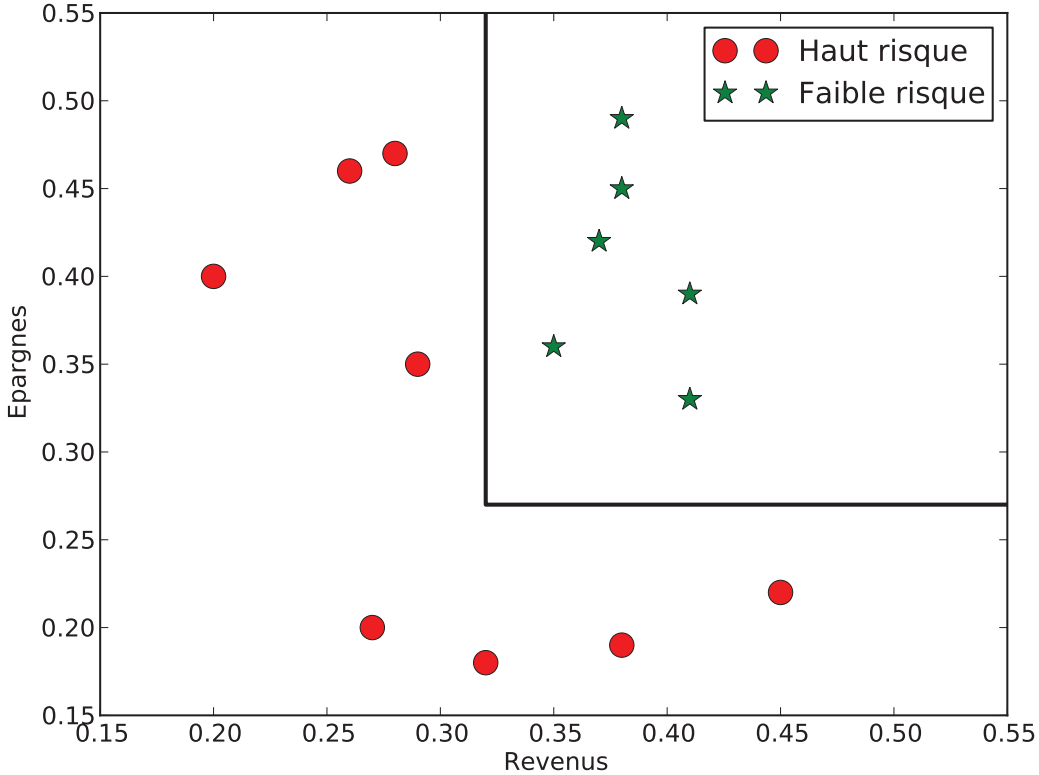
Pourquoi l'apprentissage automatique ?

- L'apprentissage automatique consiste à utiliser des ordinateurs pour **optimiser** un **modèle de traitement** de l'information selon certains **critères de performance** à partir d'**observations**, que ce soit des données-exemples ou des expériences passées.
- Lorsque l'on connaît le bon modèle de traitement à utiliser, pas besoin de faire de l'apprentissage !
- L'apprentissage automatique peut être utile lorsque :
  - On n'a pas d'expertise sur le problème (p. ex. robot navigant sur Mars)
  - On a une expertise, mais on ne sait pas comment l'expliquer (p. ex. reconnaissance de visages)
  - Les solutions au problème changent dans le temps (p. ex. routage de paquets)
  - Les solutions doivent être personalisées (p. ex. biométrie)

Exemple

- Une entreprise en crédit doit estimer automatiquement le niveau de risque de ses clients
- Mesures disponibles : revenus du client (variable x_1) et épargnes du client (variable x_2)
- Banque de données sur des clients pré-identifiés comme étant haut risque (rouge) ou à faible risque (vert)

Exemple



Si $x_1 > 0.32$ **et** $x_2 > 0.27$ **alors** *faible risque* **sinon** *haut risque*

Modèle et observations

- Objectif : inférer un **modèle de traitement général** à partir d'**observations spécifiques**
  - Le modèle inféré doit être une approximation correcte et utile des observations
- Les observations sont peu coûteuses et disponibles en quantité suffisantes ; la connaissance est coûteuse et rare
- Exemple en forage de données : relier des transactions de consommateurs à leurs comportements de consommation
  - Suggestions d'items similaires sur Amazon (livre, musique), Netflix (cinéma), etc.

Vues de l'apprentissage automatique

- Optimiser un modèle à partir d'observations selon un critère de performance
- **Vue statistique** : inférence à partir d'échantillons
- **Vue informatique** : construire des algorithmes et des représentations efficaces pour construire et évaluer les modèles
- **Vue ingénierie** : résoudre des problèmes sans devoir spécifier ou spécialiser les modèles

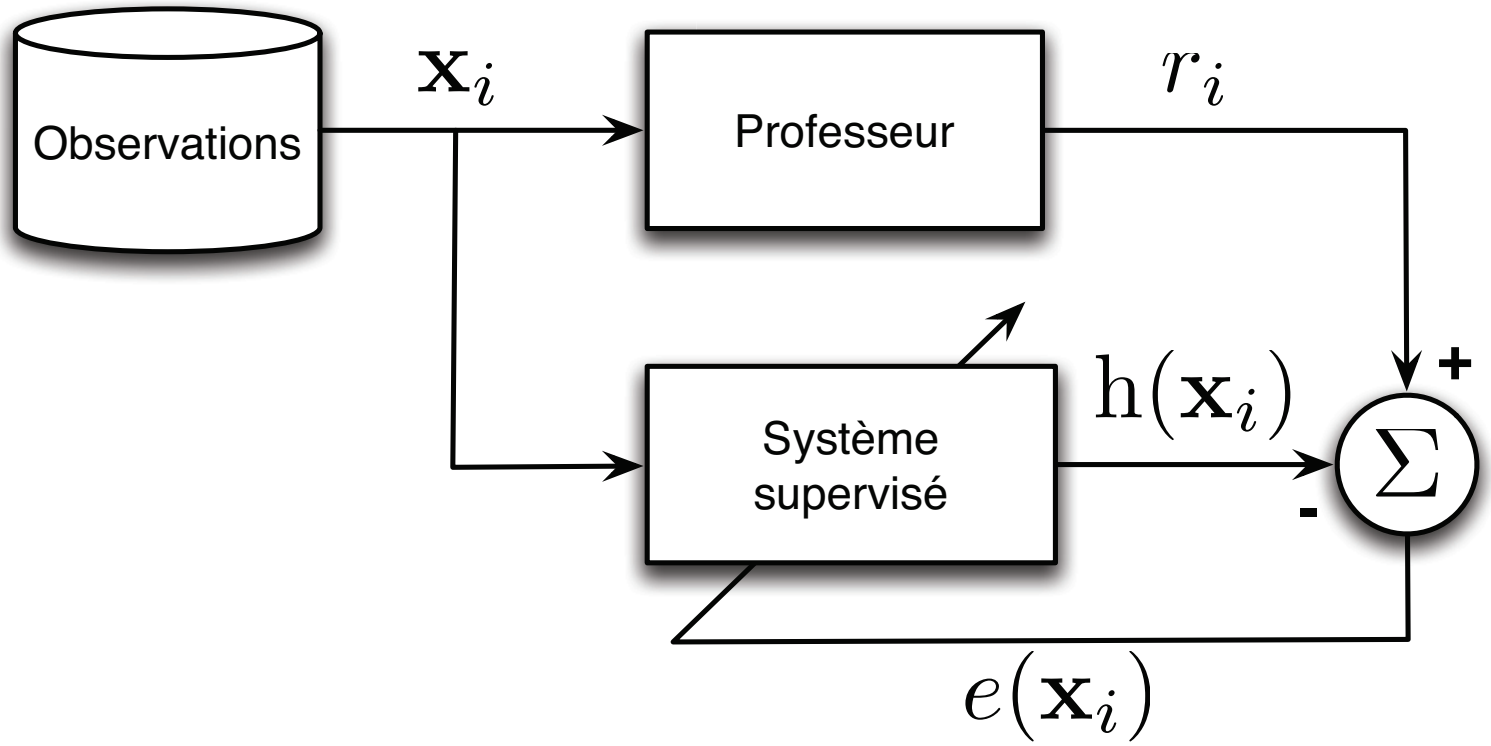
Apprentissage d'associations

- Analyse d'un panier d'épicerie
  - $P(Y|X)$ est la probabilité qu'une personne achetant X achète également Y, où X et Y sont des produits ou services
- Exemple : $P(croustilles|biere) = 0.7$

Apprentissage supervisé

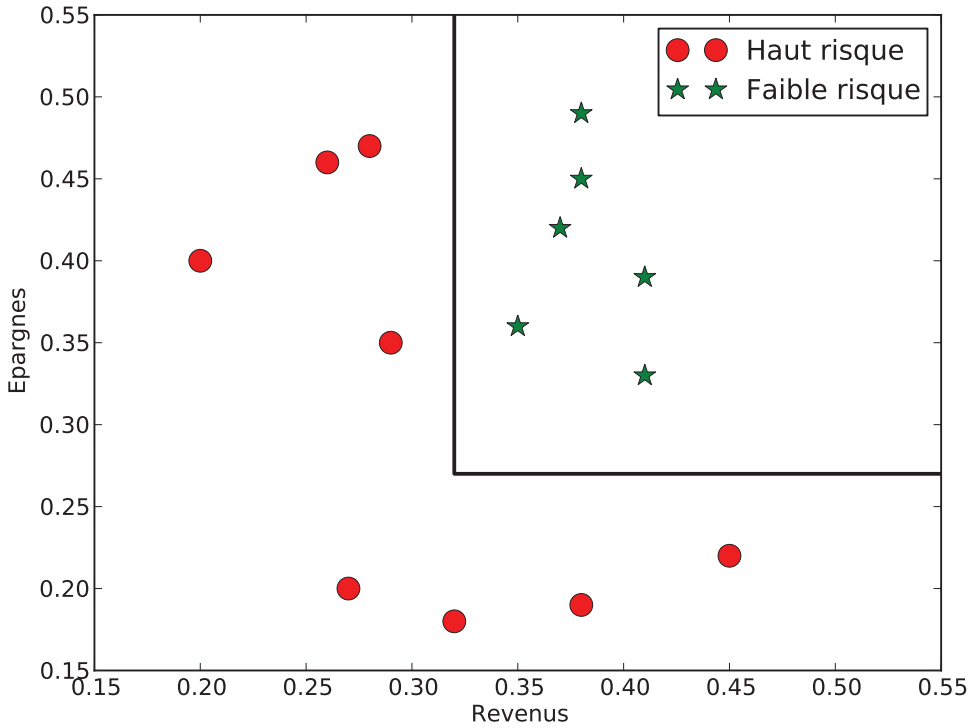
- Apprentissage supervisé
  - Objectif : apprendre une projection entre des observations X en entrée et des valeurs associées Y en sortie
- Modèlisation mathématique
  - $y = \mathrm{h}(x|\theta)$
  - $\mathrm{h}(\cdot)$: fonction générale du modèle
  - θ : paramètres du modèle

Schématisation de l'apprentissage supervisé



Classification

- Y est discret et correspond à des étiquettes de classes
- $\mathrm{h}(\cdot)$ est une fonction discriminante



Applications de la classification

- Reconnaissance des formes
  - Reconnaissance de visages : reconnaître les personnes malgré les variations (pose, éclairage, lunettes, maquillage, coiffure)
  - Reconnaissance de caractères manuscrits : reconnaître malgré différents styles d'écriture
  - Reconnaissance de la parole : dépendance temporelle de l'information, utiliser des dictionnaires de mots/structures valides
- Aide au diagnostic médical : déterminer les problèmes médicaux à partir des symptômes
- Extraction de connaissances : expliquer par des règles simples des masses de données
- Compression : représenter par des modèles compacts les données
- Détection d'irrégularités : identifier des fraudes, des intrusions

Reconnaissance de visages

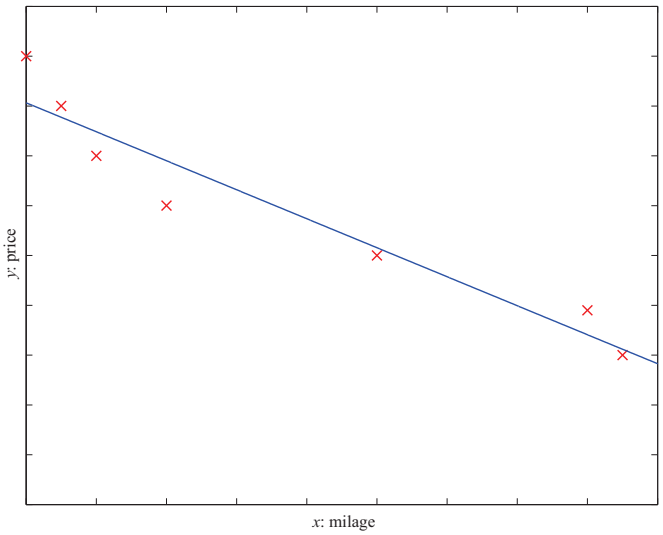
Tiré de la banque de visages ORL, `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`

Reconnaissance de caractères

Tiré de *Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, 86(11) :2278-2324, Novembre 1998.*

Régression

- Y est une valeur réelle
- $\mathrm{h}(\cdot)$ est la fonction de régression
- Exemple : prédiction du prix de vente d'une voiture usagée à partir du kilométrage parcouru
  - Observations : kilométrage parcourue (x)
  - Prédiction : prix de vente (y)
- Applications en prédiction de valeurs
  - Économie
  - Météorologie
- Applications en contrôle de haut-niveau
  - Volant de voiture (CMU NavLab)
  - Articulations de bras robotisés

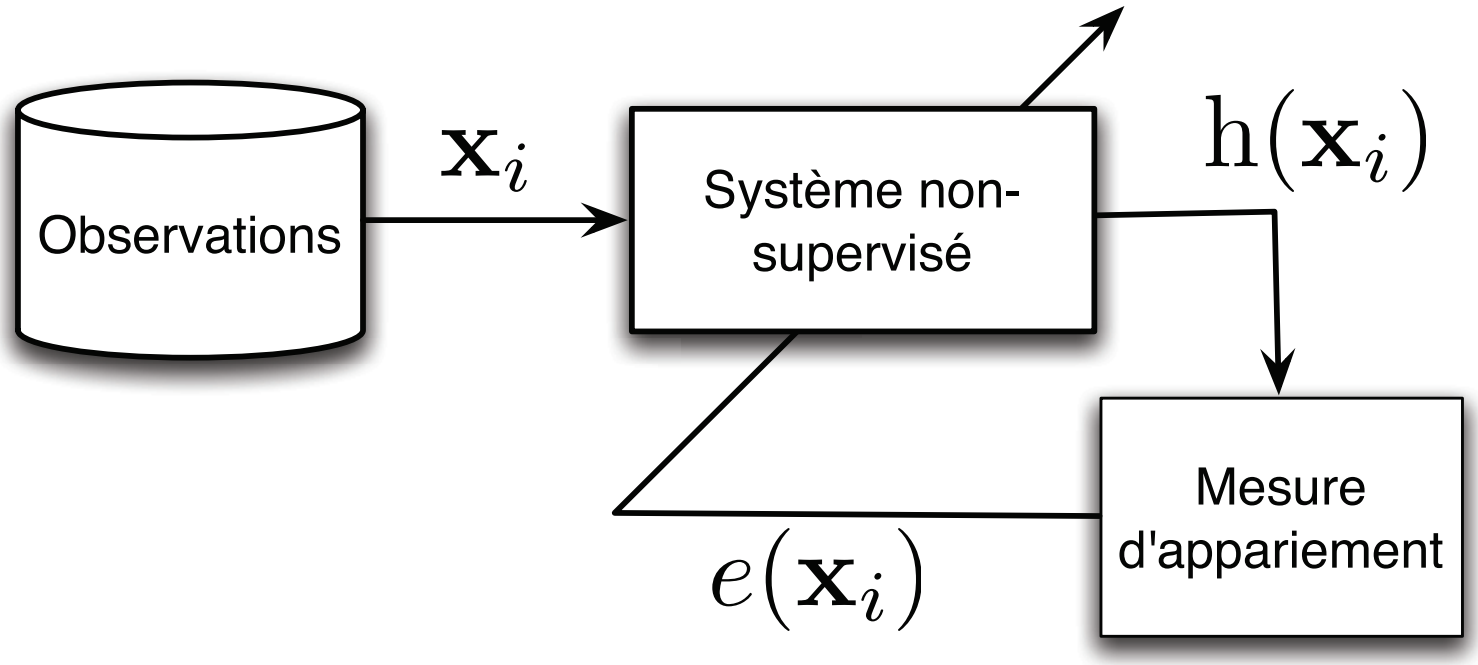


Tiré de *E. Alpaydin, Introduction to Machine Learning, 2e édition, MIT Press, 2010.*

Apprentissage non-supervisé

- Contrairement à l'apprentissage supervisé, il n'y a pas de valeurs de sortie
- Objectif : découvrir des régularités dans les observations
  - *Clustering* : découvrir des groupements d'observations similaires
- Applications
  - Segmenter les consommateurs dans les bases de données d'achats
  - Compression d'images : réduire le nombres de couleurs utilisées
  - Bio-informatique : découvrir des patrons dans l'ADN

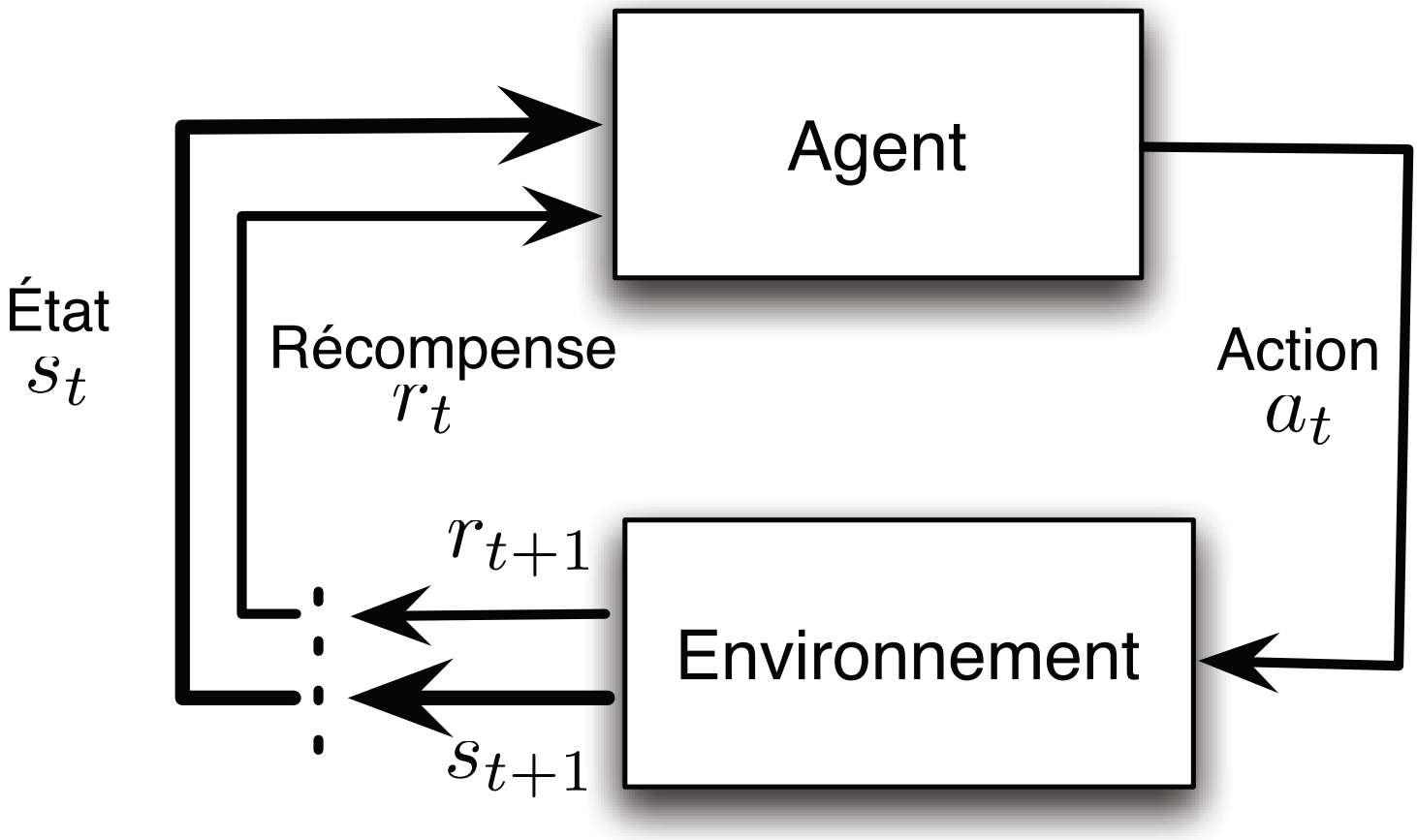
Schématisation de l'apprentissage non-supervisé



Apprentissage par renforcement

- Apprendre une politique : séquences d'actions
- L'apprentissage n'est pas supervisé, mais une récompense avec un délais est obtenue
- Problème d'assignation du crédit : quelles actions ont permis d'obtenir la récompense ?
- Applications
  - Jeux, avec un ou plusieurs participants
  - Robotique : navigation dans un environnement
  - Agents : prises de décision

Schématisation de l'apprentissage par renforcement



Banques de données (ressources)

- UCI Machine Learning Repository : `http://archive.ics.uci.edu/ml/`
- Statlib : `http://lib.stat.cmu.edu`
- MNIST Database of handwritten digits : `http://yann.lecun.com/exdb/mnist`

Revues scientifiques (ressources)

- Journal of Machine Learning Research : `http://jmlr.csail.mit.edu`
- Machine Learning : `http://www.springer.com/computer/artificial/journal/10994`
- IEEE Transactions on Pattern Analysis and Machine Intelligence : `http://www2.computer.org/portal/web/tpami/`
- Pattern Recognition : `http://www.elsevier.com/locate/pr`
- Neurocomputing : `http://www.elsevier.com/locate/neucom`

Conférences scientifiques (ressources)

- International Conference on Machine Learning (ICML)
- Neural Information Processing Systems (NIPS)
- European Conference on Machine Learning (ECML)
- International Joint Conference on Artificial Intelligence (IJCAI)
- Uncertainty in Artificial Intelligence (UAI)
- International Conference on Pattern Recognition (ICPR)

Introduction aux systèmes à apprentissage

Qu'est-ce que l'apprentissage ?

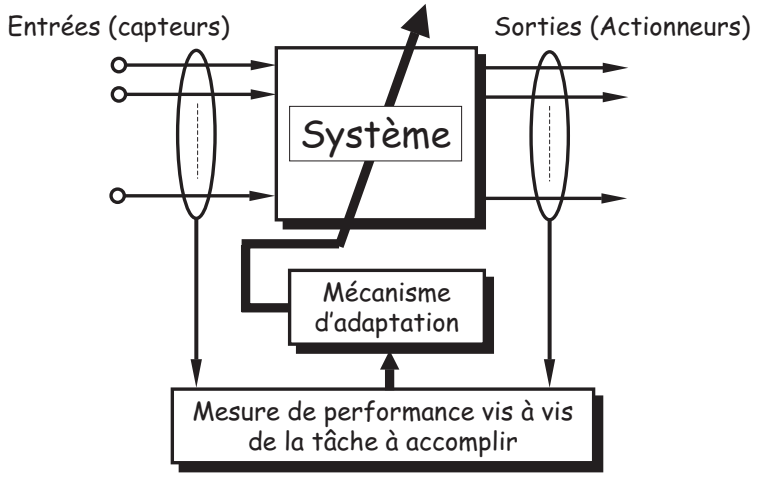
- **Mémoriser quelque chose**
- **Apprendre des faits au travers d'observations et d'explorations**
- **Amélioration sensori-motrice et/ou amélioration de ses connaissances par la pratique**
- **Organiser de nouvelles connaissances en un tout cohérent**

<u>Définition.</u> Un système est capable d'apprentissage s'il est doté d'un mécanisme d'adaptation permettant d'améliorer l'accomplissement d'une certaine tâche par répétition de celle-ci.

Introduction aux systèmes à apprentissage

Qu'est-ce que l'apprentissage ?

<u>Définition.</u> Un système est capable d'apprentissage s'il est doté d'un mécanisme d'adaptation permettant d'améliorer l'accomplissement d'une certaine tâche par répétition de celle-ci.



Introduction aux systèmes à apprentissage

Qu'est-ce que l'apprentissage ? - Induction et causalité

Un type d'apprentissage très courant est l'acquisition d'informations dans le but de faire des prédictions concernant le futur (prévisions).

<u>Principe de sens commun.</u> La répétition d'un même événement dans des circonstances identiques, nous conforte dans l'idée que cet événement se reproduira lorsque ces mêmes circonstances seront réalisées.

Chaque fois que l'on a constaté une hausse de la pression atmosphérique, cela s'est accompagné du beau temps.

Conclusion : la prochaine fois que je constaterai une hausse de pression atmosphérique je pourrai conclure à l'apparition du beau temps.

On retrouve cette idée que l'on apprend quelque chose de la répétition.

⇒ Croyance en la régularité des phénomènes : les mêmes causes produisent les mêmes effets

Introduction aux systèmes à apprentissage

Apprentissage et recherche d 'une fonction

Etant donné un ensemble d'exemples entrée/sortie, trouver une fonction permettant de représenter au mieux la relation entrée/sortie.

- Prononciation : fonction liant les lettres aux sons.
- Lecture de caractères manuscrits : fonction liant un ensemble de pixels aux lettres.
- Diagnostic : fonction liant les symptômes aux causes

Le problème de l'apprentissage d'une fonction implique 3 aspects :

- La mémorisation (base d 'apprentissage).
- La moyennisation (recherche des régularités).
- La généralisation (capacité à fournir la bonne réponse pour de nouveaux exemples).

Introduction aux systèmes à apprentissage

Différents types d 'apprentissage

Apprentissage supervisé (Supervised learning). Etant donné un ensemble d'exemples entrée/sortie, trouver une règle (une loi) permettant de prédire la sortie associée à une nouvelle entrée.

- Régression
- Classification

Apprentissage non supervisé (Unsupervised learning). Etant donné un ensemble d 'exemples non étiquetés (on ne connaît pas la sortie), grouper ces exemples par paquets (clusters) « cohérents ».

- Clustering

Apprentissage par renforcement (reinforcement learning). Le système interagit avec son environnement en exécutant des actions à partir d 'observations. En fonction des actions réalisées, le système est puni ou récompensé; il doit apprendre à choisir les actions permettant de récolter le plus grand nombre de récompenses.

Apprentissage supervisé - Régression

Sommaire

- **Position du problème**

- **Régression au sens des moindres carrés**
  - Algorithme LMS (méthode en ligne)
  - Formule des moindres carrés simples
  - Interprétation probabiliste

- **Le problème du surajustement → régularisation**
  - Interprétation probabiliste de la régularisation

- **Quelques autres méthodes**
  - Régression L1
  - Régression par noyau (kernel regression)
  - Régression linéaire localement pondérée

Apprentissage supervisé - Régression

Position du problème

Etant donné un ensemble d'exemples entrée/sortie : $\boxed{(x_i, y_i)_{i=1,\ldots,N}}$

$$x_i \in \mathbf{R}^n : x_i^T = [x_{i,1} \quad x_{i,2} \cdots x_{i,n}], \quad y_i \in \mathbf{R}, \quad i = 1, \ldots, N$$

on souhaite prédire y_{N+1} **pour la nouvelle donnée** x_{N+1} **:** $\boxed{\hat{y}_{N+1} = f(x_{N+1}, w)}$

Avec $f(\,.\,, w)$ **: Fonction de régression à choisir et à paramétrer (calcul de** w**).**

Apprentissage supervisé - Régression

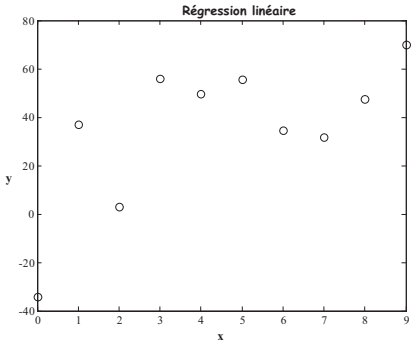
Régression linéaire (par rapport aux données et aux paramètres)

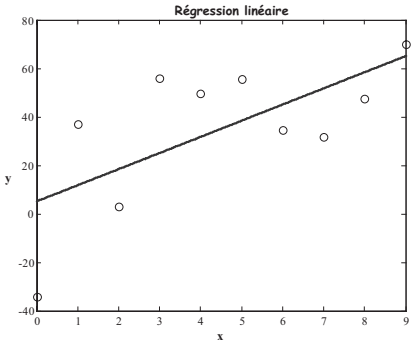
On souhaite estimer y_{n+1} **à l'aide d'une fonction linéaire des données :**

$$\boxed{\hat{y}_{N+1} = f(x_{N+1}, w) = w_0 + w_1 x_{N+1,1} + \cdots + w_n x_{N+1,n} = x_{N+1}^T w}$$

$$x_{N+1}^T = [1 \quad x_{N+1,1} \quad x_{N+1,2} \cdots x_{N+1,n}], \quad w^T = [w_0 \quad w_1 \quad w_2 \cdots w_n]$$

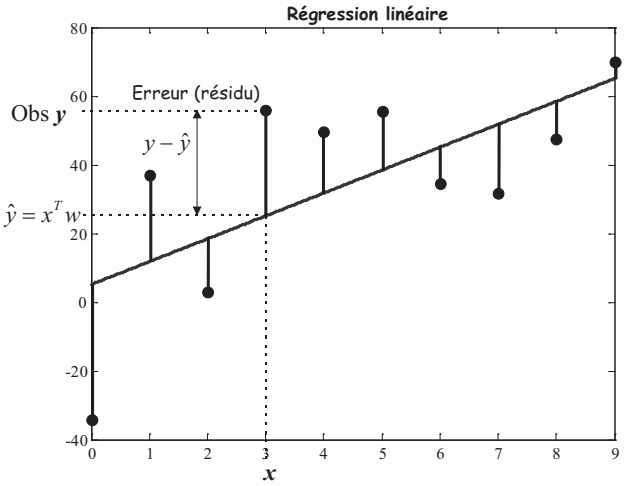
<u>Pb :</u> Calculer le vecteur des paramètres w permettant de réaliser au mieux l'estimation.





Apprentissage supervisé - Régression

Régression linéaire optimale au sens des moindres carrés



Coût à minimiser :

$$J(w) = \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2$$

Apprentissage supervisé - Régression

Méthode temps réel - Algorithme LMS

$$J(w) = \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2 = \sum_{i=1}^{N} J_i(W)$$

Méthode du gradient

$$w^{k+1} = w^k - \alpha\left(\frac{\partial}{\partial w}J(w)\right)_{w=w^k}$$

$$\frac{\partial}{\partial w}J(w) = \sum_{i=1}^{N}\frac{\partial}{\partial w}J_i(w) \quad \text{et} \quad \frac{\partial}{\partial w}J_i(w) = \frac{1}{2}\frac{\partial}{\partial w}(y_i - x_i^T w)^2$$

$$= -(y_i - x_i^T w)x_i$$

Algorithme LMS

$$w^{k+1} = w^k + \alpha(y_k - x_k^T w^k)x_k$$

Apprentissage supervisé - Régression

Méthode des moindres carrés simple

$$J(w) = \frac{1}{2}\sum_{i=1}^{N}(y_i - x_i^T w)^2$$

$$= \frac{1}{2}\|y - Xw\|_2^2 = \frac{1}{2}(y - Xw)^T(y - Xw)$$

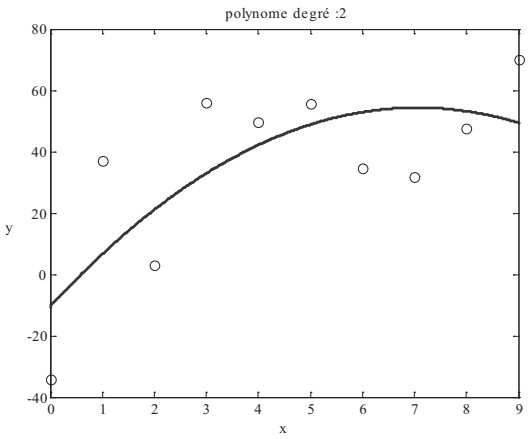
$$= \frac{1}{2}(y^T y - 2y^T Xw + w^T X^T Xw)$$

$$\frac{\partial}{\partial w}J(w) = -X^T y + X^T Xw \quad \rightarrow \quad \boxed{w = (X^T X)^{-1} X^T y}$$

$$X \in \mathbf{R}^{N\times n}: \quad X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_N^T \end{bmatrix}, \quad y \in \mathbf{R}^N: \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

Apprentissage supervisé - Régression

Régression non linéaire par rapport aux données - Exemple



$$\hat{y}_i = w_0 + w_1 x_i + w_2 x_i^2 = X_i^T w$$

$$X_i = \begin{bmatrix} 1 \\ x_i \\ x_i^2 \end{bmatrix}, \quad w = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

<u>Remarque :</u> La fonction de régression demeure linéaire par rapport aux paramètres.

Apprentissage supervisé - Régression

Régression non linéaire par rapport aux données

Etant donné un ensemble d'exemples entrée/sortie : $\boxed{(x_i, y_i)_{i=1,\ldots,N}}$

$$x_i \in \mathbf{R}^n : x_i^T = [x_{i,1} \quad x_{i,2} \cdots x_{i,n}], \quad y_i \in \mathbf{R}, \quad i = 1, \ldots, N$$

Soit : $X_i^T = \begin{bmatrix} 1 & f_1(x_i) & f_2(x_i) \cdots f_d(x_i) \end{bmatrix}$

Par exemple : $X_i^T = \begin{bmatrix} 1 & x_{i,1} & x_{i,2} & x_{i,1}^2 & x_{i,2}^2 & x_{i,1}x_{i,2} \end{bmatrix}$

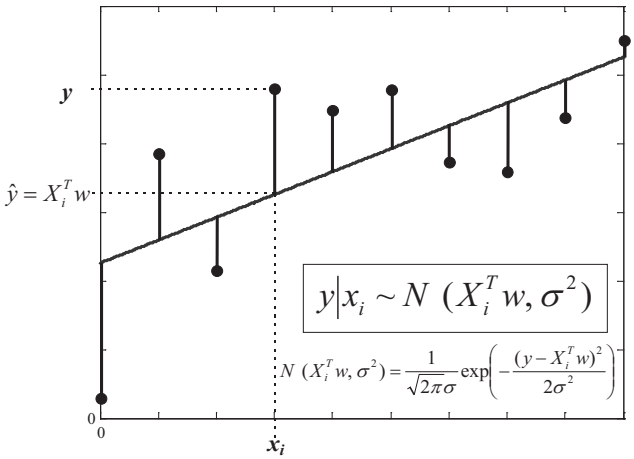
Soit : $X \in \mathbf{R}^{N \times d}$: $X = \begin{bmatrix} X_1^T \\ X_2^T \\ \vdots \\ X_N^T \end{bmatrix}$, $y \in \mathbf{R}^N$: $y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$

Le vecteur paramètre qui minimise $\|y - Xw\|_2^2$, s'écrit : $w = (X^T X)^{-1} X^T y$

Prédiction : $\hat{y}_{N+1} = X_{N+1}^T w$

Apprentissage supervisé - Régression

Interprétation probabiliste des MCS



Vraisemblance

$$L(w) = \prod_{i=1}^{N} \exp\left(-\frac{1}{2\sigma^2}(y - X_i^T w)^2\right)$$

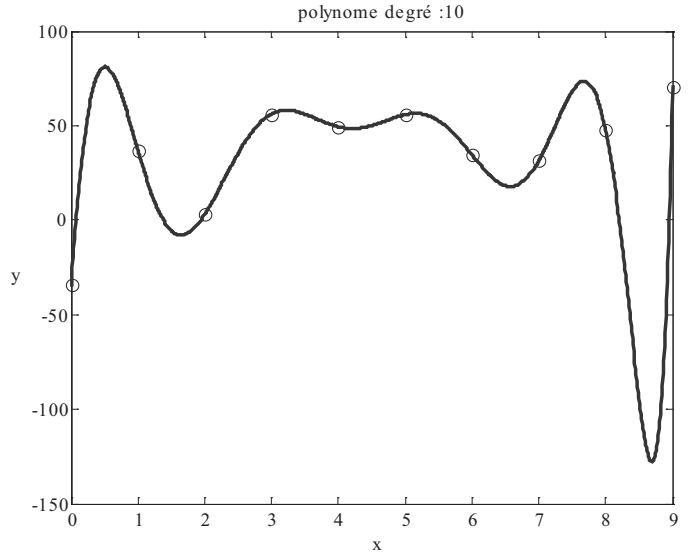
$$= \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(y - X_i^T w)^2\right)$$

Recherche de w au sens du maximum de vraisemblance = recherche de w au sens des moindres carrés :

$$w_{opt} = \arg\max_w L(w) = \arg\min_w J(w)$$

Apprentissage supervisé - Régression

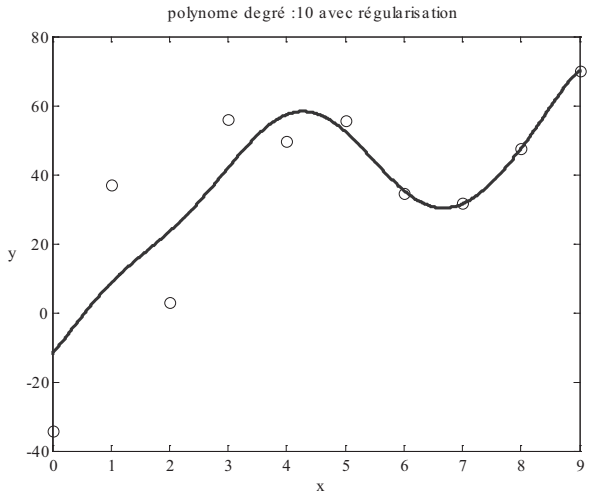
Le problème du sur-ajustement : régularisation



Le problème du sur-ajustement (sur-apprentissage) ne permet pas de généraliser

Apprentissage supervisé - Régression

Le problème du sur-ajustement : régularisation



Minimiser : $\frac{1}{2}\|y - Xw\|_2^2 + \varepsilon\|w\|_2^2$

Avec ε petit, la solution s 'écrit :

$$w = (X^T X + \varepsilon I)^{-1} X^T y$$

Apprentissage supervisé - Régression

Interprétation probabiliste de la régularisation

vraisemblance : $y|x_i \sim N\ (X_i^T w, \sigma^2) = \dfrac{1}{\sqrt{2\pi}\sigma} \exp\left(-\dfrac{(y - X_i^T w)^2}{2\sigma^2} \right)$

Information a priori : $w \sim N\ (0, \dfrac{\sigma^2}{\varepsilon}) = \dfrac{1}{\sqrt{2\pi}\sigma} \exp\left(-\dfrac{\varepsilon w^2}{2\sigma^2} \right)$

On veut maximiser l 'information a posteriori : $p(w|x_1, \ldots, x_n) = \dfrac{p(w, x_1, \ldots, x_n)}{p(x_1, \ldots, x_n)}$

$$\propto p(w, x_1, \ldots, x_n)$$

$$p(w, x_1, \ldots, x_N) = \exp\left(-\frac{\varepsilon}{2\sigma} \|w\|_2^2 \right) \prod_{i=1}^{N} \exp\left(-\frac{\varepsilon}{2\sigma} (y - X^T w)^2 \right)$$

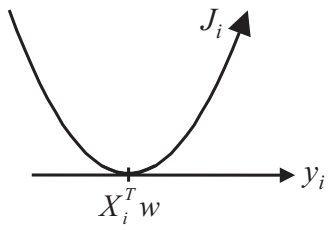
$$= \exp\left[-\frac{2}{2\sigma} \left(-\frac{\varepsilon}{2\sigma} \|w\|_2^2 + \sum_{i=1}^{N} (y - X^T w)^2 \right) \right]$$

Apprentissage supervisé - Régression

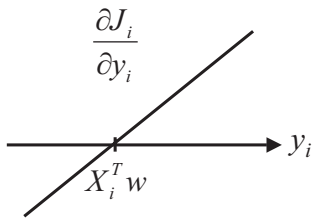
Sensibilité des MCS - Robustesse

$$J(w) = \sum_{i=1}^{N} (y_i - X_i^T w)^2$$

$$= \sum_{i=1}^{N} J_i$$

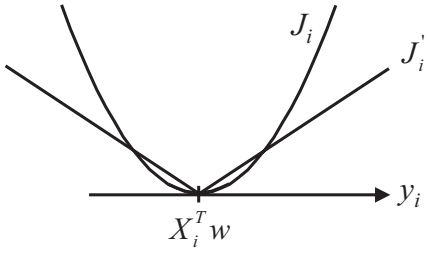


Sensibilité par rapport aux y_i



Apprentissage supervisé - Régression

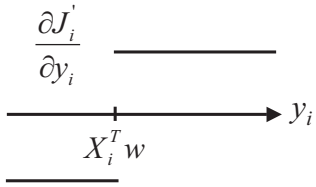
Régression L1 - Moindres valeurs absolues (MVA)



$$J'(w) = \sum_{i=1}^{N} \left| y_i - X_i^T w \right|$$

$$= \sum_{i=1}^{N} J'_i$$

Sensibilité par rapport aux y_i

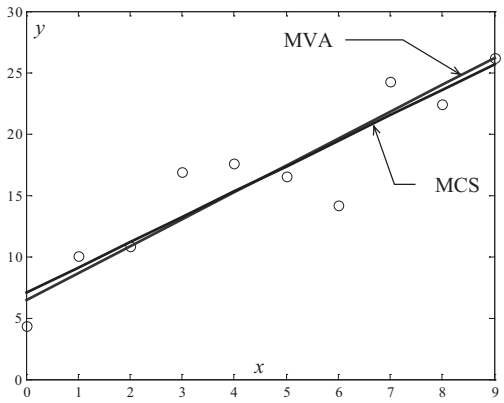


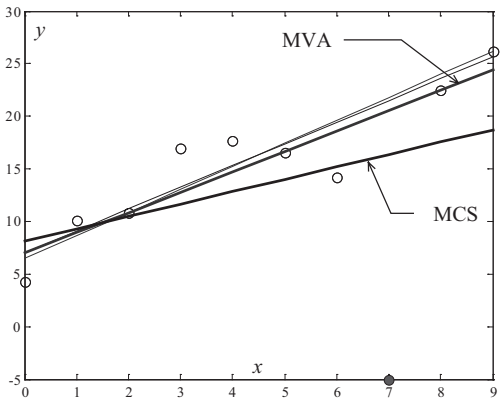
Résolution d'un problème d'optimisation linéaire (programmation linéaire)

$$\begin{cases} \min\limits_{w,\, c=[c_1 \cdots c_N]} \sum_{i=1}^{N} c_i \\ y_i - X_i^T w \le c_i \quad i = 1, \ldots, N \\ X_i^T w - y_i \le c_i \quad i = 1, \ldots, N \end{cases}$$

Apprentissage supervisé - Régression

Régression L1 - Moindres valeurs absolues (MVA)

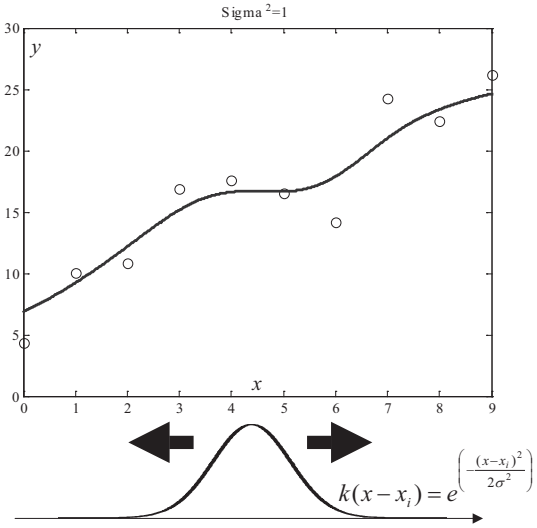




Apprentissage supervisé - Régression

Régression par noyau (Kernel regression)

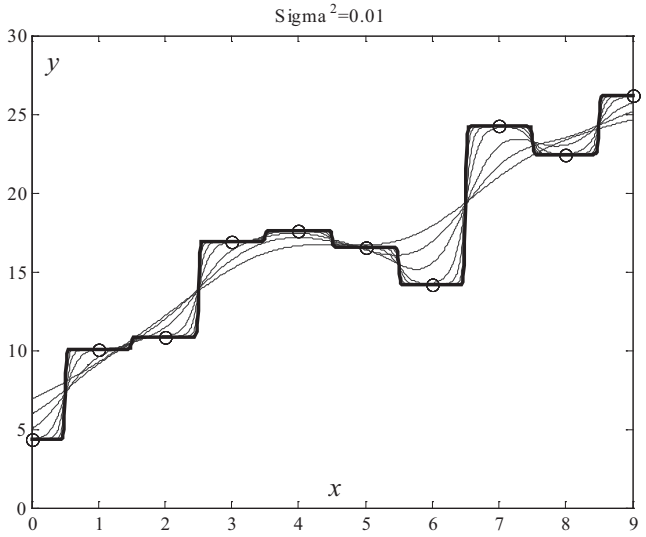
Méthode de régression non paramétrique : nécessite de disposer en permanence de la base de données



$$\hat{y}(x)=\frac{\sum_{i=1}^{N} y_i k(x-x_i)}{\sum_{i=1}^{N} k(x-x_i)}$$

Apprentissage supervisé - Régression

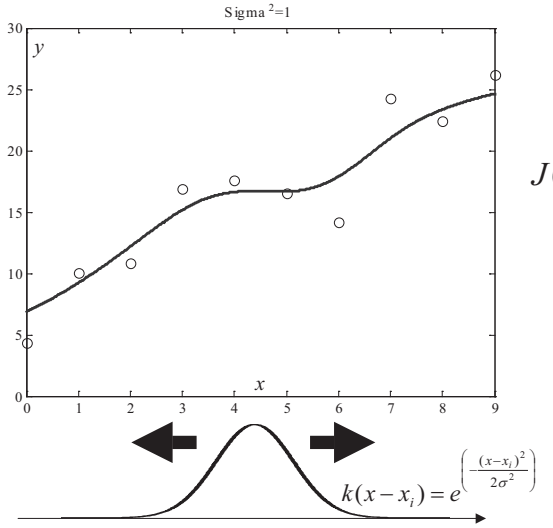
Régression par noyau (Kernel regression)



Apprentissage supervisé - Régression

Régression linéaire localement pondérée (Locally weighted linear regression (LWR))

Méthode de régression non paramétrique : nécessite de disposer en permanence de la base de données



Pour un point donné x on minimise le critère :

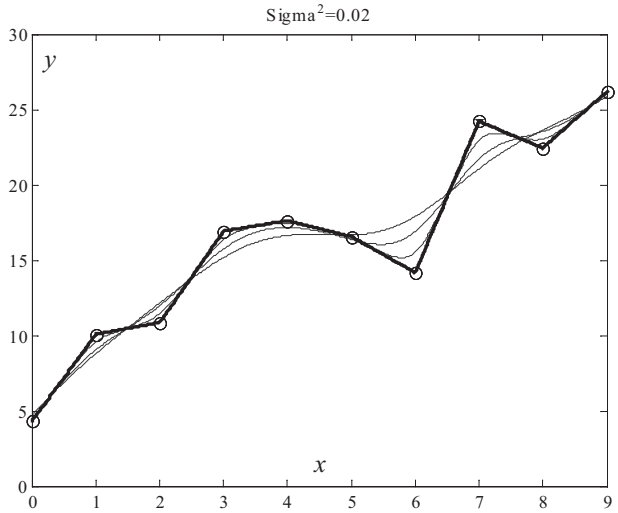
$$J(w,x)=\sum_{i=1}^{N}k(x-x_i)(y_i-X_i^T w)$$

$$w_{opt}(x)=\arg\min_{w} J(w,x)$$

$$\hat{y}(x)=X^T w_{opt}(x)$$

Apprentissage supervisé - Régression

Régression linéaire localement pondérée (Locally weighted linear regression (LWR))



Apprentissage supervisé - Classification
Sommaire

- **Position du problème**
- **Classification probabiliste**
  - Exemple introductif : le sélecteur aléatoire
  - Classification bayesienne optimale
  - Notion de rejet en classification
  - Paramétrage du classifieur : le cas gaussien
- **Classification neuronale**
- **Classification floue**

Apprentissage supervisé - Classification
Position du problème

Soit un ensemble de données réparties en k classes : $(x_i, y_i)_{i=1,\ldots,N}$

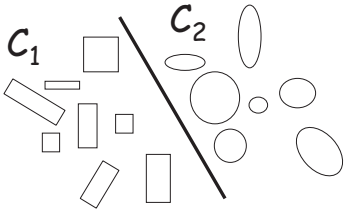
$$x_i \in \mathbf{R}^n : x_i^T = [x_{i,1} \quad x_{i,2} \cdots x_{i,n}], \quad y_i \in \{1, 2, \ldots, k\}, \quad i = 1, \ldots, N$$

on souhaite pouvoir associer une nouvelle donnée x_{N+1} **à l'une des classes.**

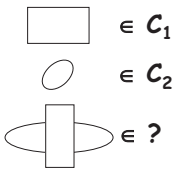
C'est un problème de décision : à quelle classe de données, la nouvelle donnée « ressemble » le plus ?

Différence avec la régression : les « sorties » sont ici discrètes, à chaque valeur discrète correspond une classe.

Base de données



Nouvelles données



Méthodes

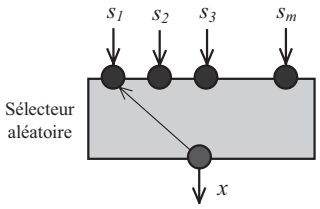
Approche probabiliste

Approche neuronale

Approche floue

Apprentissage supervisé - Classification

Approche probabiliste - Exemple introductif : le sélecteur aléatoire



On considère m sources de variables aléatoires dont on connaît les lois de densité de probabilité.

On note $p(x/Si)$ la loi de densité de probabilité de la va délivrée par la source Si

Le sélecteur aléatoire permet de sélectionner l'une des sources au gré du hasard

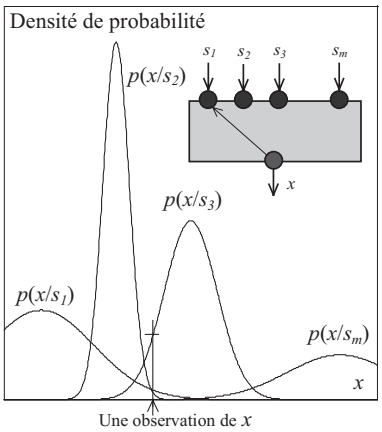
Lorsqu'une source est sélectionnée, on réalise une observation $x(t)$.
Problème : déterminer à partir de l'observation $x(t)$ la source sélectionnée

Deux cas :
- Sélection équiprobable des sources
- Sélection non équiprobable des sources

Apprentissage supervisé - Classification

Approche probabiliste - Exemple introductif : le sélecteur aléatoire

Sélection équiprobable des sources



La source sélectionnée correspond au plus grand des $p(x/S_i)$. La source S_i sélectionnée est telle que :

$$p(x/s_i) = \underset{j=1\ldots m}{Max}\, p(x/s_j)$$

Soit $P(Si/x)$ la probabilité pour que la source Si soit sélectionnée, l'observation x étant réalisée, on a :

$$\sum_{i=1}^{m} P(s_i/x) = 1 = \frac{\sum_{i=1}^{m} p(x/s_i)}{\sum_{j=1}^{m} p(x/s_j)} = \sum_{i=1}^{m} \frac{p(x/s_i)}{\sum_{j=1}^{m} p(x/s_j)}$$

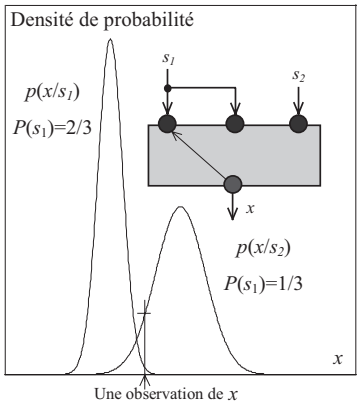
On a donc : $$P(s_i/x) = \frac{p(x/s_i)}{\sum_{j=1}^{m} p(x/s_j)}$$

Cette relation permet de calculer la probabilité pour que la source Si soit sélectionnée l'observation x étant réalisée. La source sélectionnée correspond à la plus grande des probabilités

Apprentissage supervisé - Classification

Approche probabiliste - Exemple introductif : le sélecteur aléatoire

Sélection non équiprobable des sources



Considérons deux sources S_1 et S_2 avec S_1 ayant deux fois plus de chance d'être sélectionnée que S_2.

$\Leftrightarrow$ Trois sources équiprobables dont deux identiques à S_1

La source S_1 a 2 chances sur 3 d'être sélectionnée
La source S_2 a 1 chance sur 3 d'être sélectionnée $\rightarrow \begin{cases} P(s_1) = 2/3 \\ P(s_2) = 1/3 \end{cases}$

$P(S_1)$ et $P(S_2)$ sont les probabilités a priori d'occurrence des sources.

Une mesure x étant réalisée, la source sélectionnée correspond au plus grand des deux nombres : $P(S_1)p(x/\ S_1)$ à $P(S_2)p(x/\ S_2)$.

Pour m sources, on a alors la règle de décision suivante :

$$P(s_i)\,p(x/s_i) = \underset{j=1\ldots m}{Max}\, P(s_j)\,p(x/s_j)$$

La probabilité d'occurrence de la source S_i l'observation x étant réalisée s'écrit alors :

$$P(s_i/x) = \frac{P(s_i)\,p(x/s_i)}{\sum_{j=1}^{m} P(s_j)\,p(x/s_j)}$$

C'est la formule de Bayes. Elle permet de calculer les probabilités a posteriori d'occurrence des sources, c'est à dire une fois l'observation réalisée.

Apprentissage supervisé - Classification

Approche probabiliste - Classification bayesienne optimale

L'exemple du sélecteur aléatoire n'est autre qu'un problème de classification.

Soit $x(t)$ le vecteur des observations et considérons m classes $C_1, C_2, \ldots, C_m$.

Soit $p(r/C_i)$ la loi de densité de probabilité de la variable x correspondant à la classe C_i.
Soit $P(C_i)$ la probabilité a priori d'occurrence de la classe C_i.

La probabilité d'occurrence de la classe C_i une observation r étant réalisée (probabilité a posteriori) s 'écrit :

$$P(\boldsymbol{C}_i/x) = \frac{P(\boldsymbol{C}_i)\,p(x/\boldsymbol{C}_i)}{\sum_{j=1}^{m} P(\boldsymbol{C}_j)\,p(x/\boldsymbol{C}_j)}\ ,\ i = 1\ldots m$$

L 'observation r est alors affectée à la classe C_i fournissant la plus grande probabilité a posteriori :

$$x(t) \in \boldsymbol{C}_i \ :\ P(\boldsymbol{C}_i/x) = \underset{j=1\ldots m}{Max}\, P(\boldsymbol{C}_j/x)$$

La probabilité d'une mauvaise classification s'écrit : $1 - \underset{j=1\ldots m}{Max}\, P(\boldsymbol{C}_j/x)$

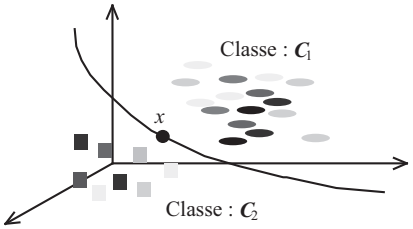
Elle est minimale, cette approche est donc optimale au sens d'une erreur de classification minimale

Apprentissage supervisé - Classification

Notion de rejet en classification

On distingue deux types de rejet : le rejet d'ambiguïté et le rejet en distance.

Le rejet d'ambiguïté se produit lorsqu'une observation se trouve sur la frontière séparant plusieurs classes ou un domaine commun à celles-ci.



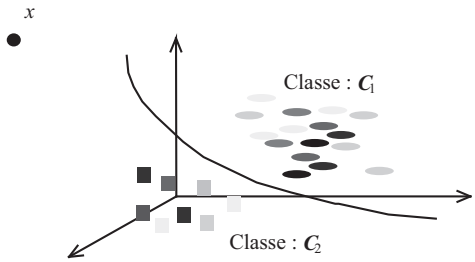
Décider dans ces conditions de l'appartenance de l'observation à l'une des classes plutôt qu'à l'autre constitue un risque d'erreur important. Soit R_a le seuil de rejet par ambiguïté, la règle de rejet s'écrit :

Si $e_{class} = 1 - \underset{j=1\ldots l}{Max}\, P(\boldsymbol{C}_j / x) > R_a$ *alors l'observation est rejetée par ambiguïté*

Apprentissage supervisé - Classification

Notion de rejet en classification

Le rejet en distance se produit lorsqu'une observation se trouve dans une zone éloignée de toutes les classes connues.



Cette situation peut signifier la nécessité de prendre en compte une nouvelle classe pour laquelle on ne dispose d'aucune information dans la base d'apprentissage. Soit R_d le seuil de rejet en distance, la règle de rejet s'écrit :

Si $\underset{j=1\ldots l}{Max}\, P(\boldsymbol{C}_j / x) < R_d$ *alors l'observation est rejetée en distance*

Apprentissage supervisé - Classification

Approche probabiliste - Paramétrage des classes : le cas gaussien

Dans les applications réelles, les lois de densité de probabilité des classes ne sont pas connues.

En pratique, l'hypothèse gaussienne est souvent adoptée. Chaque classe est modélisée par une loi de densité de probabilité gaussienne de dimension q (dimension de x) :

$$p(x/Ci) = \frac{1}{\sqrt{(2\pi)^q \det(\Sigma_i)}} \exp\left(-\frac{1}{2}(x-m_i)^T \Sigma_i^{-1}(x-m_i)\right)$$

avec m_i la moyenne de la classe C_i et Σ_i sa matrice de variance covariance. Ces deux paramètres peuvent êtres calculés à l'aide des estimateurs suivants :

$$\hat{m}_i = \frac{1}{N_i}\sum_{j=1}^{N_i} x_j, \quad \hat{\Sigma}_i = \frac{1}{N_i}\sum_{j=1}^{N_i}(x_j - m_i)(x_j - m_i)^T$$

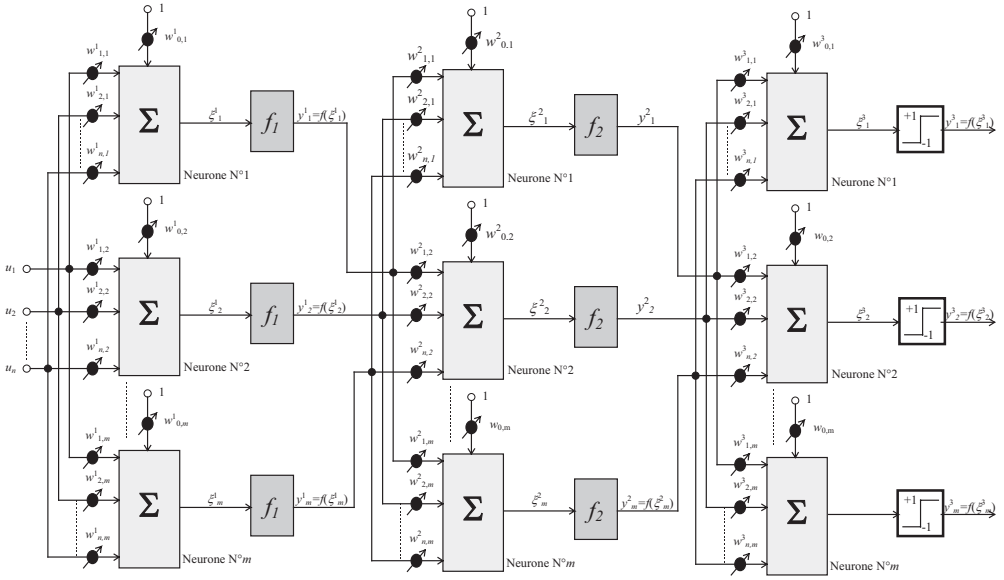
Les probabilités a priori, d'occurrences des classes $P(C_i)$, sont estimées à l'aide de la relation :

$$\hat{P}_i = \frac{N_i}{\sum_{j=1}^{l} N_j}$$

où N_i représente l'effectif de la classe C_i et/le nombre de classes connues.

Apprentissage supervisé - Classification

Approche neuronale

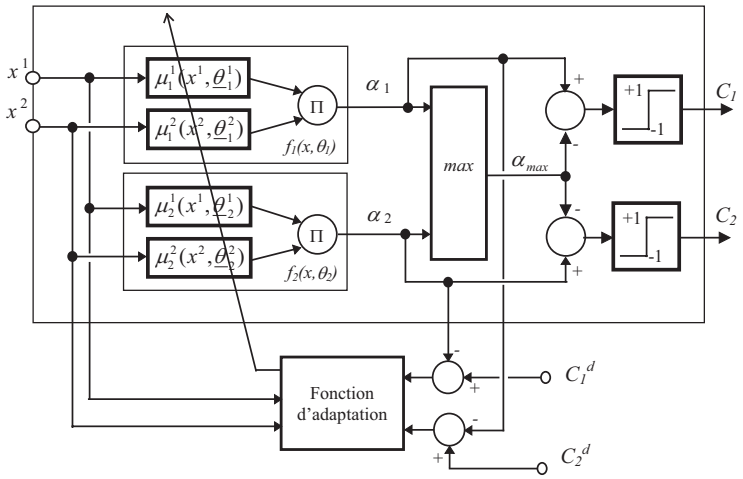


Apprentissage supervisé - Classification

Approche floue

$$\text{Si } \left(x^1 \text{ est } \widetilde{\mathbf{A}}_1^1\right) \text{et } \left(x^2 \text{ est } \widetilde{\mathbf{A}}_1^2\right) \text{ Alors la classe est } C_1$$

$$\text{Si } \left(x^1 \text{ est } \widetilde{\mathbf{A}}_2^1\right) \text{et } \left(x^2 \text{ est } \widetilde{\mathbf{A}}_2^2\right) \text{ Alors la classe est } C_2$$



Apprentissage non supervisé

Sommaire

- **Position du problème**

- **Analyse en composantes principales**
  - Motivations
  - Principe de l'ACP
  - Définition des composantes principales
  - Calcul de la 1ère composante principale
  - Calcul de la 2ème composante principale
  - Méthode générale
  - Exemple

- **Clustering (ou techniques de regroupement)**
  - Qu'est-ce que le clustering
  - Applications
  - Mesure de dissimilarité (similarité d'objets)
  - Méthode des K-moyennes (K-means)
  - Fuzzy k-means et algorithme de Gustafson-Kessel

Apprentissage non supervisé
Position du problème

• Rappelons que dans le cadre de la classification et de la régression, on dispose d 'un ensemble de couples entrée/sortie $(x_i, y_i)_{i=1,\ldots,N}$. L'objectif est de prédire la sortie y_{N+1} associée à une nouvelle donnée x_{N+1}. Ces méthodes sont dites à apprentissage supervisé (on connaît la réponse pour chaque entrée).

• Dans le cas des méthodes non supervisées, on dispose uniquement de données non étiquetées $(x_i)_{i=1,\ldots,N}$ (on ne connaît pas la réponse).

• <u>Pb :</u> que peut-on apprendre de données non étiquetées (unlabelled data) ?

Apprentissage non supervisé
Position du problème

• Généralement il est facile de collecter des données brutes, mais les étiquetées peut être une tâche d 'ampleur insurmontable.

• Les données peuvent être de grande dimension ($x_i \in \mathbb{R}^{1000}$), il est alors intéressant de rechercher une représentation de faible dimension permettant de décrire les données initiales sans trop de perte d'information.

• Pour tout traitement ultérieur, il est toujours utile de connaître comment sont structurées les données.

Les méthodes d'apprentissage non supervisé permettent de répondre à ces types de problème :

-Analyse en composantes principales
-Clustering

Apprentissage non supervisé
Analyse en composantes principales (ACP) - Motivations

L'analyse en composantes principales (ACP) permet de réduire la dimension des données initiales en minimisant la perte d 'information.

Le principe est de rechercher un nouvel ensemble de données de plus faible dimension qui conserve au mieux l'information contenue dans les données initiales (élimination des composantes fortement corrélées).

Par information, on entend le degré de corrélation entre les composantes des données. Deux composantes seront dites très informatives si elles sont non corrélées.

L 'analyse en composantes principales est une transformation linéaire permettant de ne retenir que les composantes non corrélées. Celles-ci sont appelées les composantes principales.

L'ACP peut être vue comme une rotation d 'axes telle que les données projetées sur les nouveaux axes soient de variance maximale. Le nouveau système d'axes est orthogonal.

Apprentissage non supervisé
Analyse en composantes principales (ACP) - Motivations

Caractéristiques de 24 modèles de voiture (*Source : L'argus de l'automobile, 2004*)

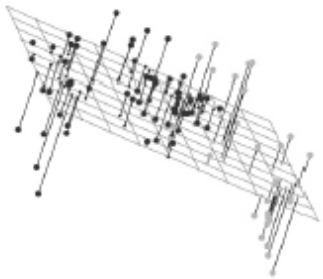
Modèle	*Cylindrée* (cm^3)	*Puissance* (*ch*)	*Vitesse* (*km/h*)	*Poids* (*kg*)	*Largeur* (*mm*)	*Longueur* (*mm*)
Citroën C2 1.1 Base	1124	61	158	932	1659	3666
Smart Fortwo Coupé	698	52	135	730	1515	2500
Mini 1.6 170	1598	170	218	1215	1690	3625
Nissan Micra 1.2 65	1240	65	154	965	1660	3715
Renault Clio 3.0 V6	2946	255	245	1400	1810	3812
Audi A3 1.9 TDI	1896	105	187	1295	1765	4203
Peugeot 307 1.4 HDI 70	1398	70	160	1179	1746	4202
Peugeot 407 3.0 V6 BVA	2946	211	229	1640	1811	4676
Mercedes Classe C 270 CDI	2685	170	230	1600	1728	4528
BMW 530d	2993	218	245	1595	1846	4841
Jaguar S-Type 2.7 V6 Bi-Turbo	2720	207	230	1722	1818	4905
BMW 745i	4398	333	250	1870	1902	5029
Mercedes Classe S 400 CDI	3966	260	250	1915	2092	5038
Citroën C3 Pluriel 1.6i	1587	110	185	1177	1700	3934
BMW Z4 2.5i	2494	192	235	1260	1781	4091
Audi TT 1.8T 180	1781	180	228	1280	1764	4041
Aston Martin Vanquish	5935	460	306	1835	1923	4665
Bentley Continental GT	5998	560	318	2385	1918	4804
Ferrari Enzo	5998	660	350	1365	2650	4700
Renault Scenic 1.9 dCi 120	1870	120	188	1430	1805	4259
Volkswagen Touran 1.9 TDI 105	1896	105	180	1498	1794	4391
Land Rover Defender Td5	2495	122	135	1695	1790	3883
Land Rover Discovery Td5	2495	138	157	2175	2190	4705
Nissan X-Trail 2.2 dCi	2184	136	180	1520	1765	4455

X_i = [cylindrée puissance vitesse poids largeur longueur]^T, i=1,…,24

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Principe de l 'ACP

Chaque visage est représenté par un vecteur de grande dimension $x \in \mathbf{R}^{361}$



$$\mathbf{x} \in \mathbb{R}^{361}$$

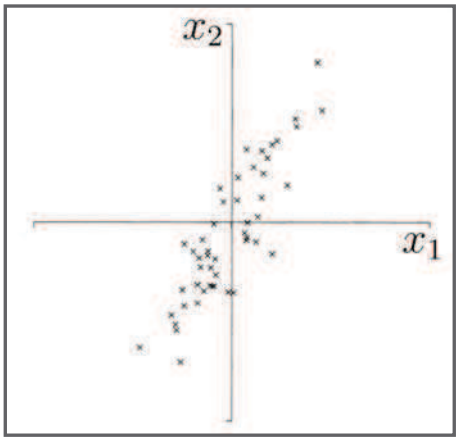
$$\mathbf{z} = \mathbf{U}^{\top}\mathbf{x}$$

$$\mathbf{z} \in \mathbb{R}^{10}$$

Comment choisir U ?

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Principe de l 'ACP



Echantillon de N observations dans un espace de dimension 2.

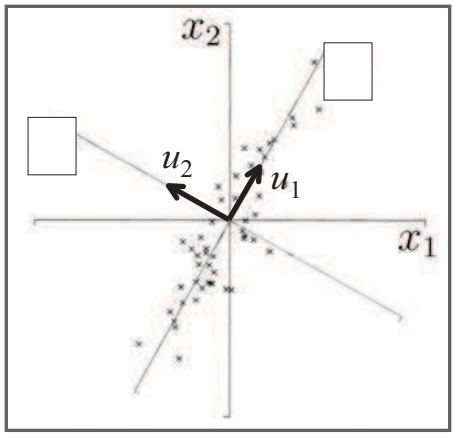
Les données sont des vecteurs d 'un espace multidimensionnel. Dans la suite on suppose les données centrées réduites.

$$X = [X_1 \quad \cdots \quad X_N],$$
$$X_i^T = [x_{i,1} \cdots x_{i,n}], \quad i = 1, \ldots, N$$

<u>Objectif :</u> projeter les données (les Xi) suivant des axes permettant de maximiser la variance des données.

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Principe de l 'ACP



La projection d 'un vecteur Xi sur un axe uk s 'écrit :

$$\text{proj}(X_i, u_k) = u_k^T X_i$$

La direction de plus grande variance est telle que $E((u_1^T X_i)^2)$ est maximale sur tout les X_i. La direction u1 est la première composante principale.

$$E((u_1^T X_i)^2) = \frac{1}{N}\sum_{i=1}^{N}(u_1^T X_i)^2 = \frac{1}{N}\sum_{i=1}^{N} u_1^T X_i X_i^T u_1$$

$$= u_1^T\left(\frac{1}{N}\sum_{i=1}^{N} X_i X_i^T\right)u_1 = u_1^T C_{XX} u_1$$

Cxx est la matrice de variance covariance des données (les Xi).

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Définition des composantes principales

Etant donné un ensemble de N observations d'un vecteur de dimension n

$$X = [X_1 \quad \cdots \quad X_N], \quad X_i^T = [x_{i,1} \cdots x_{i,n}], \quad i = 1, \ldots, N$$

On définit *la première composante principale* des N données, par la transformation linéaire

$$z_1 = u_1^T X, \quad z_1^T = [z_{1,1} \cdots z_{1,N}], \quad z_{1,i} = u_1^T X_i = \sum_{j=1}^{n} u_{1,j} x_{i,j}$$

Où le vecteur $u_1^T = [u_{1,1} \quad u_{1,2} \cdots u_{1,n}]$ est choisi de sorte que $E((u_1^T X_i)^2)$ soit maximum sur tout les Xi, et que u1 soit unitaire : $u_1^T u_1 = 1$

$$u_1 = \arg\max_{\|u\|=1} u^T C_{XX} u \Leftrightarrow u_1 = \arg\min_{\|u\|=1} -u^T C_{XX} u$$

$$C_{XX} = \left(\frac{1}{N}\sum_{i=1}^{N} X_i X_i^T\right)$$

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Définition des composantes principales

De la même façon on définit *la k^ième^ composante principale* des N données, par la transformation linéaire :

$$\boxed{z_k = u_k^T X, \quad z_k^T = \left[z_{k,1} \cdots z_{k,N}\right], \quad z_{k,i} = u_k^T X_i = \sum_{j=1}^{n} u_{k,j} x_{i,j}, \quad k = 1, \ldots, n}$$

Où le vecteur $u_k^T = \left[u_{k,1} \quad u_{k,2} \cdots u_{k,n}\right]$ est choisi de sorte que $E((u_k^T X_i)^2)$ soit maximum sur tout les Xi, sous les contraintes : $u_k^T u_k = 1$ et $u_k^T u_l = 0, \quad k > l \geq 1$

$$\boxed{\begin{cases} u_k = \arg\max\limits_{\|u\|=1} u^T C_{XX} u \\ u_k^T u_l = 0, \quad k > l \geq 1 \end{cases} \Leftrightarrow \begin{cases} u_k = \arg\min\limits_{\|u\|=1} -u^T C_{XX} u \\ u_k^T u_l = 0, \quad k > l \geq 1 \end{cases}}$$

$$C_{XX} = \left(\frac{1}{N} \sum_{i=1}^{N} X_i X_i^T \right)$$

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Calcul de la 1ère composante principale

$$\begin{cases} \max\limits_{u_1} u_1^T C_{XX} u_1 \\ u_1^T u_1 = 1 \end{cases} \Leftrightarrow \begin{cases} \min\limits_{u_1} -u_1^T C_{XX} u_1 \\ u_1^T u_1 = 1 \end{cases} \Leftrightarrow \min_{u_1} J(u_1), \quad J(u_1) = -u_1^T C_{XX} u_1 + \lambda(u_1^T u_1 - 1)$$

$$\frac{\partial}{\partial u_1} J(u_1) = -2C_{XX} u_1 + 2\lambda u_1 = 0 \rightarrow \boxed{(C_{XX} - \lambda I) u_1 = 0}$$

u1 est le vecteur propre de Cxx associé à la valeur propre $\lambda = \lambda_1$

On a maximisé : $u_1^T C_{XX} u_1 = u_1^T \lambda_1 u_1 = \lambda_1$

Par conséquent, λ_1 est la plus grande valeur propre de Cxx
λ_1 représente la variance des données projetées sur u1.

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Calcul de la 2ème composante principale

$$\begin{cases} \max\limits_{u_2} u_2^T C_{XX} u_2 \\ u_2^T u_2 = 1, \quad u_2^T u_1 = 0 \end{cases} \Leftrightarrow \begin{cases} \min\limits_{u_2} -u_2^T C_{XX} u_2 \\ u_2^T u_2 = 1, \quad u_2^T u_1 = 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} \min\limits_{u_2} J(u_2) \\ u_2^T u_1 = 0 \end{cases}, \quad J(u_2) = -u_2^T C_{XX} u_2 + \lambda(u_2^T u_2 - 1)$$

$$\frac{\partial}{\partial u_2} J(u_2) = -2C_{XX} u_2 + 2\lambda u_2 = 0 \rightarrow (C_{XX} - \lambda I) u_2 = 0$$

u2 est le vecteur propre de Cxx associé à la valeur propre $\lambda=\lambda_2$.

(Notons que la contrainte $u_2^T u1=0$ est satisfaite puisque u1 et u2 sont des vecteurs propres)

On a maximisé : $u_2^T C_{XX} u_2 = u_2^T \lambda_2 u_2 = \lambda_2, \quad (\lambda_2 \neq \lambda_1)$

Par conséquent, λ_2 est la deuxième plus grande valeur propre de Cxx.
λ_2 représente la variance des données projetées sur u2.

D 'une manière générale, la kème composante principale correspond au kème vecteur propre de Cxx associé à la kème plus grande valeur propre de Cxx.

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Méthode générale

Etant donné un ensemble de N observations d'un vecteur V de dimension n

$$V_1, V_2, \ldots V_N \quad V_i^T = [v_{i,1} \cdots v_{i,n}], \quad i = 1, \ldots, N$$

1. Construire la matrice X des données normalisées

$$X = \begin{bmatrix} X_1 & \cdots & X_N \end{bmatrix}, \quad X_i^T = [x_{i,1} \cdots x_{i,n}], \quad i = 1, \ldots, N$$

$$x_{i,1} = \frac{v_{i,1} - m_1}{\sigma_1}, \quad x_{i,2} = \frac{v_{i,2} - m_2}{\sigma_2}, \ldots, x_{i,n} = \frac{v_{i,n} - m_n}{\sigma_n}$$

$$\begin{cases} m_1 = \frac{1}{N} \sum_{i=1}^{N} x_{i,1} \\ \sigma_1 = \frac{1}{N-1} \sum_{i=1}^{N} (x_{i,1} - m_1)^2 \end{cases}, \begin{cases} m_2 = \frac{1}{N} \sum_{i=1}^{N} x_{i,2} \\ \sigma_2 = \frac{1}{N-1} \sum_{i=1}^{N} (x_{i,2} - m_2)^2 \end{cases}, \ldots, \begin{cases} m_n = \frac{1}{N} \sum_{i=1}^{N} x_{i,n} \\ \sigma_n = \frac{1}{N-1} \sum_{i=1}^{N} (x_{i,n} - m_n)^2 \end{cases}$$

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Méthode générale

Etant donné la matrice X des données normalisées

$$X = \begin{bmatrix} X_1 & \cdots & X_N \end{bmatrix}, \quad X_i^T = [x_{i,1} \cdots x_{i,n}], \quad i = 1, \ldots, N$$

2. Calculer la matrice de variance covariance Cxx de X :

$$C_{XX} = \frac{1}{N-1} XX^T = \left(\frac{1}{N-1} \sum_{i=1}^{N} X_i X_i^T \right)$$

3. Calculer les valeurs propres de Cxx : $\lambda_1 > \lambda_2 > \ldots \lambda_{n-1} > \lambda_n$.

4. Calculer les vecteurs propres associés aux valeurs propres λ_1 à λ_n : $u_1, u_2, \ldots, u_n$ et construire la matrice :

$$U = \begin{bmatrix} u_1^T \\ \vdots \\ u_n^T \end{bmatrix}, \quad u_i^T = [u_{i,1} \cdots u_{i,n}], \quad i = 1, \ldots, n$$

Notons que U n 'est autre qu'une matrice de changement de base

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Méthode générale

3. Réduction de la dimension des données. Construire la matrice Ur correspondant aux r vecteurs propres associés aux r plus grandes valeurs propres de Cxx : $\lambda_1 > \ldots \lambda_k$.

$$U_r = \begin{bmatrix} u_1^T \\ \vdots \\ u_r^T \end{bmatrix}, \quad u_i^T = [u_{i,1} \cdots u_{i,n}], \quad i = 1, \ldots, r$$

5. Calculer le nouveau jeu de données (dimension réduites) :

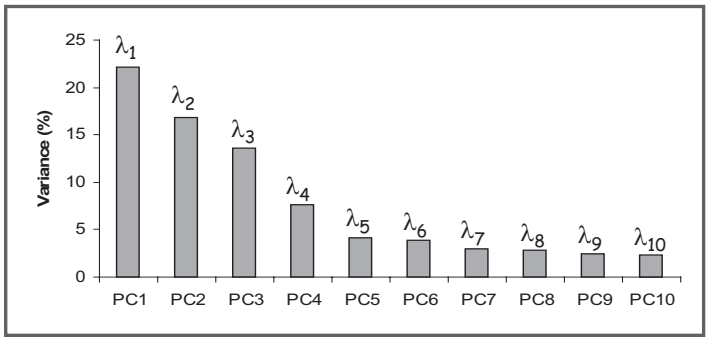
$$Z = \begin{bmatrix} z_1 & z_2 \cdots z_N \end{bmatrix} = U_r X, \quad z_i^T = [z_{i,1} \cdots z_{i,r}] \in \mathbf{R}^r, \quad i = 1, \ldots, N$$

Reconstruction des données initiales : $\hat{X} = U^T Z$

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Choix du nombre de composantes

On élimine les composantes de moindre variance.



On ne perd pas trop d'information si les valeurs propres associées aux composantes éliminées sont faibles relativement à celles que l'on conserve :

- Données initiales de dimension n
- On calcule n valeurs propres et vecteurs propres
- On choisit les r vecteurs propres correspondant aux r plus grandes valeurs propres
- Les données finales sont de dimension r

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Exemple

- d = number of pixels
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a face image
- $\mathbf{x}_{ji}$ = intensity of the j-th pixel in image i

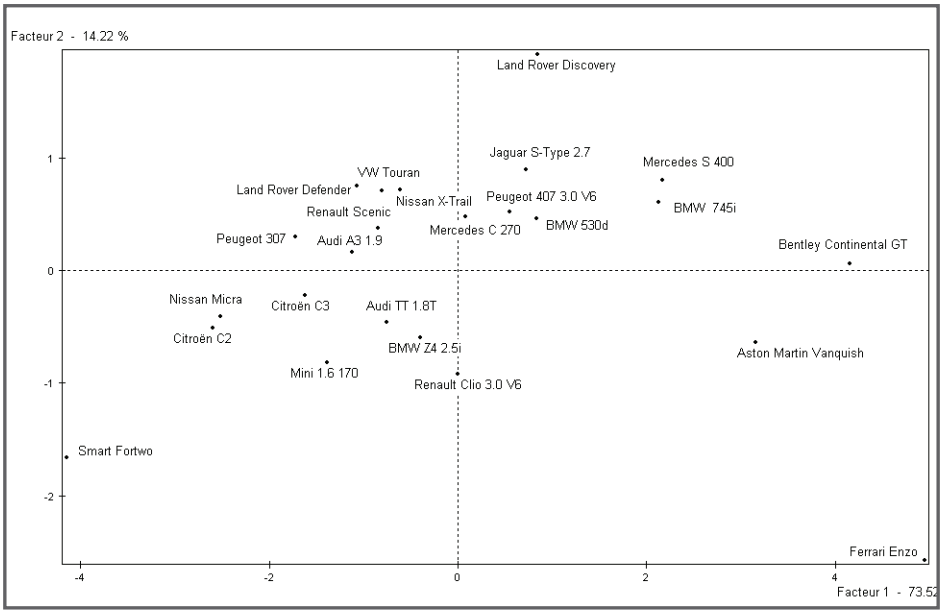
$$\mathbf{X}_{d\times n} \approx \mathbf{U}_{d\times r} \mathbf{Z}_{r\times n}$$

$\mathbf{z}_i$ more "meaningful" representation of faces than $\mathbf{x}_i$.

Use $\mathbf{z}_i$ for nearest-neighbor classification

Apprentissage non supervisé

Analyse en composantes principales (ACP) - Exemple



Apprentissage non supervisé

Clustering - Sommaire

- **Clustering (ou techniques de regroupement)**
  - Qu'est-ce que le clustering
  - Applications
  - Mesure de dissimilarité (similarité d'objets)
  - Méthode des K-moyennes (K-means)
  - Fuzzy k-means et algorithme de Gustavson-Kessel

Apprentissage non supervisé

Clustering (regroupement) - Qu'est-ce que le clustering

Le clustering consiste en une organisation de données en groupes (clusters) de façon que les données appartenant à un même groupe soient fortement similaires.

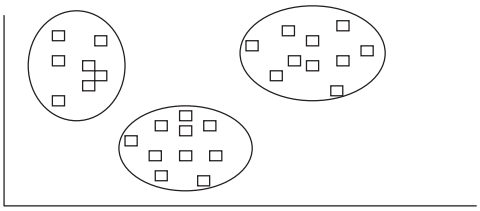
• Définir une mesure de similarité (dissimilarité) entre objets (données).

• Construire un algorithme permettant de créer des groupes de similarité tels que:

- La mesure de similarité des éléments dans un même groupe soit maximale.

- La mesure de similarité des éléments appartenant à des groupes distincts soit minimale.

Le clustering permet d 'avoir une bonne vision de la manière dont sont structurées les données



Apprentissage non supervisé

Clustering (regroupement) - Applications

- Segmentation
- Classification (pour trouver les classes)
- Recherche d 'information (Web)
- Compression de données
- Modélisation
- etc

Apprentissage non supervisé
Clustering (regroupement) - Mesure de dissimilarité

Soit X la matrice des données :

$$X = [X_1 \quad \cdots \quad X_N], \quad X_i^T = [x_{i,1} \cdots x_{i,n}], \quad i = 1, \ldots, N$$

Un choix courant de la mesure de dissimilarité entre deux données Xi et Xj, est la distance euclidienne :

$$D(X_i, X_j) = \|X_i - X_j\|_2 = \sqrt{\sum_{l=1}^{n} (x_{i,l} - x_{j,l})^2}$$

Les clusters définis à l'aide de la distance Euclidienne sont invariants par translations ou rotations dans l'espace de représentation, cela n'est pas vrai dans le cas d'une mise à l'échelle des données

$\Rightarrow$ il n 'est pas toujours désirable d 'effectuer une normalisation des données.

Apprentissage non supervisé
Clustering (regroupement) - Méthode des k-moyennes

- Les données sont représentées à l 'aide de K clusters (groupes)

- Chaque groupe est défini par un prototype pk représentant la moyenne des éléments du groupe (centre du groupe)

- Chaque donnée est affectée à un seul groupe. On peut définir la fonction d'appartenance μ_{ik} de la donnée n°i au cluster n° k :

$$\mu_{ik} \in \{0,1\}, \quad \sum_{k=1}^{K} \mu_{ik} = 1 \quad \forall i$$

Exemple : 4 données et 3 clusters :

$$[\mu_{ik}] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{array}{l} X_1 \in \text{cluster } k = 1 \\ X_2 \in \text{cluster } k = 3 \\ X_3 \in \text{cluster } k = 2 \\ X_4 \in \text{cluster } k = 3 \end{array}$$

<u>Remarque :</u> les prototypes pk sont inconnus ainsi que les fonctions d 'appartenance.

Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes

Calcul des prototypes pk et des fonctions d 'appartenance μ_{ik}

On minimise la fonction coût :

$$\begin{cases} J = \sum_{i=1}^{N} \sum_{k=1}^{K} \mu_{ik} \| X_i - p_k \| \\ \sum_{k=1}^{K} \mu_{ik} = 1 \quad \forall i \end{cases}$$

1. Choisir le nb K de clusters et initialisation des prototypes pk, k=1,…,K

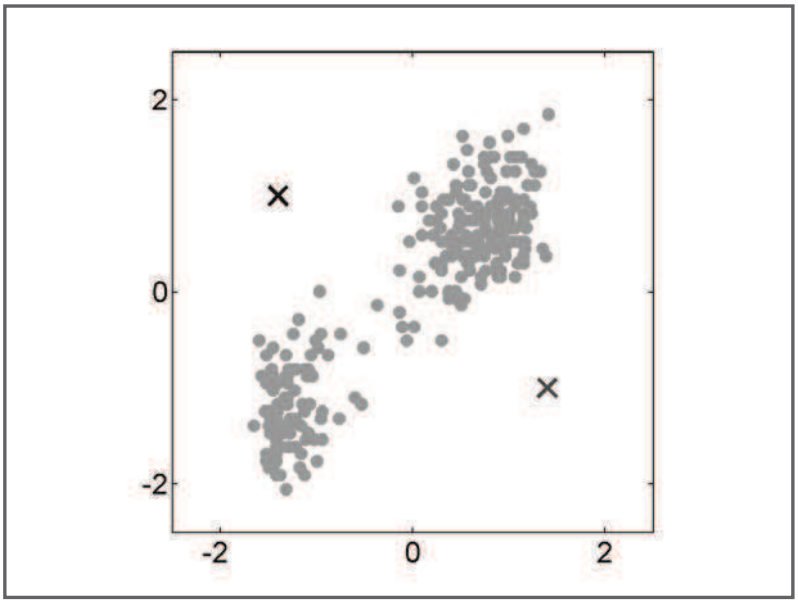
2. Minimiser J par rapport aux μ_{ik} → affecter chaque donnée Xi à l 'unique prototype le plus proche pk.

3. Minimiser J par rapport aux pk → $p_k = \sum_{i=1}^{N} \mu_{ik} X_i \Big/ \sum_{i=1}^{N} \mu_{ik}$

4. Tant que J>e aller en 2

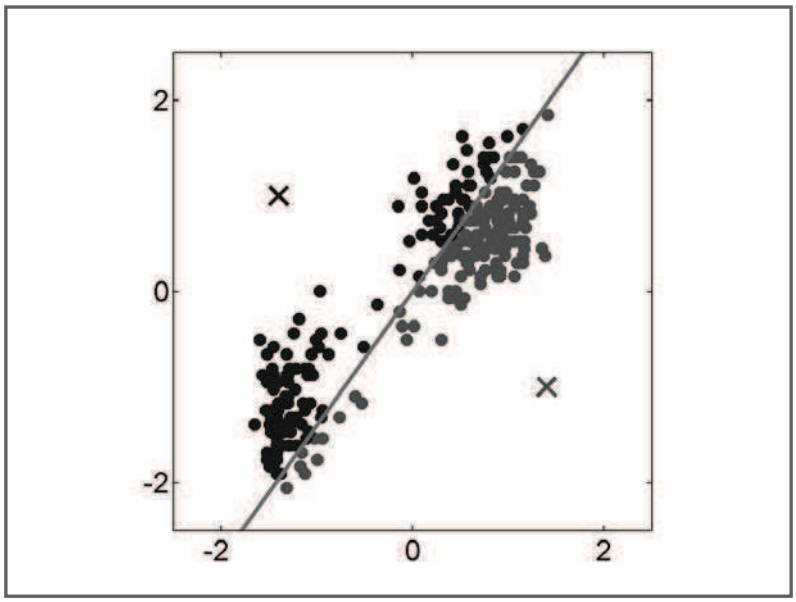
Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



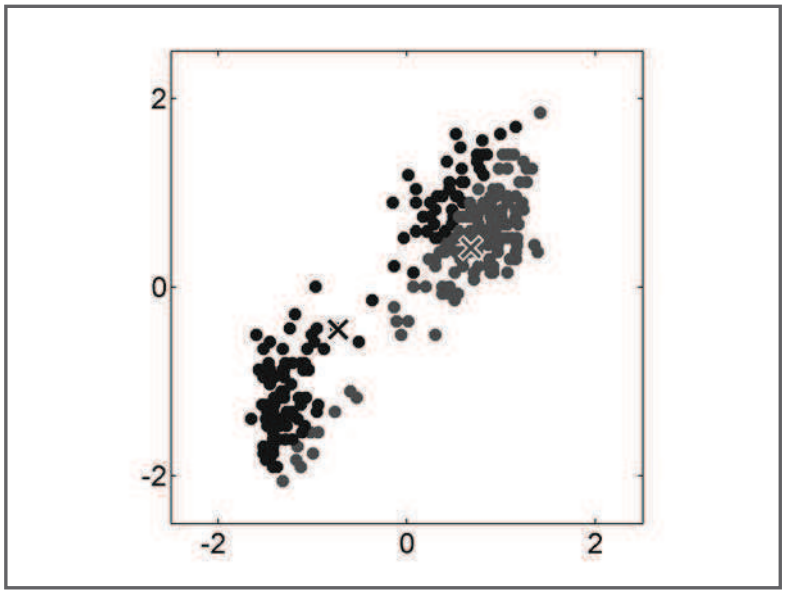
Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



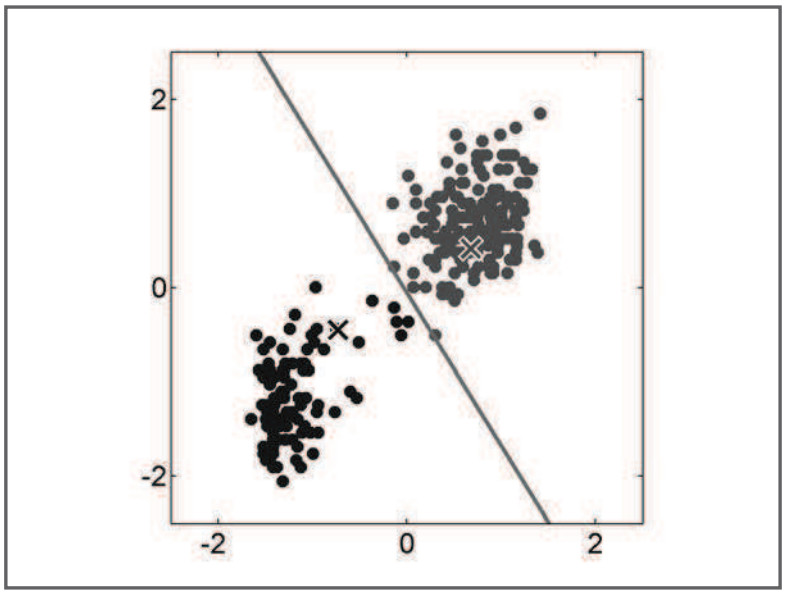
Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



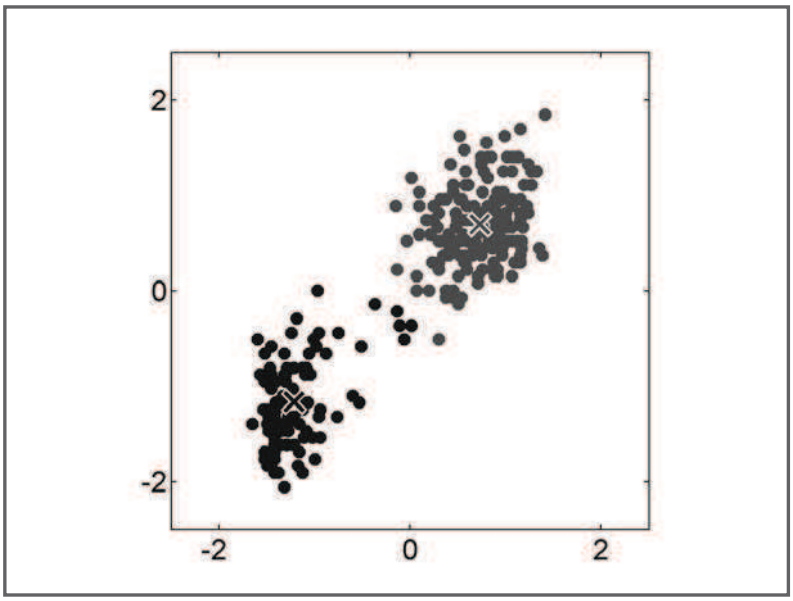
Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



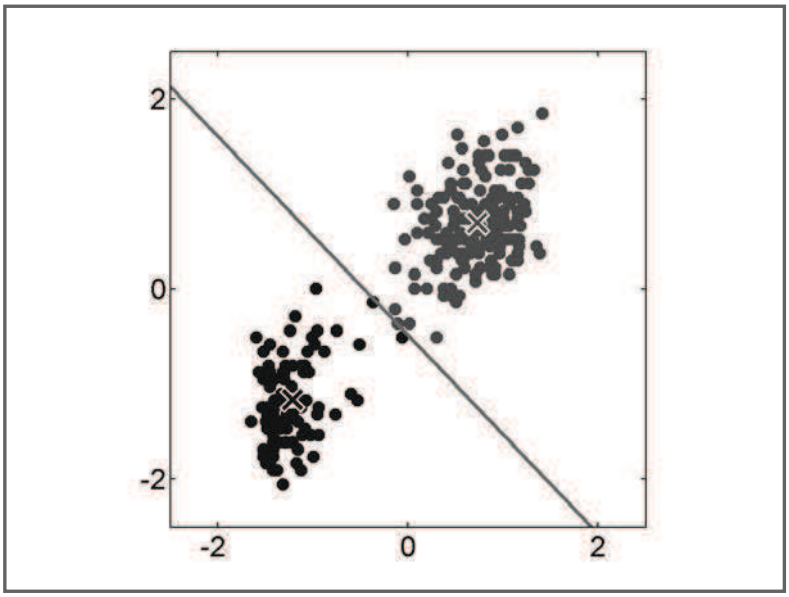
Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



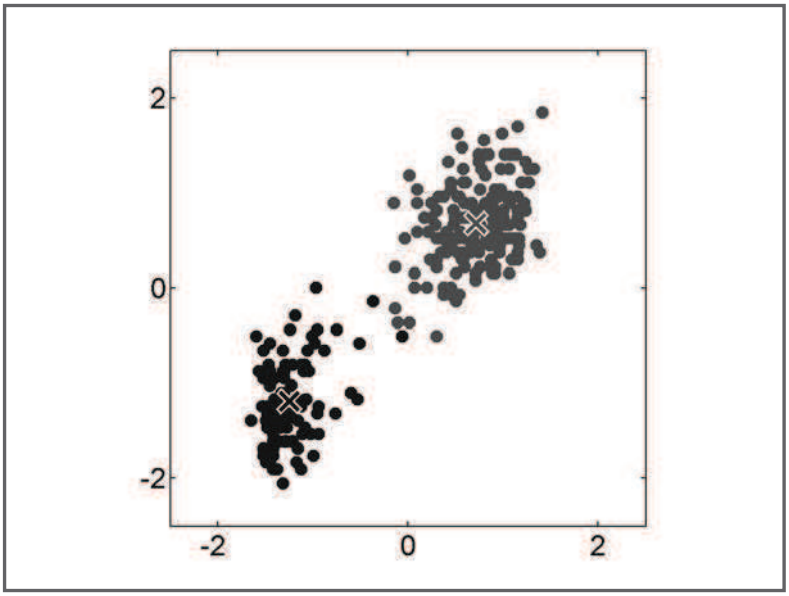
Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



Apprentissage non supervisé

Clustering (regroupement) - Méthode des k-moyennes



Apprentissage non supervisé
Clustering (regroupement) - Fuzzy c-means

Permet de partitionner les données en ensembles non-disjoints (recouvrements) à l'aide d'une mesure de similarité des données

A partir de l'ensemble des données

$$\mathbf{x}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,n}]^T \in \Re^n, \quad i = 1, \ldots, N$$

Trouver la matrice de partitionnement flou:

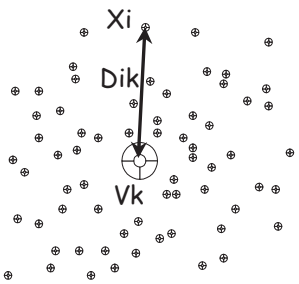
$$\mathbf{M} = [\mu_{ik}] = \begin{bmatrix} \mu_{11} & \cdots & \mu_{1C} \\ \vdots & \ddots & \vdots \\ \mu_{N1} & \cdots & \mu_{NC} \end{bmatrix}, \quad \mu_{ik} \in [0,1]$$

et les centres des clusters:

$$\mathbf{V} = \{\mathbf{v}_1, \ldots \mathbf{v}_C\}, \quad \mathbf{v}_k \in \Re^n$$

Apprentissage non supervisé
Clustering (regroupement) - Fuzzy c-means

Fonction objectif à minimiser



$$J = \sum_{i=1}^{N} \sum_{k=1}^{C} \mu_{ik}^m \left\| X_i - V_k \right\|_2^2$$

Sous les contraintes

$$0 \le \mu_{ik} \le 1, \quad i = 1, \ldots, N, k = 1, \ldots, C$$ Degré d'appartenance

$$\sum_{k=1}^{C} \mu_{ik}, \quad i = 1, \ldots, N$$ total membership

$$0 < \sum_{i=1}^{N} \mu_{ik} < N, \quad k = 1, \ldots, C$$ pas de cluster vide

$m \in (1, \infty)$ fuzziness parameter

Apprentissage non supervisé
Clustering (regroupement) - Fuzzy c-means

Répéter

On initialise **V** ou **U**

1. Calculer le centre des clusters

$$\mathbf{v}_k = \frac{\sum_{i=1}^{N} \mu_{ik}^m \mathbf{x}_i}{\sum_{i=1}^{N} \mu_{ik}^m}$$

2. Calculer les distances

$$d_{ik}^2 = (\mathbf{x}_i - \mathbf{v}_k)^T (\mathbf{x}_i - \mathbf{v}_k)$$

3. Mettre à jour la matrice de partitionnement

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{C} (d_{ik}^2 / d_{ij}^2)^{1/(m-1)}}$$

Jusqu'à $\|\Delta \mathbf{U}\| < \varepsilon$ D'autres tests d'arrêt sont possibles

Apprentissage non supervisé
Clustering (regroupement) - Algorithme de Gustafson-Kessel

- Utilise une mesure de distance adaptative

$$d^2(\mathbf{x}_i - \mathbf{v}_k) = (\mathbf{x}_i - \mathbf{v}_k)^T \mathbf{A}_k (\mathbf{x}_i - \mathbf{v}_k)$$
$$\mathbf{A}_k = |\mathbf{F}_k|^{1/n} \mathbf{F}_k^{-1}$$

- Fuzzy covariance matrix

$$\mathbf{F}_k = \frac{\sum_{i=1}^{N} (\mu_{ik})^m (\mathbf{x}_i - \mathbf{v}_k)(\mathbf{x}_i - \mathbf{v}_k)^T}{\sum_{i=1}^{N} (\mu_{ik})^m}$$

Apprentissage non supervisé

Clustering (regroupement) - Algorithme de Gustafson-Kessel

Répéter

1. Calculer le centre des clusters

$$\mathbf{v}_k = \frac{\sum_{i=1}^{N} \mu_{ik}^m \mathbf{x}_i}{\sum_{i=1}^{N} \mu_{ik}^m}$$

2. Calculer les distances

$$d_{ik}^2 = \left|\mathbf{F}_k\right|^{1/n} (\mathbf{x}_i - \mathbf{v}_k)^T \mathbf{F}_k^{-1} (\mathbf{x}_i - \mathbf{v}_k)$$

$$\mathbf{F}_k = \frac{\sum_{i=1}^{N} (\mu_{ik})^m (\mathbf{x}_i - \mathbf{v}_k)(\mathbf{x}_i - \mathbf{v}_k)^T}{\sum_{i=1}^{N} (\mu_{ik})^m}$$

3. Mettre à jour la matrice de partitionnement

$$\mu_{ik} = \frac{1}{\sum_{j=1}^{C} (d_{ik}^2 / d_{ij}^2)^{1/(m-1)}}$$

Jusqu'à $\|\Delta \mathbf{U}\| < \varepsilon$

Réseaux de neurones

Réseaux de neurones

- Repères historiques
- Inspiration biologique
- Le neurone formel
- Caractéristiques générales des réseaux de neurones
- Les architectures

Apprentissage des réseaux

- Principe de l'apprentissage
- Réseaux monocouche
- Réseaux multicouche

Réseaux de neurones

Le point de départ : la cybernétique (1942, congrès sur l'organisation du cerveau, NY)

Norbert Wiener 1948 : La cybernétique est la science qui étudie les processus de contrôle chez l'animal et dans la machine.

Problématique : construire des machines « intelligentes », i.e. donner aux machines des performances de plus en plus proches de celles de l'homme.

↓

Intelligence artificielle (IA)

↙ ↘

Approche connexionniste (réseaux de neurones)
modélisation du système cérébral

Approche symbolique (systèmes experts)
modélisation des processus cognitifs

Réseaux de neurones

Repères historiques

En 1943 : Mc Culloch et Pitts → Neurone formel.

En 1949 : Donald Hebb → Règle de Hebb (explication de l 'apprentissage)

En 1957 : Franck Rosenblatt → Perceptron (règle du perceptron).

En 1959 : Bernard Widrow → Adaline (LMS).

En 1969 : Marvin Minsky et Seymour Papert → critique du perceptron.

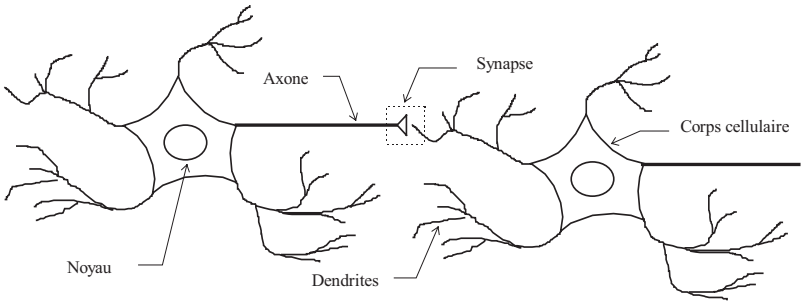
En 1970 : Teuvo Kohonen → Apprentissage compétitif.

En 1982 : John Hopfield → Modèle de Hopfield (réseau entièrement connecté).

En 1985 : David Rumelhart et Yann Le Cun → Algorithme de rétro-propagation.

Le neurone formel

Inspiration biologique



Le corps cellulaire : il effectue les transformations biologiques nécessaires à la survie du neurone

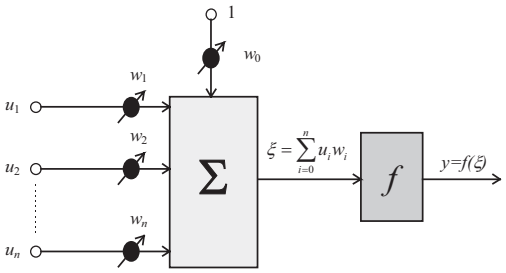
Les dendrites : elles forment une arborescence autour du corps cellulaire et jouent le rôle de récepteur de signaux

L'axone : il transmet le signal émis par le neurone

Les Synapses : ils réalisent l'interconnexion des neurones

Le neurone formel

Un neurone formel est une fonction non linéaire de la somme pondérée de ses entrées:



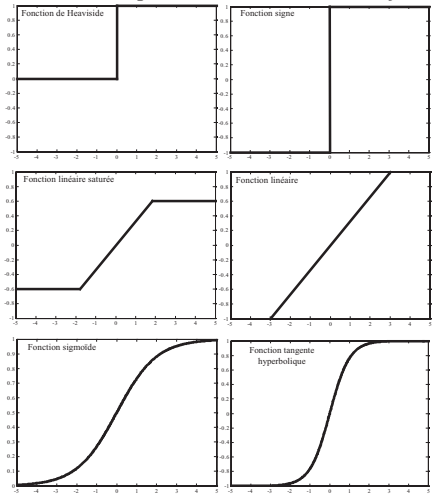
$$\xi = \sum_{i=0}^{n} u_i w_i = u^T w = w^T u, \quad y = f(\xi)$$

$$u^T = \begin{bmatrix} 1 & u_1 & u_2 & \cdots & u_n \end{bmatrix}$$

$$w^T = \begin{bmatrix} w_0 & w_1 & w_2 & \cdots & w_n \end{bmatrix}$$

Les coefficients de pondération w_i, ou poids, sont les paramètres ajustables du neurone

Principales fonctions d'activation *f*



L'interconnexion de plusieurs neurones formels réalise un réseau de neurones

Caractéristiques générales des réseaux de neurones

Un réseau de neurones réalise une fonction non linéaire de ses entrées par composition des fonctions réalisées par chacun de ses neurones.

Les réseaux de neurones sont des approximateurs universels.

Toute fonction continue bornée peut être approchée avec une précision arbitraire, dans un domaine fini de l'espace de ses variables, par un réseau de neurones comportant une couche de neurones cachés en nombre fini, possédant tous la même fonction d'activation, et un neurone de sortie linéaire

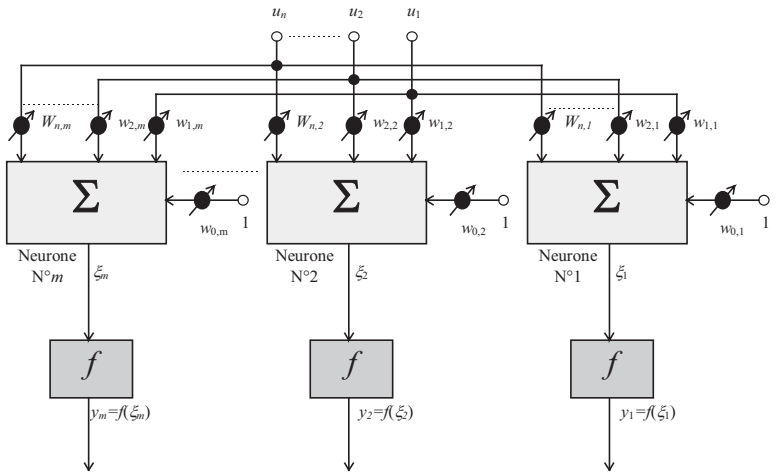
Deux types d'architecture

- **Les réseaux non bouclés**
  - Approximation de fonctions non linéaires
  - Classification automatique d'objets
  - Modélisation de processus statiques non linéaires
- **Les réseaux bouclés**
  - Identification des systèmes dynamiques non linéaires
  - Commande non-linéaire
  - Mémoires associatives dynamiques (bassin d'attraction)

Les réseaux de neurones non bouclés ou statiques

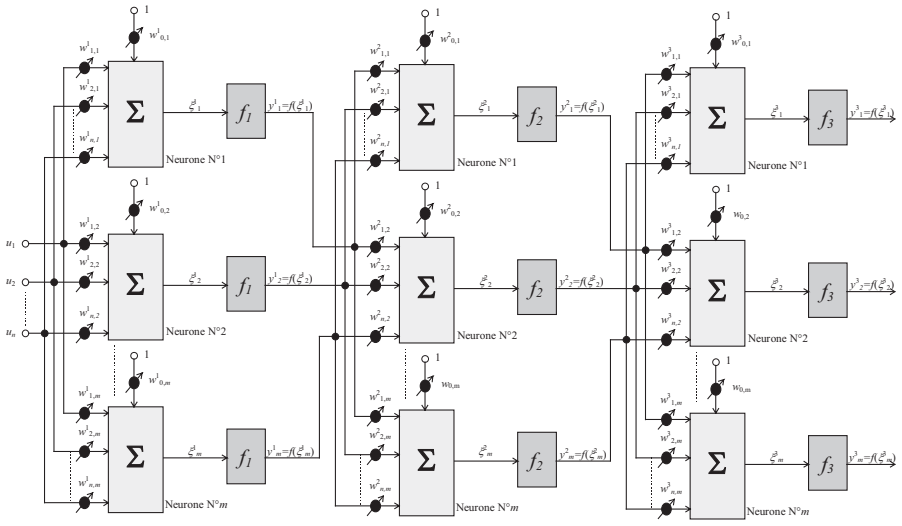
Un réseau de neurones non bouclé réalise une fonction statique non linéaire de ses entrées, par composition des fonctions réalisées par chacun de ses neurones

Réseau non bouclé monocouche (perceptron monocouche) : $y(k) = h(u(k), w)$



Les réseaux de neurones non bouclés ou statiques

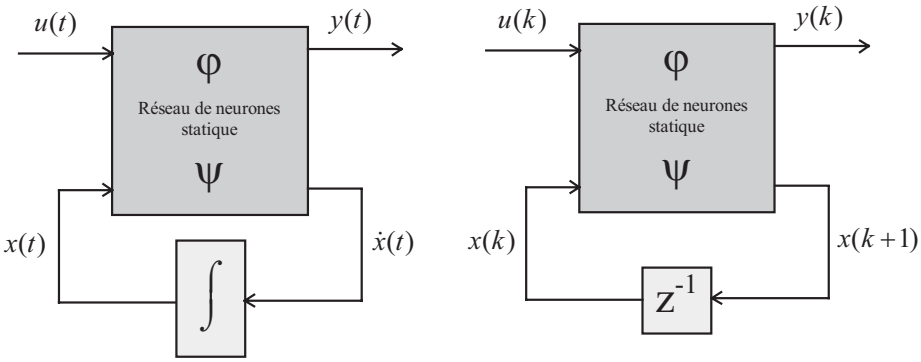
Réseau non bouclé multicouche (perceptron multicouche) : $y(k) = h(u(k), w)$



Les réseaux de neurones bouclés ou dynamiques

Un réseau de neurones bouclé est un système dynamique non linéaire, régi par des équations différentielles ou aux différences.

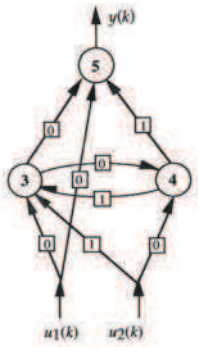
$$\dot{x}(t) = \varphi\big(x(t), u(t)\big) \qquad x(k+1) = \varphi\big(x(k), u(k)\big)$$
$$y(t) = \psi\big(x(t), u(t)\big) \qquad y(k) = \psi\big(x(k), u(k)\big)$$



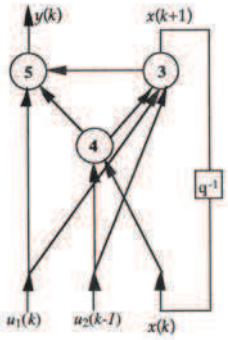
Les réseaux de neurones bouclés ou dynamiques

Exemple de mise sous forme d'état d'un réseau bouclé

Forme canonique d'un réseau de neurones bouclé



Un réseau de neurones bouclé. Les nombres dans les carrés sont les retards (exprimés en nombre de périodes d'échantillonnage) associés à chaque connexion.

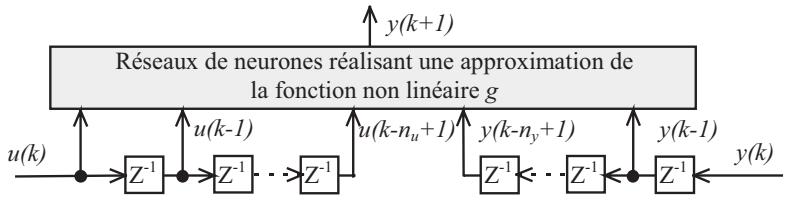


La forme canonique du réseau de la Figure 4.

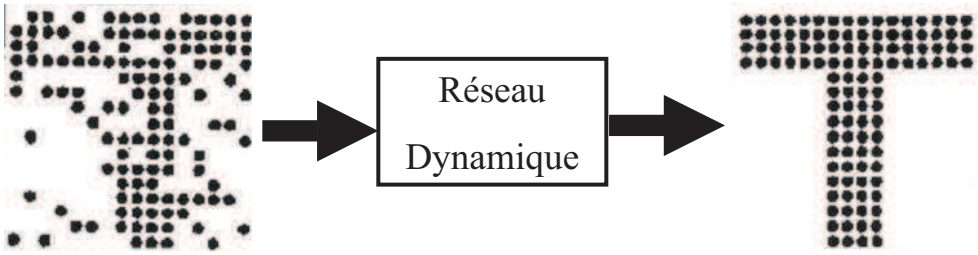
Les réseaux de neurones bouclés ou dynamiques

Modélisation de systèmes dynamiques non linéaires

$$y(k+1) = g(y(k),\ldots,y(k-n_y+1),u(k),\ldots,u(k-n_u+1))$$



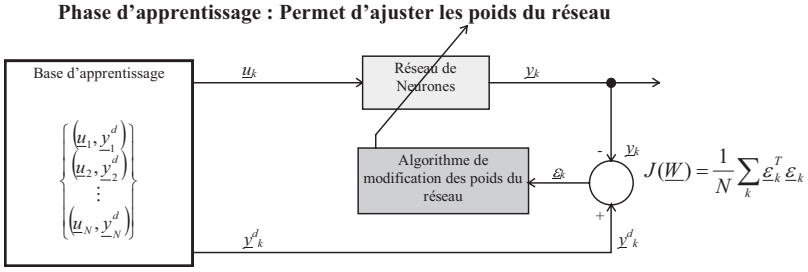
Mémoires associatives



Apprentissage des réseaux de neurones

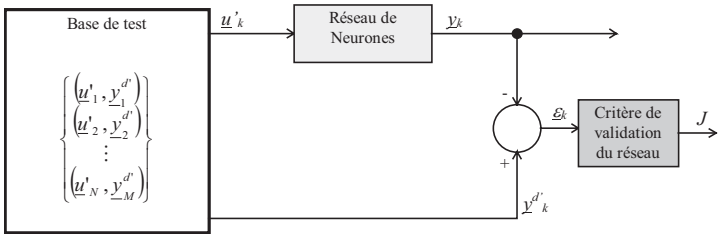
L'apprentissage d'un réseau de neurones consiste à modifier son comportement jusqu'à l'obtention d'un comportement désiré. L'apprentissage fait appel à des exemples de comportement

Phase d'apprentissage : Permet d'ajuster les poids du réseau



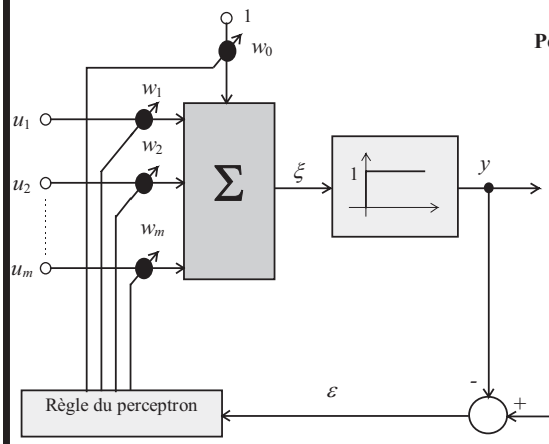
$$J(\underline{W}) = \frac{1}{N}\sum_k \underline{\varepsilon}_k^T \underline{\varepsilon}_k$$

Phase de test : Permet d'évaluer les capacités de généralisation du réseau



Algorithmes d'apprentissage, cas des réseaux monocouche

Règle du Perceptron



Pour une entrée donnée, on peut distinguer 4 cas :

1er cas :	**d=1**	**y=1**	**On ne modifie pas les poids**
2ème cas :	**d=0,**	**y=0**	
3ème cas :	**d=0,**	**y=1**	**On modifie les poids**
4ème cas :	**d=1,**	**y=0**	

Dans le 3ème cas if faut diminuer les poids

Dans le 4ème ces il faut augmenter les poids

$$w_{k+1} = w_k + \alpha\left(d_k - y_k\right)u_k$$

$$w_{k+1} = w_k + \alpha\varepsilon_k u_k$$

Algorithmes d'apprentissage, cas des réseaux monocouche

Algorithme d'apprentissage Perceptron

1. Initialisation des poids (par exemple à 0)

2. Présentation d'un exemple u_k de la base d'apprentissage

3. Calcul de la sortie y_k obtenue pour l 'entrée u_k

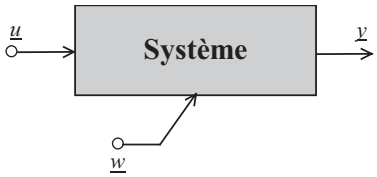
4. Si la sortie du Perceptron est différente de la sortie désirée d_k pour l'exemple u_k alors modification des poids :

$$w_{k+1} = w_k + \alpha\varepsilon_k u_k$$

5. Tant que tous les exemples de la base d'apprentissage ne sont pas traités correctement, retour à l'étape 2.

Algorithmes d'apprentissage, cas des réseaux monocouche

L'apprentissage, un problème d'optimisation



Soit un système d'entrée $\underline{u}$, de sortie $\underline{y}$ et possédant des éléments de réglage $\underline{w}$

L'optimalité d'un système n'a de sens que relativement à un certain critère de performance

Un système est dit optimal relativement au critère $J(\underline{w})$, si lors de son fonctionnement la quantité J est extrémale (maximale ou minimale)

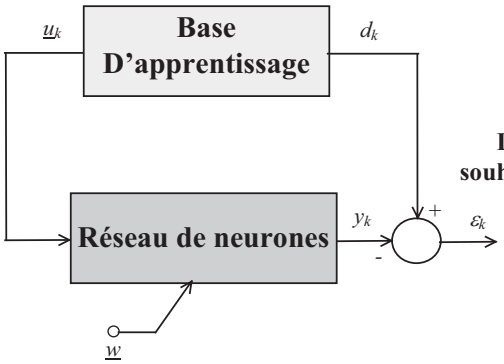
Optimiser un système consiste à déterminer le réglage $\underline{w}$ conduisant à un critère de performance $J(\underline{w})$ extremum :

$$\underline{w}_{opt} = \arg\min_{\underline{w}\in W} J(\underline{w}) \qquad \underline{w}_{opt} = \arg\max_{\underline{w}\in W} J(\underline{w})$$

Fonctionnement avec une consommation de carburant minimale
Fonctionnement avec une vitesse de déplacement maximale

Algorithmes d'apprentissage, cas des réseaux monocouche

L'apprentissage, un problème d'optimisation



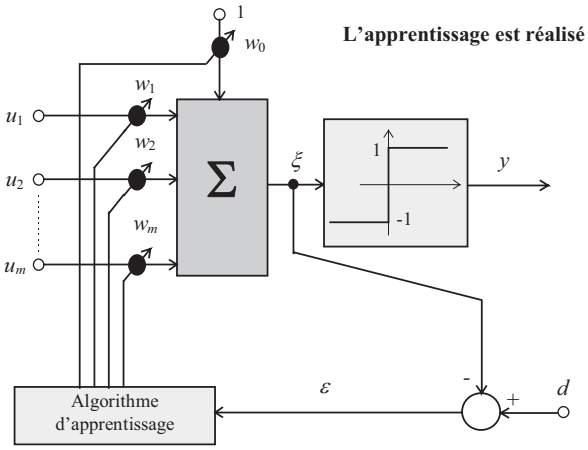
Dans le cas d'un réseau de neurones, on souhaite un fonctionnement à erreur minimale

L'apprentissage du réseau consiste alors à déterminer le vecteur poids $\underline{w}$ permettant de minimiser un critère sur l'erreur d'apprentissage

$$\cancel{J(\underline{w}) = \frac{1}{N}\sum_{k=1}^{N}\varepsilon_k} \qquad J(\underline{w}) = \frac{1}{N}\sum_{k=1}^{N}\left|\varepsilon_k\right| \qquad J(\underline{w}) = \frac{1}{N}\sum_{k=1}^{N}\varepsilon_k^2$$

Algorithmes d'apprentissage, cas des réseaux monocouche

L'ADALINE (adaptative linear element) de Widrow



L'apprentissage est réalisé sur la sortie linéaire du neurone

Apprentissage

- **MCS**
- **MCR**
- **Gradient → LMS**

$$\underline{u}_k = \begin{bmatrix} 1 \\ u_1^k \\ \vdots \\ u_m^k \end{bmatrix} \qquad \underline{w}_k = \begin{bmatrix} w_0^k \\ w_1^k \\ \vdots \\ w_m^k \end{bmatrix}$$

Algorithmes d'apprentissage, cas des réseaux monocouche

Méthode des moindres carrés simples (MCS)

C'est une méthode non itérative. Il s 'agit de déterminer le vecteur poid $\underline{w}$ permettant la minimisation du critère quadratique suivant:

$$J(\underline{w}) = \frac{1}{N}\sum_{k=1}^{N}\left(d_k - \xi_k\right)^2 = \frac{1}{N}\sum_{k=1}^{N}\varepsilon_k^2$$

$J(\underline{w})$ est minimum pour $\underline{w}=\underline{w}_{opt}$ si : $\left(\dfrac{\partial J}{\partial \underline{w}}\right)_{\underline{w}=\underline{w}_{opt}} = 0$ et $\left(\dfrac{\partial^2 J}{\partial \underline{w}^2}\right)_{\underline{w}=\underline{w}_{opt}} > 0$

Ce qui conduit à : $\underline{w}_{opt} = \left(U^T U\right)^{-1} U^T \underline{d}$ *avec* : $\underline{d} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix}, \quad U = \begin{bmatrix} \underline{u}_1^T \\ \underline{u}_2^T \\ \vdots \\ \underline{u}_N^T \end{bmatrix}$

Algorithmes d'apprentissage, cas des réseaux monocouche

Méthode des moindres carrés récursifs (MCR)

C'est une méthode itérative (implémentation temps réel possible), qui permet la mise à jour du vecteur poids à partir de la donné d 'un nouvel exemple.

Pour *N* exemples la méthode des MCS permet le calcul direct du vecteur poids: $\underline{w}_N = \left(U^T U\right)^{-1} U^T \underline{d}$

Problème : comment modifier le vecteur poids wn lorsque l'on connaît un nouvel exemple $\underline{u}_{N+1}$, d_{N+1} ?

On pose $F_N=(U^T U)^{-1}$, et l'on utilise les récurrences suivantes :

$$\begin{cases} F_{N+1} = F_N - \dfrac{F_N \underline{u}_{N+1} \underline{u}_{N+1}^T F_N}{1 + \underline{u}_{N+1} F_N \underline{u}_{N+1}^T} \\ \underline{w}_{n+1} = \underline{w}_N + F_{N+1} \underline{u}_{N+1} \left(d_{N+1} - \underline{u}_{N+1}^T \underline{w}_N\right) \end{cases}$$

Algorithmes d'apprentissage, cas des réseaux monocouche

Méthode du gradient, algorithme LMS

C'est une méthode itérative. Il s 'agit de déterminer le vecteur poids wk permettant de diminuer le critère $J(\underline{w}_k)$ à chaque présentation de la base d 'exemples.

$$J(\underline{w}_k) = \frac{1}{N} \sum_{n=1}^{N} \left(d_n - \xi_n\right)^2 = \frac{1}{N} \sum_{n=1}^{N} \varepsilon_n^2$$

Le vecteur poids est calculé à l 'aide de la formule du gradient : $\underline{w}_{k+1} = \underline{w}_k - \mu \left(\dfrac{\partial J}{\partial \underline{w}} \right)_{\underline{w}=\underline{w}_k}$

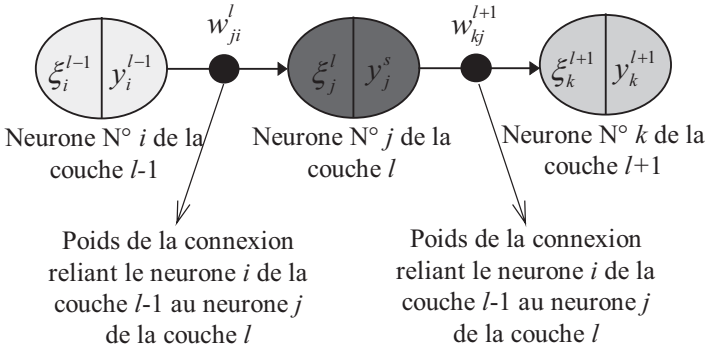
Ce qui conduit à : $\underline{w}_{k+1} = \underline{w}_k + \dfrac{2\mu}{N} \sum_{n=1}^{N} \varepsilon_n \underline{u}_n$

Dans le cas de l 'algorithme LMS, la méthode du gradient est appliquée sur $(\varepsilon_k)^2$ au lieu de $J(\underline{w}_k)$. Le vecteur poids est donc modifié à chaque présentation d'un exemple. La règle de modification des poids s 'écrit :

$$\underline{w}_{k+1} = \underline{w}_k + 2\mu \varepsilon_k \underline{u}_k$$

Algorithmes d'apprentissage, cas des réseaux multicouche

Algorithme de rétropropagation



$$\xi_j^l = \sum_i w_{ji}^l y_i^{l-1}, \quad y_j^l = f\left(\xi_j^l\right), \quad \xi_k^{l+1} = \sum_j w_{kj}^{l+1} y_j^l, \quad y_k^{l+1} = f\left(\xi_k^{l+1}\right)$$

Objectif : minimiser l'erreur quadratique commise un exemple donné :

$J = \frac{1}{2}(\underline{d} - \underline{y}^o)^2$ **Où d est la sortie désirée et y_o la sortie effective du réseau**

Algorithmes d'apprentissage, cas des réseaux multicouche

Algorithme de rétropropagation

Pour chaque exemple présenté, on réalise les calculs suivants :

Calcul des ξ_j et y_j pour l 'ensemble des neurones

$$\xi_j^l = \sum_i w_{ji}^l y_i^{l-1}, \quad y_j^l = f\left(\xi_j^l\right)$$

Calcul des erreurs locales pour la couche de sortie

$$e_k^o = (y_k^o - d_k) f'\left(\xi_k^o\right)$$

Calcul des erreurs locales pour l 'ensemble des neurones des couches cachées

$$e_j^l = f'\left(\xi_k^o\right) \sum_k e_k^{l+1} w_{kj}^{l+1}$$

Modification des poids

Couche de sortie : $\Delta w_{kj}^o = \mu e_k^o y_j^{o-1}$

Couches cachées : $\Delta w_{ji}^l = -\mu e_j^l y_i^{l-1}$