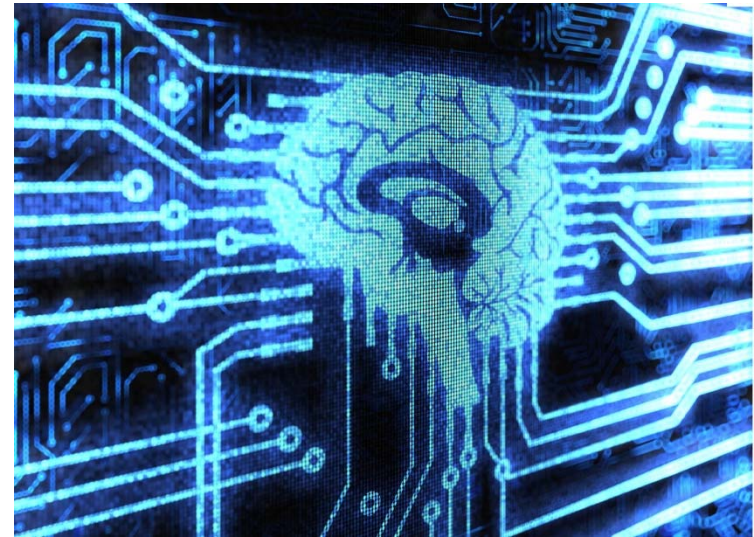


Généralités



Concepts fondamentaux



Méthodes de la RdF :

- **Présentation** : Discipline qui cherche à reproduire sur ordinateur différentes activités humaines (essentiellement la perception).
- **Comment ça marche ?**
 - ✓ On donne à l'algorithme des données d'entraînement.
 - ✓ l'algorithme d'apprentissage machine apprend un modèle capable de **généraliser** à de nouvelles données.

Notion d'apprentissage :

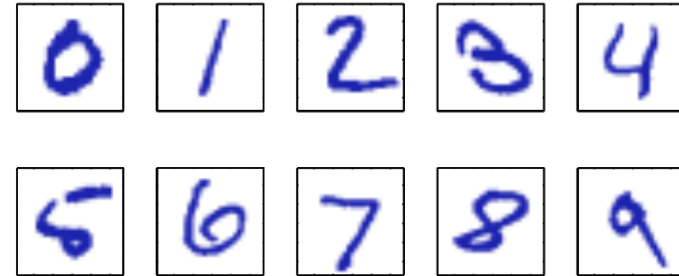
Méthodes d'apprentissage :

Imitation (enfant), association (stimuler), essais/erreurs, explication, répétitions, immersion (langue), combiné...

Consiste à améliorer, si possible de façon automatique, les capacités de discrimination entre les différentes classes proposées. Il repose sur l'exploitation judicieuse de formes de référence convenablement choisies.

Exemple :

Reconnaissance de chiffre.



- On appelle ensemble d'entraînement :

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

- x_n une **observation** : entrée du système.
- t_n la **cible(étiquette, label)** correspondante : sortie du système.

| | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|
| x_n | 9 | 6 | 6 | 5 | 4 | 0 |
| t_n | '9' | '6' | '6' | '5' | '4' | '0' |

Exemple : Reconnaissance de chiffre.

- L'apprentissage machine fournit un modèle $y(x)$
- L'objectif est de trouver un modèle tel que :

$$y(x_n) = \hat{t}_n \simeq t_n$$

- On mesure la qualité de l'apprentissage (la qualité du modèle) sur un ensemble de test :

$$\mathbb{D}_{test} = \{(x_{N+1}, t_{N+1}), \dots, (x_{N+M}, t_{N+M})\}$$

Types d'apprentissage :

Supervisé :

si les différentes classes sont connues « à priori », ou le Nombre de classes connu et on connaît la classe de chaque forme de l'ensemble d'échantillons.

$$\mathcal{D} = \{(\mathbf{x}_1, t_1), \dots, (\mathbf{x}_N, t_N)\}$$

Semi-supervisé :

on dispose à la fois d'un (petit) ensemble de données étiquetées et un (grand) ensemble de données non étiquetées. Entraîner sur les données avec label, et l'exploiter sur des données sans label.

Types d'apprentissage :

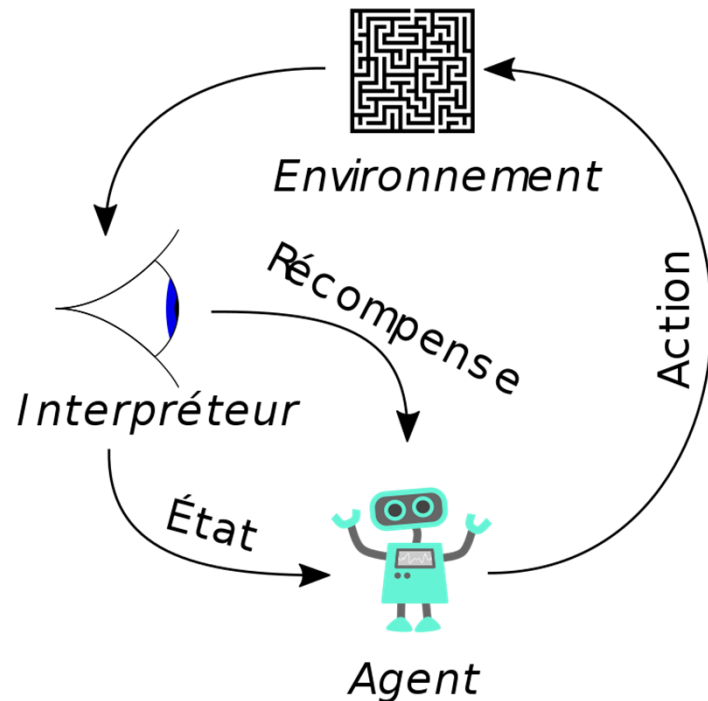
Non supervisé ou classification automatique :

Nombre de classes connu ou non, et on ne connaît pas la classe des échantillons. Il s'agit de diviser un groupe de données, en sous-groupes de manière à ce que les données considérées comme les plus similaires soient associées au sein d'un groupe homogène et qu'au contraire les données considérées comme différentes se retrouvent dans d'autres groupes distincts(inter et intra).

Poser des hypothèses :

- les points proches ont probablement le même label de la classe.
- deux points qui sont connectés par un chemin traversant des régions de fortes densités doivent avoir le même label

Types d'apprentissage :



Par renforcement :

Désigne une méthode adaptative permettant de résoudre un problème de décision, le terme adaptative signifie qu'on part d'une solution inefficace et qu'elle est améliorée progressivement en fonction de l'expérience des agents.

état courant + fct récompense \Rightarrow positif / négatif (on apprend un comportement étant donné une observation).

Apprentissage par Renforcement (A/R), les algorithmes de bandit pour le compromis exploration-exploitation, et la programmation dynamique avec approximation (PDA), dans le cadre des processus de décision markoviens (PDM). Monte-Carlo...

Apprentissage Supervisé :

Classification ou Régression

L'apprentissage supervisé est lorsqu'on a une cible à prédire :

Classification : la cible est un indice de classe $t \in \{1, \dots, K\}$

-**exemple** : reconnaissance de caractères

x : vecteur des intensités de tous les pixels de l'image.

t : *identité du caractère.*

Régression : la cible est un nombre réel $t \in \mathbb{R}$

-**exemple** : prédiction de la valeur d'une action à la bourse

x : vecteur contenant l'activité économique de la journée.

t : *valeur d'une action à la bourse le lendemain.*

Méthodes de la RdF :

- Soit F un ensemble de formes, et X des représentations : $X = \{x_1, x_2, \dots, x_n\}$.
- La classification est parfaitement réalisée si les classes X_i obtenues forment une partition de X (aucune classe n'est vide, les classes sont deux à deux disjointes et leur réunion est X).

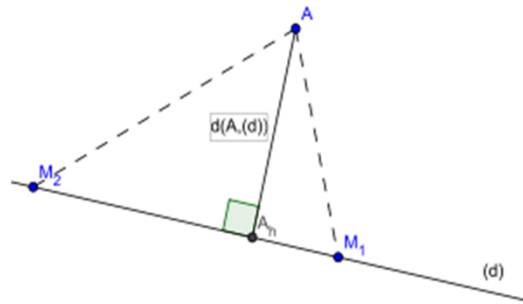
Notion de classification :

- Consiste à affecter une représentation à une classe déjà connue. Ces classes peuvent être connues «a priori » ou par apprentissage.
- Pour réaliser cette classification, on utilise souvent la notion de distance(moyen intuitif pour mesurer la proximité entre deux représentations de formes.)

Notion de Distance :

On appelle distance d'un point $p \in X$ à une classe à une classe X_0 :

$$d(p, X_0) = \inf \{d(p, m), m \in X_0\}$$



Distance : Métrique qui satisfait 3 axiomes :

- Réflexivité : $d(i, i) = 0$
- Symétrie : $d(i, j) = d(j, i)$
- Inégalité triangulaire : $d(i, j) \leq d(i, k) + d(k, j)$

Notion de Distance :

$V_1(x_1, x_2, \dots, x_n)^t$ et $V_2(y_1, y_2, \dots, y_n)^t$

Distance Euclidienne : $D_E = (V_1, V_2) = \sqrt{\sum_{i=1}^n (y_i - x_i)^2}$

Distance Hamming : $D_H = (V_1, V_2) = \sum_{i=1}^n |y_i - x_i|$

Autres types de distance : Itakura, Manhattan, mahalanobis ; Minkowski, Tchebychev etc...

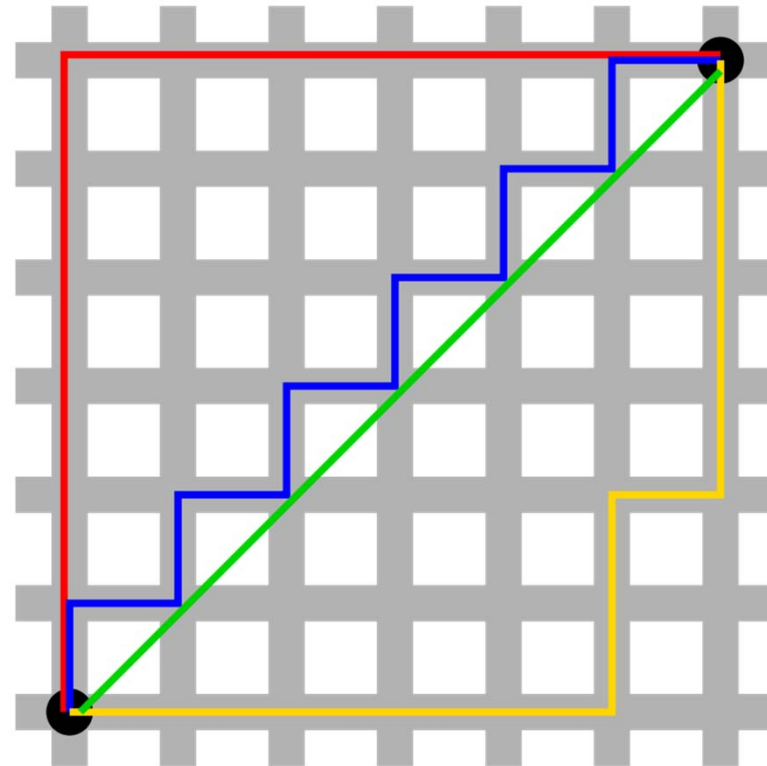
Notion de Distance :

Exemples :

Distance de Manhattan chemins (rouge, jaune et bleu).

Contre :

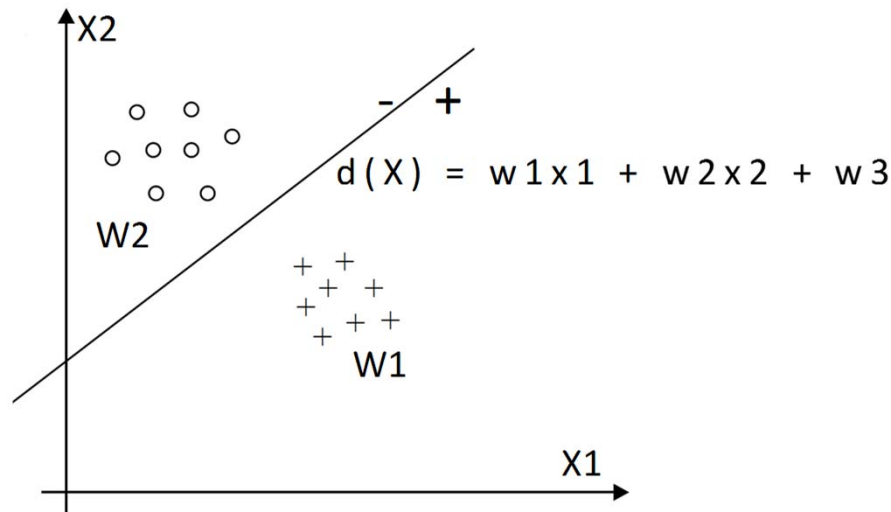
Distance euclidienne en vert.



Levenshtein par exemple la correction automatique pour des programmes de traitement de texte (le programme va rechercher dans un dictionnaire les mots présentant les distances les plus faibles avec le mot mal orthographié).

Classification binaire :

Le rôle principal d'un système de reconnaissance est de prendre une décision sur l'appartenance d'une forme aux classes, il est nécessaire de définir les règles sur lesquelles seront basées ces décisions. On considère la figure où deux classes de formes sont représentées, on remarque qu'il suffit d'une droite pour les séparer.



$$X \in W1 \quad \text{si} \quad d(X) > 0$$

$$X \in W2 \quad \text{si} \quad d(X) < 0$$

$$X \in \text{frontière} \quad \text{si} \quad d(X) = 0$$

$$d(X) = w_1x_1 + w_2x_2 + \dots + w_nx_n + w_{n+1}$$

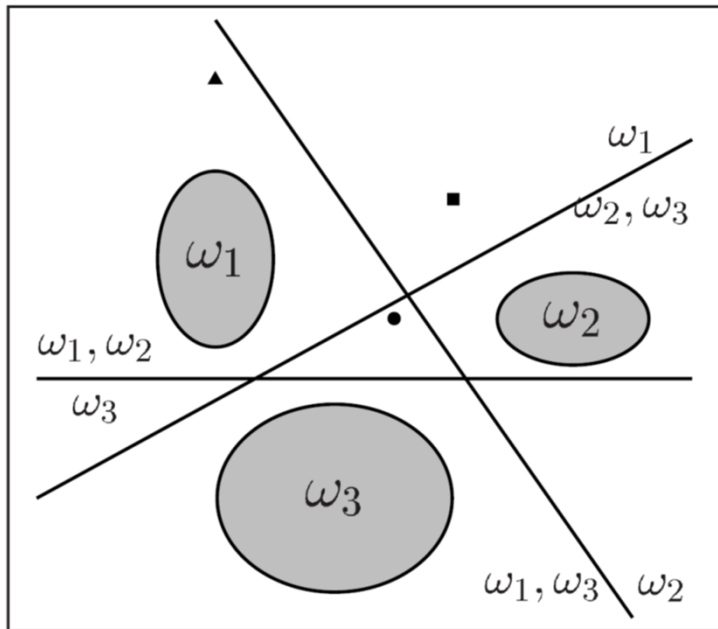
$$X = (x_1, x_2, \dots, x_n, 1)$$

Toute forme X appartenant à la classe W1 conduit à une valeur positive pour d(x) et inversement pour W2

Multi classe :

Premier cas :

chaque classe est séparée des autres par une simple surface de décision, il y a M fonctions de décision.

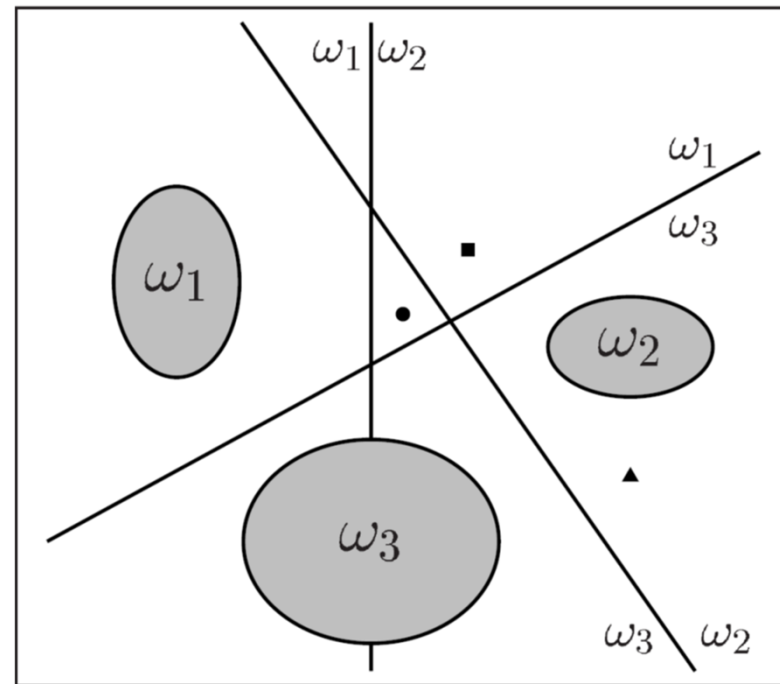


$$d_i(X) = W_i^t X = \begin{cases} > 0 & \text{si } X \in W1 \\ < 0 & \text{sinon} \end{cases} \quad i = 1, \dots, M$$

Multi classe :

Deuxième cas :

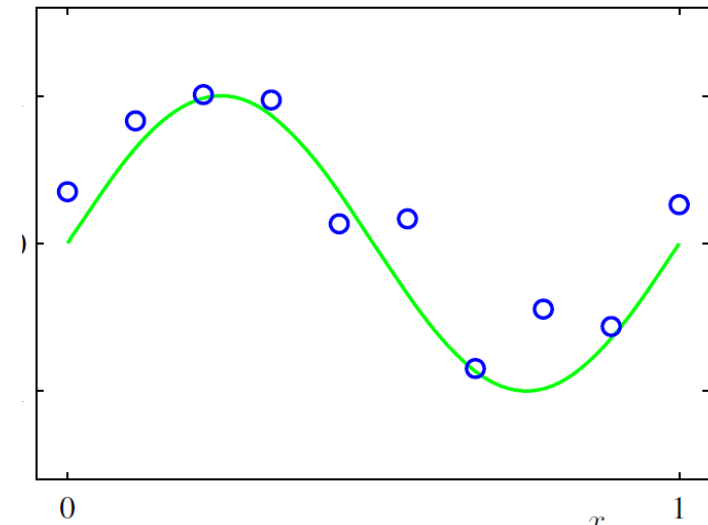
une autre idée qui consiste à séparer entre chaque couple de classes, elle présente l'avantage de dessiner moins de zones d'ambiguïté, mais elle a l'inconvénient de d'exiger $M(M-1)/2$ séparateurs ou fonctions de décision.



Sous apprentissage / Sur apprentissage

On va supposer qu'une bonne prédiction aurait une forme polynomiale :

- Données d'entraînement \mathcal{D} contiennent :
 - $\mathbf{x} \equiv (x_1, \dots, x_N)^T$
 - $\mathbf{t} \equiv (t_1, \dots, t_N)^T$
- Objectif :
 - faire une prédiction \hat{t} pour une nouvelle entrée \hat{x}

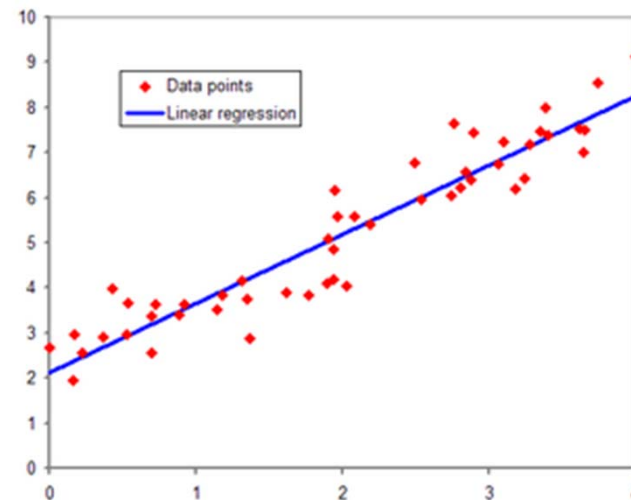
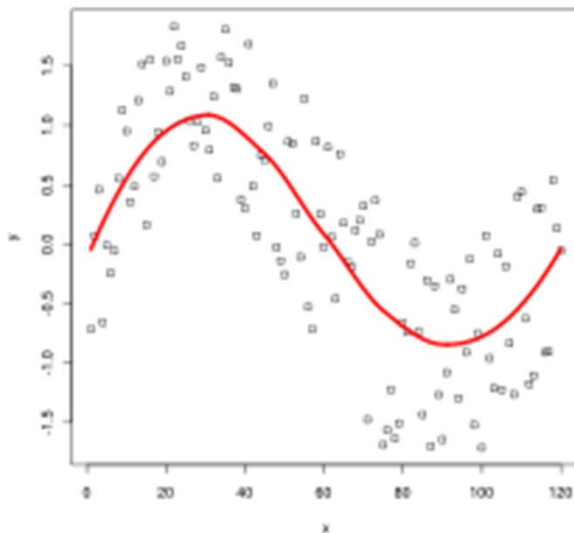


$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M$$

Sous apprentissage / Sur apprentissage

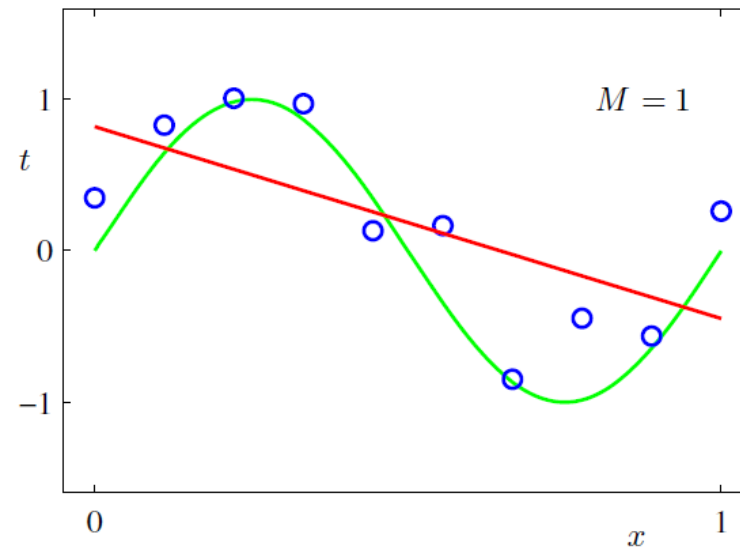
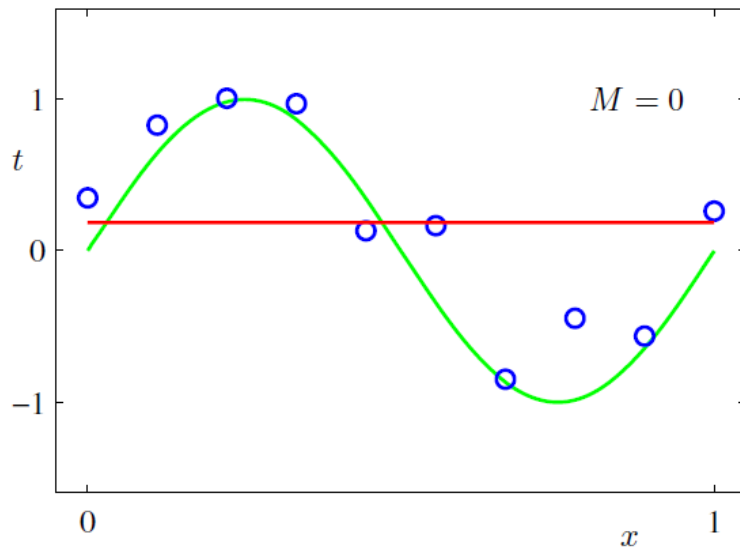
- Estimation du modèle = estimation des coefficients w_m
 - Pb : Comment trouver W^* qui minimise l'erreur ?

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2$$



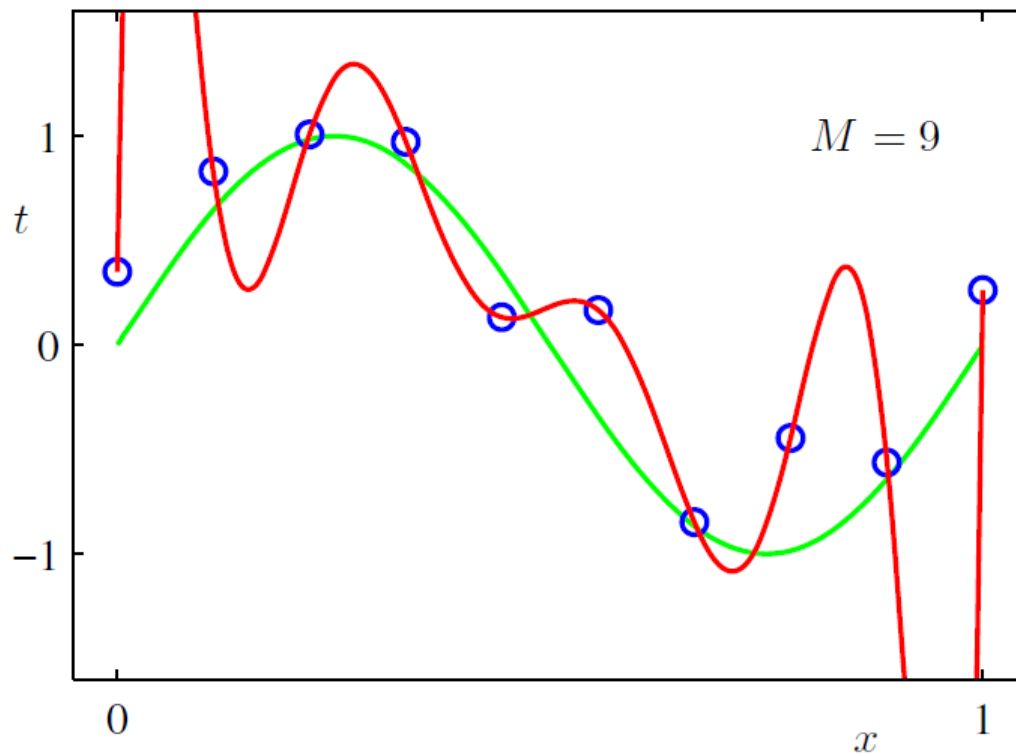
Sous apprentissage / Sur apprentissage

- Un algorithme d'apprentissage résoudrait ce problème.
 - ❖ à partir des données, il va retourner \mathbf{W}^*
- Comment choisir l'ordre du polynôme M ?
 - de trop petites valeurs auront une grande perte sur l'ensemble d'entraînement (on modélise mal les données) : situation de **sous-apprentissage**.



Sous apprentissage / Sur apprentissage

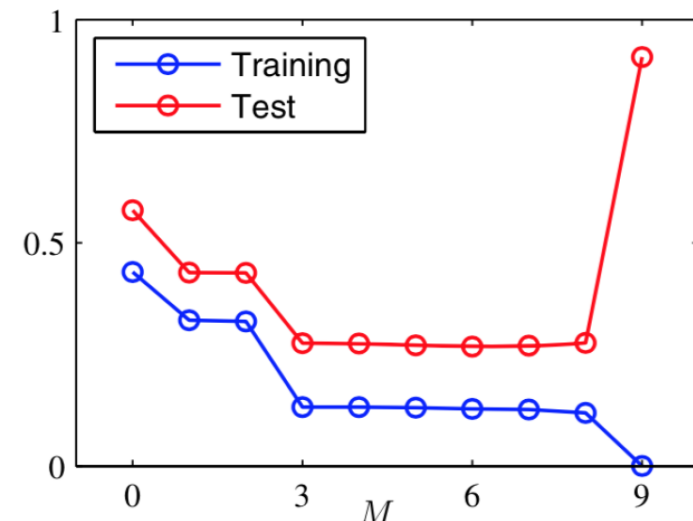
- si M est trop grand : on apprend "par cœur" de l'ensemble d'entraînement: **sur-apprentissage**.



Résultat : va apprendre à prédire ce qui n'est pas prévisible à partir de x seulement (p. ex. du bruit)

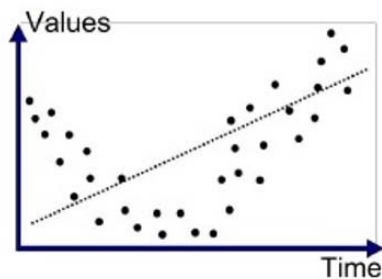
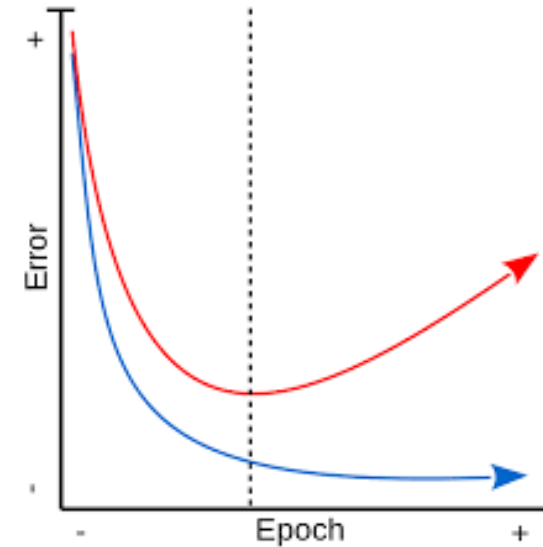
Sous apprentissage / Sur apprentissage

- **Généralisation** : On cherche une valeur de M qui permet de retrouver la tendance générale de la relation entre x et t :
 - sans apprendre le bruit
 - va permettre de généraliser à de nouvelles données
 - Trouver cette meilleure valeur de M s'appelle de la **sélection de modèle** (M et λ sont appelés des hyper-paramètres).
- **Capacité** : Aptitude d'un modèle à apprendre "par cœur"
 - plus M est grand, plus le modèle a de la capacité.
 - plus la capacité est grande, plus la différence entre l'erreur d'entraînement et l'erreur de test augmente.

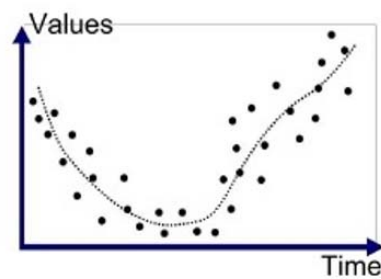


Sur-apprentissage (overfitting) :

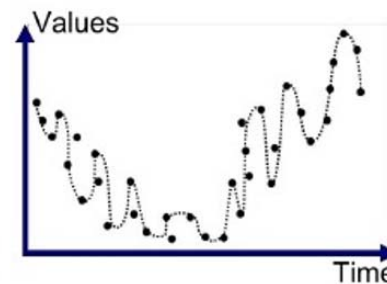
- est un apprentissage ayant dépassé le stade où aucun nouveau progrès n'est observé ; on dit que le système aura du mal à généraliser.



Underfitted



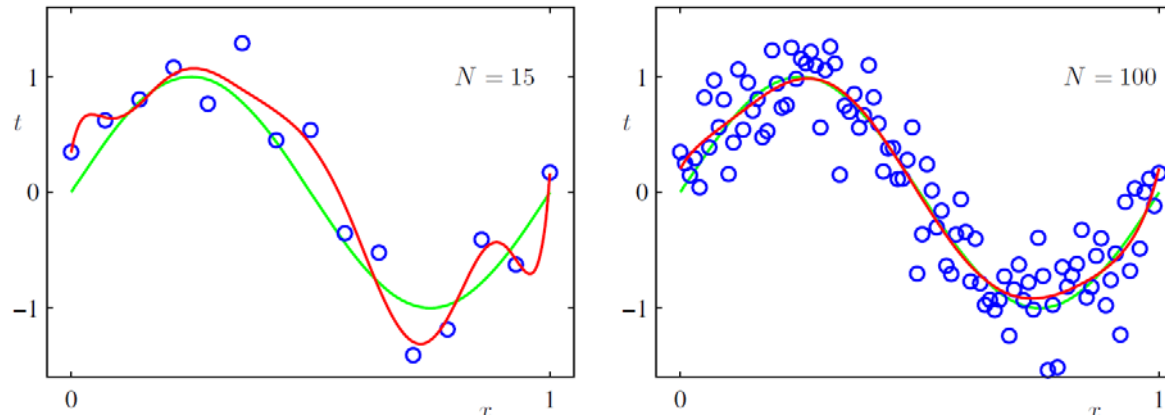
Good Fit/Robust



Overfitted

Sur-apprentissage (overfitting) :

- **généralisation vs. quantité de données** : Plus la quantité de données d'entraînement augmente, plus le modèle entraîné va bien généraliser.



- Ce phénomène de **Sur-apprentissage** peut être résolu par un ensemble de **validation** ; diviser la base d'apprentissage en deux parties, une pour l'apprentissage ($\sim 80\%$) et une deuxième pour la validation ($\sim 20\%$).
- Surveiller l'évolution de l'erreur dans les deux ensembles, dès que l'erreur de validation commence à croître alors que l'erreur continue à décroître dans la base d'apprentissage, on arrête le processus. On dit que le système est en train de perdre son pouvoir de généralisation.

Sélection de Modèle : (ex : régression)

Solution 1 :

On réserve un ensemble d'entraînement ***train***, le reste ***valid*** servira à comparer les hyper-paramètres.

Pour chaque valeur d'hyper-paramètres à comparer :

- obtenir un modèle entraîné à partir de ***train***.
- évaluer la performance du modèle sur ***valid***.

Retourner le choix d'hyper-paramètres ayant donné le modèle avec la meilleure performance sur ***valid***.

Sélection de Modèle : (ex : régression)

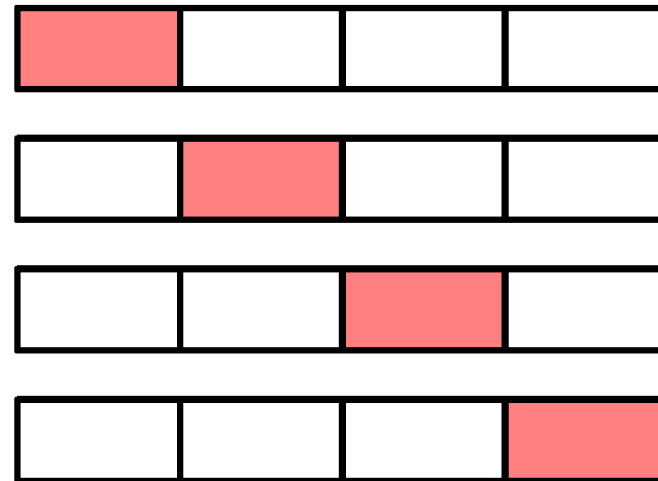
Solution 2 : *S-fold cross-validation*

Lorsqu'on a peu de données, 20% est trop peu pour estimer la performance de généralisation.

- Répéter la procédure de séparation *train/valid* plus d'une fois.

S-fold cross-validation : divise les données en S portions différentes; chaque portion est utilisée une fois en tant que ***valid***.

Exemple : $S = 4$



Sélection de Modèle : (ex : régression)

Solution 2 : *S-fold cross-validation*

Le but est d'utiliser tous nos exemples au moins une fois.

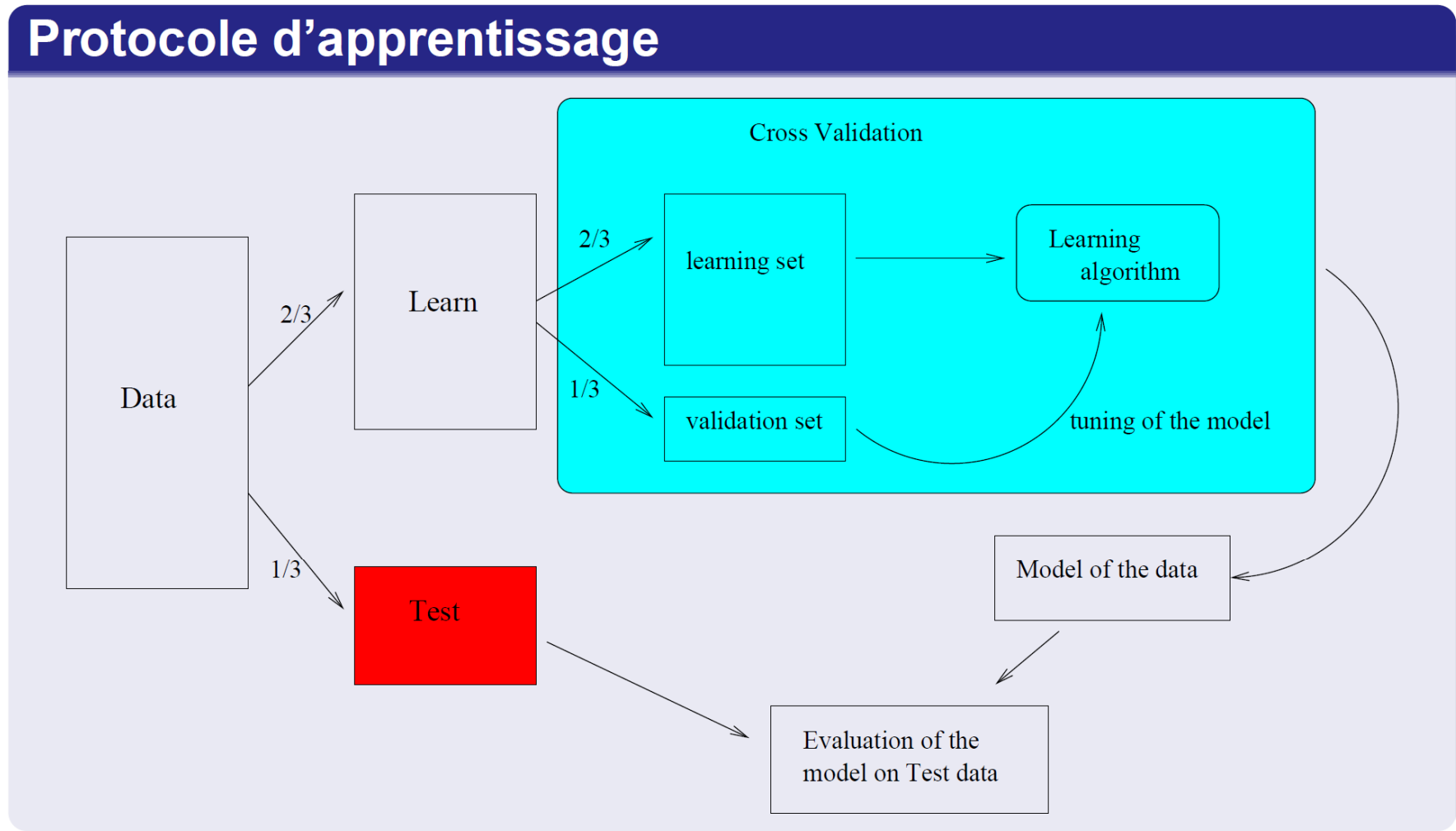
pour $s = 1 \dots S$

- pour chaque valeur d'hyper-paramètres à comparer :
 - obtenir un modèle entraîné à partir de train = $D - D_s$
 - évaluer la performance du modèle sur valid = D_s

Retourner la valeur des hyper-paramètres ayant la meilleure performance **moyenne** sur les ensembles valid.

Si $S = N$, on parle alors de méthode *leave-one-out*

Evaluation d'un système de classification



Principe: comparer les prédictions de classes aux « vérités terrains ».

Evaluation d'un système de classification

Indices de performance :

- Le taux de bonne classification (tbc_s) : l'indicateur le plus naturel et le plus évident permettant d'évaluer les performances d'un système de classification ; correspond au nombre d'éléments correctement identifiés.

$$tbc_s = \frac{\text{Nombre d'éléments correctement identifiés}}{\text{Nombre d'éléments total}}$$

- le taux d'erreur : $te_s = 1 - tbc_s$
- Lorsque le rejet d'une forme est possible : $te_s = 1 - tbc_s - tr$

Le problème avec le taux de bonne classification : mesure "faible", car elle ne tient pas compte de la distribution des classes et des coûts de classification.

Evaluation d'un système de classification

Exemple: le cas d'un diagnostic médical, une base constituée de 5 personnes malades et de 995 personnes saines. Si nous décidons que tout le monde est sain, nous obtenons alors un taux de bonne reconnaissance $tbc_s = 99,5\%$. Au vu de ce résultat, le classifieur est donc très performant !!!

Matrice de confusion ou tableau de contingence sert à évaluer la qualité d'une classification:

1. Accuracy, precision, rappel, f-mesure, sensibilité, spécificité, etc..
2. Courbe Précision–Rappel et la courbe ROC (Insensible à la distribution et Indépendant des couts d'erreur).

Evaluation d'un système de classification:

Évaluation du test : Matrice de confusion

| | | Réel | |
|--------|-----|------|-----|
| | | Pos | Neg |
| Prédit | Pos | TP | FP |
| | Neg | FN | TN |

- TP : True Positive
- FP : False Positive
- FN : False Negative
- TN : True Negative

Précision, Rappel, Accuracy

- $Precision = \frac{TP}{TP+FP}$
- $Rappel = \frac{TP}{TP+FN}$
- $Accuracy = \frac{TP+TN}{TP+FN+FP+TN}$

Sensibilité, Spécificité

- $sensibilite = \frac{TP}{TP+FN}$
- $specificite = \frac{TN}{FP+TN}$

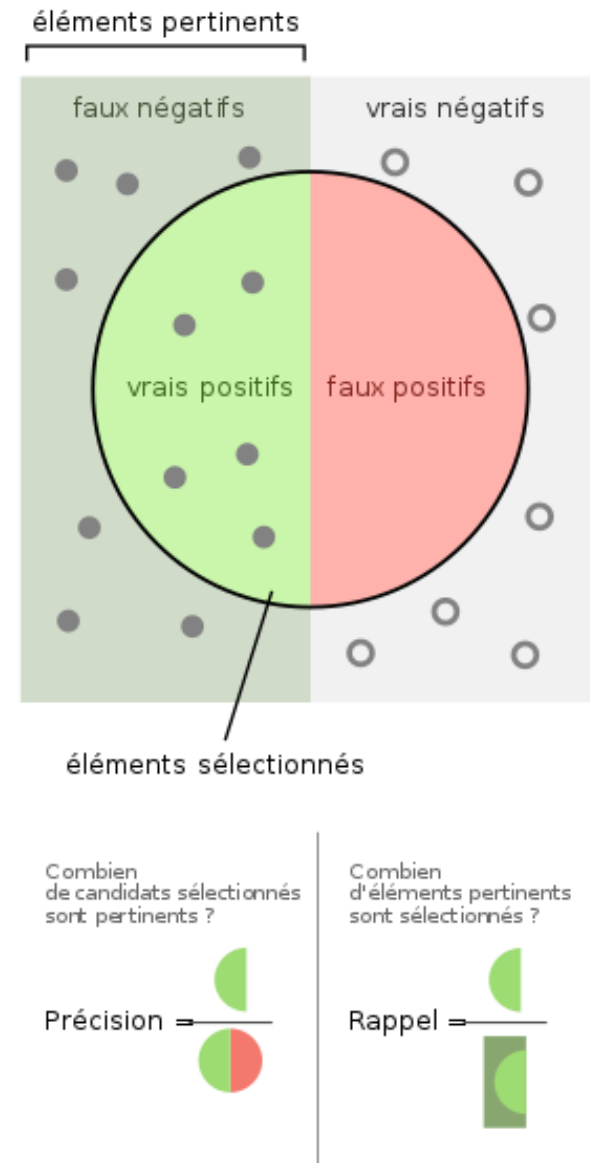
Evaluation d'un système de classification:

Accuracy (exactitude): le taux de bonne classification, mesure les performances globales indépendamment de la distribution.

Précision : données détectées comme c_i et étant réellement c_i / données détectées comme c_i (correctes ou fausses); mesure la capacité à retrouver uniquement des c_i

Rappel (recall) : données détectées comme c_i et étant réellement c_i / données étant réellement c_i ; mesure la capacité à retrouver tous les c_i

F-measure: prend en compte simultanément le Rappel et la Précision.



Evaluation d'un système de classification:

Matrice de confusion multi-classes

| | | Réal | | | | | |
|--------|-------|---------|---------|-----|---------|-----|---------|
| | | C_1 | C_2 | ... | C_i | ... | C_n |
| Prédit | C_1 | c_1^1 | c_1^2 | | c_1^i | | c_1^n |
| | C_2 | c_2^1 | | | | | |
| | ... | ... | | | | | |
| | C_i | c_i^1 | | | c_i^i | | |
| | ... | ... | | | | | |
| | C_n | c_n^1 | | | | | |

- Prédiction correcte : c_i^i
- Prédiction incorrecte : c_i^j avec $i \neq j$

Précision, Rappel, Accuracy

- $Precision(C_i) = \frac{c_i^i}{\sum_{j=1}^n c_i^j}$

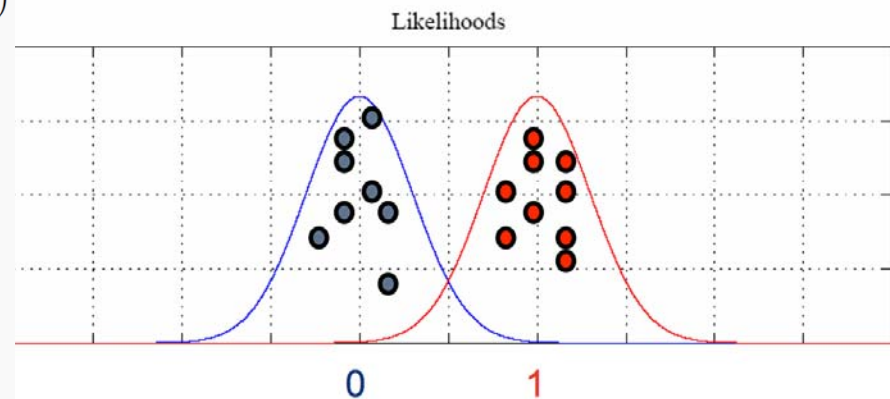
- $Rappel(C_i) = \frac{c_i^i}{\sum_{j=1}^n c_j^i}$

- $Accuracy = \frac{\sum_{i=1}^n c_i^i}{\sum_{i,j=1}^n c_i^j}$

Trois grandes méthodes pour l'apprentissage supervisé

Approche générative :

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



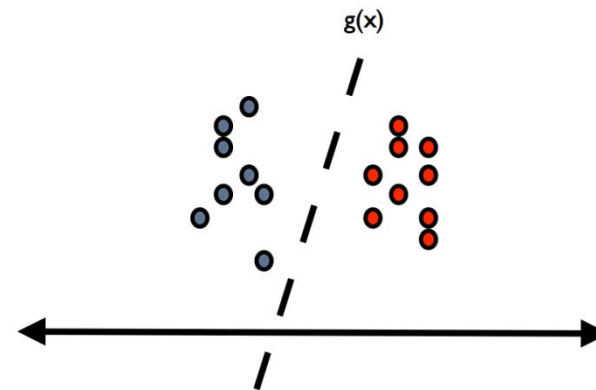
nous partons de l'hypothèse qu'il existe une famille de modèles paramétriques permettant de générer X connaissant Y .

- Exemple : apprentissage Bayésien, modèle de Markov caché, réseaux de neurones artificiels.

Trois grandes méthodes pour l'apprentissage supervisé

Approche discriminante :

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



nous n'avons pas d'hypothèse sur le modèle sous-jacent à X mais nous étudions comment séparer ses valeurs.

Exemple : machine à vecteur support (SVM), regression logistique.

Trois grandes méthodes pour l'apprentissage supervisé

Approche par exemplification (non paramétriques) :

On possède une série d'exemples de couples assignant une observation X à une cible Y :

$$D_{train} = \{ (x_1, y_1), \dots, (x_N, y_N) \}$$

- Pour une nouvelle observation x^* , on cherche les observations X de la base d'entraînement les plus proches de x^* .
- On assigne à x^* le y correspondant aux X les plus proches.
Exemple : K-plus-proche-voisin, random forest, les *arbres de décision*

Prétraitement de données

Les variables mesurées dans une expérimentation peuvent être catégoriques, quantitatives, discrètes ou continues.

Catégorique (qualitative) ou quantitative ?

Les variables quantitatives mesurent donc des “quantités” comme :

- Le poids d’une voiture, en kilogrammes.
- Le temps de réalisation d’une tâche en secondes.
- Un âge, une distance, une durée, une note etc...

Prétraitement de données

Une variable qui n'est pas quantitative est qualitative.

Les variables catégoriques (ou qualitatives) mesurent juste des “états”, des catégories. Il n’y a pas d’échelle de valeurs :

Une couleur, un diplôme, un prénom, Oui ou non, Homme ou femme, Code postal, numéro de téléphone.

Le fait que la variable soit numérique, n’implique pas nécessairement que ce soit une variable quantitative.

Prétraitement de données

Discrète ou continue ? une variable quantitative peut être discrète ou continue.

Une variable discrète a une valeur finie. Il est possible de les énumérer (" 1, 2, 3,...").

- **Le nombre d'items dans une liste.**
- **Le nombre de personnes dans une salle.**

Une variable continue peut prendre, en théorie, une infinité des valeurs, formant un ensemble continu.

- **Le temps de réalisation d'une tâche.**
- **La taille, le poids d'une personne.**
- **La vitesse d'une voiture.**

Prétraitement de données

Jusqu'à maintenant, on a supposé que les entrées x prenaient la forme d'un vecteur dans \mathbb{R}^D

Chaque élément x_i de x est un scalaire réel.

— **Quoi faire lorsque ce n'est pas le cas ?**

Une phase de *prétraitement* doit être suivie afin de convertir les entrées en vecteur.

Prétraitement de données

Une **donnée catégorique** prend une valeur parmi un ensemble de symboles fini Ω

Exemples :

- le sexe d'une personne : $\Omega = \{ \text{'femme', 'homme'} \}$
- la réponse à un sondage : $\Omega = \{ \text{'oui', 'non', 'peut-être'} \}$
- un mot : $\Omega = \{ \text{'le', 'la', ... } \}$
- etc.

Prétraitement de données

On convertit sous une forme vectorielle appelée **one-hot**

- crée un vecteur de taille $|\Omega|$ et rempli de 0s
- on associe chaque position dans le vecteur à un symbole différent.
- on assigne à 1 l'élément à la position du symbole observé

Exemple :

- réponse au sondage observée : 'non'
- positions assignées dans cet ordre : 'oui', 'non', 'peut-être'
- vecteur **one-hot** généré sera : (0,1,0)

Prétraitement de données

Exemple :

{‘femme’, ‘homme’} {‘oui’, ‘non’, ‘peut-être’}

↓ ↙ ↓ ↘

(0.3, -5.0, ‘homme’, 2.5, ‘oui’, ‘oui’, ‘peut-être’)



(0.3, -5.0, 0, 1, 2.5, 1, 0, 0, 1, 0, 0, 0, 0, 1)

Prétraitement de données

Donnée manquante :

Si une donnée catégorique est manquante, une approche simple est de laisser tous les éléments du vecteur à 0

- Pour une donnée réelle, on ne peut simplement laisser à 0
 - si $x_i = 0$, est-ce que la donnée originale était 0 ou était manquante ?
- On ajoute plutôt une entrée binaire indiquant les valeurs manquantes (seulement pour les x_i qui peuvent manquer)
 - ☐ (0.5, -0.8, ?, 2.1) devient (0.5, -0.8, **0**, **1**, 2.1)
 - ☐ (1.1, 0.2, **3.8**, -0.7) devient (0.5, -0.8, **3.8**, **0**, 2.1)

Prétraitement de données

Normalisation :

Il est souvent préférable de normaliser les données réelles, afin qu'elle prennent des valeurs «proches» de 0

o des valeurs trop élevées pourraient créer des instabilités numériques

- Suffit de normaliser les données, en soustrayant la moyenne et divisant par l'écart-type

o cette opération est appliquée individuellement pour chaque x_j de \mathbf{x} , c'est-à-dire chaque colonne de la matrice \mathbf{X}

Prétraitement de données

Normalisation

Exemple : L'étude comporte trois échantillons pour lesquelles on dispose de la surface totale de l'exploitation, ainsi que de l'âge et du revenu de l'exploitant.

| | Age | Revenu | Surface |
|---|-----|---------|---------|
| 1 | 30 | 290 000 | 20 |
| 2 | 50 | 300 000 | 30 |
| 3 | 52 | 320 000 | 28 |

Pour déterminer quels sont les échantillons qui se ressemblent le plus, on utilise dans un premier temps la distance euclidienne!

Prétraitement de données

Normalisation

| | Age | Revenu | Surface |
|---|-----|---------|---------|
| 1 | 30 | 290 000 | 20 |
| 2 | 50 | 300 000 | 30 |
| 3 | 52 | 320 000 | 28 |

| | Age | Revenu | Surface |
|---|-------|--------|---------|
| 1 | -1.15 | -0.85 | -1.13 |
| 2 | 0.49 | -0.19 | 0.75 |
| 3 | 0.65 | 1.11 | 0.37 |

| | 1 | 2 | 3 |
|---|----------------|----------------|---|
| 1 | 0 | | |
| 2 | 10^4 | 0 | |
| 3 | $3 \cdot 10^4$ | $2 \cdot 10^4$ | 0 |

| | 1 | 2 | 3 |
|---|------|------|---|
| 1 | 0 | | |
| 2 | 2.70 | 0 | |
| 3 | 2.35 | 1.36 | 0 |

les deux échantillons les plus proches sont : 1 et 2.

Après Normalisation, les échantillons 2 et 3 sont plus proches.

Prétraitement de données

Normalisation :

La normalisation des données donne un poids *égal* à toutes les variables dans le calcul des distances. Si cette égalité de traitement des variables évite tout effet d'échelle, elle a aussi pour conséquence de ne pas distinguer les variables portant une information importante pour l'objectif de classification des variables non pertinentes pour cet objectif. Une fois les variables normalisées, il semble donc préférable d'utiliser ensuite la distance euclidienne pondérée.

Comparaison d'Algorithmes

Comment déterminer si un algorithme A est meilleur qu'un algorithme B ?

Après un bon apprentissage des deux algorithmes, on regarde l'erreur de test (notre mesure de leur performance de généralisation)

Est-ce que l'algorithme ayant l'erreur de test la plus basse est nécessairement (strictement) meilleur ?

peut-être que l'algorithme a «été chanceux» et que la conclusion aurait été différente sur un autre ensemble de test; il avait plus d'exemples sur lesquelles l'algorithme se comporte bien.

Solution : En plus l'erreur moyenne sur l'ensemble de test, **définition de « intervalle de confiance »**

Comparaison d'Algorithmes

intervalle de confiance :

Intervalle de valeurs dans lequel on pense qu'il contient avec une grande probabilité la vraie performance de généralisation.

- **e.g. l'intervalle tel que la probabilité qu'il contienne la vraie performance de généralisation est de 95%.**

(95% valeur du niveau de confiance utilisé souvent)

- Si les intervalles de deux algorithmes ne se chevauchent pas, ils ont probablement des performances différentes (donc je dois choisir le meilleur), autrement ils sont presque équivalents.

Comparaison d'Algorithmes

intervalle de confiance : L'idée derrière l'intervalle de confiance est l'utilisation d'un résultat statistique. Calculer l'erreur moyenne sur plusieurs ensembles de test différents, l'histogramme de ces erreurs ressemblera à la densité d'une gaussienne ; ayant la moyenne μ et la variance σ_{ste}^2 :

$$\mu = \frac{1}{|\mathcal{D}_{test}|} \sum_{(\mathbf{x}_n, t_n) \in \mathcal{D}_{test}} L(t_n, y(\mathbf{x}_n))$$

**Fonction de coût (perte ou erreur) vaut :
0 si $t_n = y(\mathbf{x}_n)$ et 1 sinon.**

Comparaison d'Algorithmes

intervalle de confiance :

Comment calculer cet intervalle ?

$$\sigma^2 = \frac{1}{|\mathcal{D}_{\text{test}}| - 1} \sum_{(\mathbf{x}_n, t_n) \in \mathcal{D}_{\text{test}}} (L(t_n, y(\mathbf{x}_n)) - \mu)^2$$

standard error : $\sigma_{\text{ste}}^2 = \frac{\sigma^2}{|\mathcal{D}_{\text{test}}|}$

Remarque : plus je vais avoir d'exemples de test, plus je vais avoir de certitude sur la performance de mon algorithme d'apprentissage (cad : intervalle de confiance plus petit).

Comparaison d'Algorithmes

intervalle de confiance :

La probabilité que la vraie erreur de généralisation se trouve dans l'intervalle suivant est de 95% :

$$\mu \pm 1.96\sigma_{ste}$$

Ou encore, on peut dire que 95% de la probabilité de se trouve dans cet intervalle,

Exemple :

$$\mu = 0$$

$$\sigma_{ste} = 1$$

