eXplainable Artificial Intelligence

Cours 7 – mardi 6 novembre 2023

Marie-Jeanne Lesot
Christophe Marsala
Jean-Noël Vittaut
Gauvain Bourgne

LIP6 – Sorbonne Université

XAI – 2023-2024

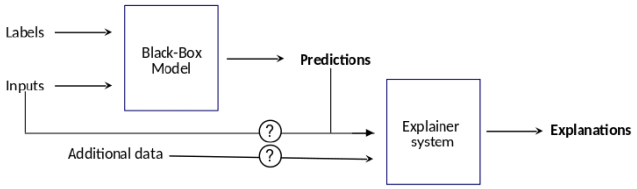
C'est quand on a raison qu'il est difficile de prouver qu'on n'a pas tort.

Pierre Dac

Plan du cours

Contexte

- Un modèle déjà construit : $f : \mathcal{D} \to \mathcal{Y}$
  - approche agnostique : on n'a que f (et on ne sait rien de plus)
  - boîte noire
  - hypothèse : on peut utiliser f autant que l'on veut
- Un exemple d'intérêt : $\mathbf{x} \in \mathcal{D}$
  - exemple à classer et pour lequel on voudrait une explication locale
- Approches post-hoc



Source : (Laugel, 2018)

Les approches par modèle de substitution (surrogate)

- But : étant donné f (boîte noire), trouver un modèle interprétable à lui substituer pour fournir une explication à la décision $f(\mathbf{x})$ donnée pour $\mathbf{x}$.
  $\Longrightarrow$ substitut (surrogate) à f
- Cadre :
  - approche agnostique
  - approche globale ou locale
  - approche simple et interprétable
- Différentes approches
  - explications par "feature importance" : LIME, SHAP,...
  - explications par des règles : LORE, TREPAN, FoilTree, ANCHOR...

Rappels : LIME

- revoir le cours 3
- LIME : Local Interpretable Model-agnostic Explanations
  (Ribeiro et al., 2016)
  - génération d'un échantillon autour du centre des données
  - pondération des exemples selon leur proximité à $\mathbf{x}$
  - mise en évidence des caractéristiques (features) importantes
- Etapes principales :
  1. créer une base d'apprentissage étiquetée grâce à f
  2. pondérer les exemples à l'aide d'une mesure de proximité à $\mathbf{x}$
  3. apprendre un modèle interprétable g
  4. produire une explication pour $f(\mathbf{x})$ grâce à g
- Problèmes
  - aspect global de l'explication
  - interprétabilité ("peut mieux faire")

Une approche par substitution et explication contre-factuelle

- L'approche LORE : LOcal Rule-based Explainer (Guidotti et al., 2019)
- Combinaison des approches par substitution et contre-factuelle
  1. créer une base d'apprentissage dans le voisinage de $\mathbf{x}$ et dont la classification par f couvre les 2 classes
  2. apprendre un modèle interprétable g :
     $\Longrightarrow$ choix de l'arbre de décision
  3. produire une explication pour $f(\mathbf{x})$ grâce à g
     - explication contre-factuelle dérivée de la structure de g
- Avantages
  - pas d'hyperparamètre lié au nombre de features ($\neq$ LIME)
  - pas d'a priori (par exemple : discrétisation ($\neq$ ANCHOR))

Approche LORE : l'algorithme

1. Créer une base d'apprentissage $\mathbf{X}_\mathbf{x}$ dans le voisinage de $\mathbf{x}$ et dont la classification par f couvre les 2 classes
   - génération de $\mathbf{X}_\mathbf{x}$ par algorithme génétique
   - 2 étapes : exemples positifs puis exemples négatifs (selon f)
   - fonction de fitness et opérateurs dédiés
2. Apprendre un modèle interprétable g
   - construction d'un arbre de décision g à partir de $\mathbf{X}_\mathbf{x}$
3. Produire une explication pour $f(\mathbf{x})$ grâce à g
   - explication contre-factuelle dérivée de la structure de g
   - trouver la branche de g pour $\mathbf{x}$: classe $g(\mathbf{x})$
   - recenser les branches de g associées à la classe $\overline{g(\mathbf{x})}$
   - en déduire une explication pour $f(\mathbf{x})$

Génération de $\mathbf{X}_\mathbf{x}$

- Algorithmes génétiques (Holland, 1975)
  - algorithmes évolutionnaires
- Approche inspirée par des lois de l'évolution
  - population d'individus qui évoluent de génération en génération
    - croisements entre individus
    - mutations d'individus : saut génétique
  - mesure de performance d'un individu : mesure de fitness
- Méthode d'optimisation d'une mesure de fitness M
  - trouver un optimum de M par une recherche aléatoire
  - croisement : mélange de propriétés de 2 individus pour tenter d'augmenter M
  - mutation : changer légèrement un individu pour tenter d'augmenter M

Algorithmes génétiques : éléments de base

- Contexte
  - soit la fonction $M : \mathcal{E} \rightarrow \mathbb{R}$ à optimiser (max)
  - $\mathcal{E}$: ensemble de solutions possibles pour M
- Population P_0 à l'intant t : sous-ensemble de $\mathcal{E}$ de taille donnée
- Chromosome : représentation d'élément de $\mathcal{E}$
  $\longrightarrow$ représenté pour l'approche génétique (vecteur de $\mathcal{D}$)
- Opérations possibles
  - croisement (cross-over) : $c : \mathcal{D} \times \mathcal{D} \rightarrow \mathcal{D}$
  - mutation : $m : \mathcal{D} \rightarrow \mathcal{D}$
- Propriétés
  - aucune contrainte sur M
  - exploration aléatoire et guidée de l'espace des solutions
  - minimiser le risque de tomber dans un optimum local

Algorithmes génétiques : l'algorithme général

1. étant donné
   - M la fonction à optimiser
   - taille de la population N
   - un taux de croisement pc et un taux de mutation pm
2. soit P_0 une population initiale à $t = 0$
3. sélectionner des chromosomes dans la population courante P_t
   - sélection selon leur valeur pour M
4. effectuer des croisements et des mutations pour obtenir P_{t+1}
5. recommencer en 3

Algorithmes génétiques : usage dans LORE (1)

- Génération de $\mathbf{X}_\mathbf{x}$ en 2 étapes
  1. générer des exemples proches de $\mathbf{x}$ et de même classe ($f(\mathbf{x})$)
  2. générer des exemples proches de $\mathbf{x}$ et de classe différente
- Mesures de fitness
  1. $M^{\mathbf{x}}_{=}(\mathbf{z}) = I_{f(\mathbf{x})=f(\mathbf{z})} + (1 - d(\mathbf{x}, \mathbf{z})) - I_{\mathbf{x}=\mathbf{z}}$
  2. $M^{\mathbf{x}}_{\neq}(\mathbf{z}) = I_{f(\mathbf{x})\neq f(\mathbf{z})} + (1 - d(\mathbf{x}, \mathbf{z})) - I_{\mathbf{x}=\mathbf{z}}$

  avec :
  - $I_{true} = 1$ et $I_{false} = 0$
  - $d : \mathcal{E} \times \mathcal{E} \rightarrow [0, 1]$ une mesure de la distance
    $d(\mathbf{x}, \mathbf{z}) = \frac{h}{d}\ \text{match}(\mathbf{x}, \mathbf{z}) + \frac{d-h}{d}\ \text{norm}(\mathbf{x} - \mathbf{z})$
    $\text{match}(\mathbf{x}, \mathbf{z})$: comparaison sur les h attributs catégoriels
    $\text{norm}(\mathbf{x}, \mathbf{z})$: norme euclidienne sur les $d - h$ attributs numériques
- Interprétation de la fitness : on cherche des exemples $\mathbf{z}$
  - proches de $\mathbf{x}$ $(1 - d(\mathbf{x}, \mathbf{z}))$ et différents de $\mathbf{x}$ $(-I_{\mathbf{x}=\mathbf{z}})$
  - avec la même classe par f ou une classe différente

Algorithmes génétiques : usage dans LORE (2)

- Population initiale : duplication de l'exemple $\mathbf{x}$
- Opérations de croisement et de mutation
  - croisement 2-points : tirage aléatoire de 2 attributs entre lesquels tous les attributs seront croisés
  - mutation : tirage aléatoire d'un attribut
    - la valeur de remplacement est tirée aléatoirement suivant la distribution des valeurs dans l'ensemble test
  - Chromosome = exemple

parent 1	25	clerk	10k	yes
parent 2	30	other	5k	no
children 1	25	other	5k	yes
children 2	30	clerk	10k	no

Figure 1: Crossover.

parent	25	clerk	10k	yes
children	27	clerk	7k	yes

Figure 2: Mutation

(Guidotti et al., 2018)

Approche LORE : génération des exemples

Algorithm 2: *GeneticNeigh*(x, *fitness*, b, N, G, pc, pm)

Input : x - instance to explain, b - black box, *fitness* - fitness function, N - population size, G - # of generations, pc - crossover probability, pm - mutation probability

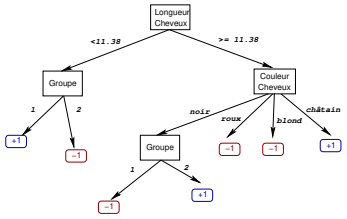
Output : Z - neighbors of x

$P_0 \leftarrow \{x | \forall 1 \ldots N\}$; $i \leftarrow 0$; // population init.
evaluate(P_0, *fitness*, b); // evaluate population
while $i < G$ **do**
  $P_{i+1} \leftarrow$ *select*(P_i); // select sub-population
  $P'_{i+1} \leftarrow$ *crossover*(P_{i+1}, pc); // mix records
  $P''_{i+1} \leftarrow$ *mutate*(P'_{i+1}, pm); // perform mutations
  evaluate(P''_{i+1}, *fitness*, b); // evaluate population
  $P_{i+1} = P''_{i+1}$; $i \leftarrow i + 1$ // update population
end
$Z \leftarrow P_i$ **return** Z;

(Guidotti et al., 2018)

Approche LORE : l'algorithme

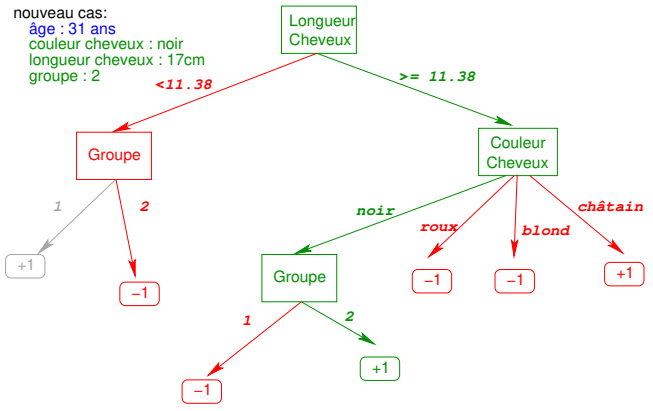
1. Créer une base d'apprentissage $\mathbf{X_x}$ dans le voisinage de $\mathbf{x}$ et dont la classification par f couvre les 2 classes
   - génération de $\mathbf{X_x}$ par algorithme génétique
   - génération en 2 temps : exemples positifs puis exemples négatifs
   - fonction de fitness et opérateurs dédiés
2. Apprendre un modèle interprétable g
   - construction d'un arbre de décision g à partir de $\mathbf{X_x}$
     - algorithme classique



Approche LORE : algorithme

1. Créer une base d'apprentissage $\mathbf{X_x}$ dans le voisinage de $\mathbf{x}$ et dont la classification par f couvre les 2 classes
   - génération de $\mathbf{X_x}$ par algorithme génétique
   - génération en 2 temps : exemples positifs puis exemples négatifs
   - fonction de fitness et opérateurs dédiés
2. Apprendre un modèle interprétable g
   - construction d'un arbre de décision g à partir de $\mathbf{X_x}$
     - algorithme classique
3. Produire une explication pour $f(\mathbf{x})$ grâce à g
   - explication contre-factuelle dérivée de la structure de g
   - trouver la branche de g pour $\mathbf{x}$: classe $g(\mathbf{x})$
   - recenser les branches de g associées à la classe $\overline{g(\mathbf{x})}$
   - en déduire une explication pour $f(\mathbf{x})$

Rappel : classification avec un arbre de décision



Classification : ensemble de règles

- Règle de classification : l'exemple est classé $+1$
  - si Longueur cheveux ≥ 11.38 et Couleur cheveux est noir et Groupe 2 alors classe $+1$
- Règles contre-exemples : celles dont la conclusion n'est pas $+1$
  - si Longueur cheveux < 11.38 et Groupe 2 alors classe -1
  - si Longueur cheveux ≥ 11.38 et Couleur cheveux est roux alors classe -1
  - si Longueur cheveux ≥ 11.38 et Couleur cheveux est blond alors classe -1
  - si Longueur cheveux ≥ 11.38 et Couleur cheveux est noir et Groupe 1 alors classe -1

Approche LORE : explication contre-factuelle

- Soit r la règle déclenchée par $\mathbf{x}$ pour donner $g(\mathbf{x})$
- Soit r' une règle représentant une branche produisant $\overline{g(\mathbf{x})}$
  - on note $n_{r'}$: nombre de tests de r' invalidés par $\mathbf{x}$
- Explication contre-factuelle : règle r_{best} qui minimise $n_{r'}$
  - il peut exister plusieurs telles règles
- Exemple précédent :
  - description :
    - âge=31ans, couleur cheveux=noirs, longueur cheveux=17cm, groupe=2
  - règles r' qui minimisent $n_{r'}$
    - si Longueur cheveux < 11.38 et Groupe 2 alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est roux alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est blond alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est noir et Groupe 1 alors classe -1

Approche LORE : explication contre-factuelle

- Soit r la règle représentant la branche fournissant $g(\mathbf{x})$
- Soit r' une règle représentant une branche telle que $\overline{g(\mathbf{x})}$
  - $n_{r'}$: nombre de tests de r' invalidés par $\mathbf{x}$
- Explication contre-factuelle :
  - règle r_{best} qui minimise $n_{r'}$
- Exemple précédent :
  - description :
    - âge=31ans, couleur cheveux=noirs, longueur cheveux=17cm, groupe=2
  - règles r' qui minimisent $n_{r'}$
    - si Longueur cheveux < 11.38 et Groupe 2 alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est roux alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est blond alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est noir et Groupe 1 alors classe -1

Approche LORE : explication contre-factuelle

- Soit r la règle représentant la branche fournissant $g(\mathbf{x})$
- Soit r' une règle représentant une branche telle que $\overline{g(\mathbf{x})}$
  - $n_{r'}$: nombre de tests de r' invalidés par $\mathbf{x}$
- Explication contre-factuelle :
  - règle r_{best} qui minimise $n_{r'}$
- Exemple précédent :
  - description :
    - âge=31ans, couleur cheveux=noirs, longueur cheveux=17cm, groupe=2
  - règles r' qui minimisent $n_{r'}$
    - si Longueur cheveux < 11.38 et Groupe 2 alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est roux alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est blond alors classe -1
    - si Longueur cheveux ≥ 11.38 et Couleur cheveux est noir et Groupe 1 alors classe -1

Approche LORE : explication contre-factuelle

- Explications pour notre exemple : il est de classe $+1$
  - car la longueur de ses cheveux dépasse 11.38cm
  - car il a les cheveux noirs et pas roux ou blonds
  - car il est dans le groupe 2 et pas le groupe 1

Approche LORE : bilan

- Approche efficace et basée sur l'utilisation d'arbres de décision
- Construction de contre-factuels
  - approche plus générale qui peut s'appliquer dans un cadre non agnostique...
  - ... si le modèle f est un arbre ou une base de règles
- Remarques
  - génération de la base d'exemples
  - hyper-paramètres
  - recherche des règles contre-factuelles (tout ou rien pour le test)
- D'autres pistes avec des surrogates
  - TREPAN, ANCHOR, FoilTREE,...

Approche LORE : quelques références

- "Factual and counterfactual explanations for black box decision making". R. Guidotti, A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, F. Turini. IEEE Intelligent Systems 34 (6), 14-23. 2019.
- "Local rule-based explanations of black box decision systems". R. Guidotti, A. Monreale, S. Ruggieri, D. Pedreschi, F. Turini, F. Giannotti. arXiv 2018.

Plan du cours

Approches par modèle de substitution

Apprentissage de règles floues
- systèmes d'inférence floue
- modèles de Mamdani
- modèles de Takagi & Sugeno
- approches neuro-floues
- arbres de décision flous

Apprentissage inductif flou

- Théorie des sous-ensembles flous pour aider l'apprentissage
  - prise en compte de données numériques, imprécises ou floues
  - mise en œuvre d'un raisonnement flou
- La base d'apprentissage contient des exemples décrits par des données numériques ou des valeurs floues
- Discrétisation / fuzzification des données
  - une valeur floue généralise un ensemble de valeurs précises
  - robustesse

Apprentissage inductif flou : pouvoir de généralisation élevé.

- Une valeur floue est déjà une généralisation

Apprentissage de règles floues

- Étant donné une base de règles floues,
  - apprendre les caractéristiques de chaque règle : optimiser des fonctions d'appartenance
- Étant donné une base d'apprentissage
  - apprendre une base de règles floues : trouver des relations entre les exemples de la base
  - exemple : construction d'une base de règles par arbres de décision flous

Quelques approches floues (1)

- Fuzzy cluster analysis
  - prise en compte d'appartenance graduelle
  - les frontières entre les clusters ne peuvent pas être définies précisément
- Apprentissage de règles floues
  - classification ou régression
  - combinaison avec d'autres approches
    - p.ex. optimisation de fonctions d'appartenance par algorithmes génétiques
  - approches neuro-floues
    - représentation d'un système flou comme un réseau de neurones
    - techniques classiques d'optimisation
    - combine les avantages du flou (interprétabilité) et des réseaux de neurones (flexibilité)

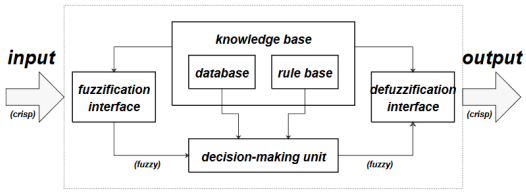
Quelques approches floues (2)

- Apprentissage d'arbres de décision flous
  - exemple type de fuzzification d'un algorithme
  - paramètres : interface avec les représentations floues / vagues
- Règles d'association floues
  - notion floue de support / confiance
  - mesures de qualité
- Représentations floues pour l'apprentissage à partir de cas
  - importance de la similarité
  - agrégation (opérateurs OWA ou approches de fusion)
- Réseaux possibilistes
  - lien possibilistes ($\neq$ probabilistes) entre les informations

Comment rendre flou ?

- Données floues
  - construction de fonctions d'appartenance
  - les données sont floues
    - connaissances subjectives : sensation, sentiment, opinion, perception
    - connaissances vagues : capteurs imprécis, bruit
- Frontières de décision floues
  - les classes ou groupes ne sont pas connus avec précision
  - mal définies ou mal identifiées
- Mesures floues : paramètres de l'algorithme
  - similarité entre individus
  - mesures d'information floues

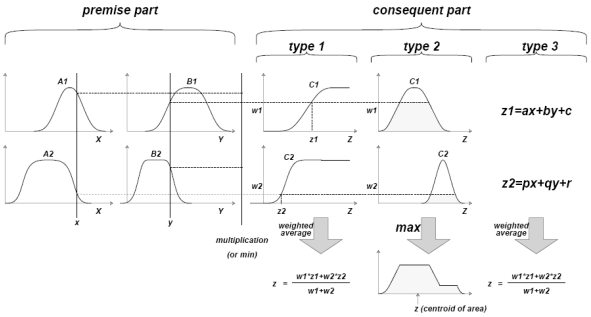
Systèmes d'inférence floue (SIF) à base de règles



- Règles : IF <premisse> THEN <conséquent>
- Selon le type du conséquent :
  - modèle de Mamdani
  - modèle de Takagi-Sugeno

[Jang 94]

SIF : différents modèles



[Jang 94]

- types 1 & 2 : modèle de Mamdani
- type 3 : modèle de Takagi-Sugeno

Principe théorique

(Zadeh, 73)

- Principe : modélisation sous la forme de règles floues

  Si θ vaut environ 0 et si $\dot{\theta}$ vaut environ zéro
  alors F doit être approximativement zéro

  - mesure de θ, $\dot{\theta}$, ... : observations précises
  - modus ponens généralisé (MPG) pour en déduire la commande

- Avantages
  - connaissance mathématique du système non nécessaire
  - simulation de l'expert qui contrôle le système
  - flexibilité et adaptabilité de la commande

Modèle de Mamdani

Règles de type Mamdani

R_1 : IF V_1 is $A_{1,1}$ and ... and V_k is $A_{1,k}$ THEN W is B_1
...
R_p : IF V_1 is $A_{p,1}$ and ... and V_k is $A_{p,k}$ THEN W is B_p

- Variables linguistiques :
  (V_1, X_1, T_1), ..., (V_k, X_k, T_k), (U, Y, T_Y)
- Observation de données précises : $(x_1, \ldots, x_k) \in X_1 \times \cdots \times X_k$
- Conclusion floue :
  - agrégation des conséquents des règles par un "or"
- Sortie précise si nécessaire :
  - $y \in Y$ obtenue par défuzzification :
    centre de gravité, moyenne du noyau, milieu, etc.

Traitement

- On considère
  - une règle $R =$ *Si* V_1 *est* A_1 *et ... et* V_n *est* A_n, *alors* W *est* B
  - l'entrée $x_0 = (x_1, ..., x_n) \in X_1 \times ... \times X_n$
- Etape 1 : application de la règle
  - utilisation des résultats sur le MPG
  - avec une observation précise
  - avec l'"implication" de Mamdani : $f_R(x, y) = \min(x, y)$

$$\begin{aligned} f_{res}(y) &= \sup_{x \in X} \top(f_{A'}(x), \min(f_A(x), f_B(y))) \\ &= \min(f_A(x_0), f_B(y)) \end{aligned}$$

$A = A_1 \times \cdots \times A_n$ défini sur $X = X_1 \times \cdots \times X_n$

Etape 1 : application d'une règle

- On considère
  - une règle $R =$ *Si* V_1 *est* A_1 *et ... et* V_n *est* A_n, *alors* W *est* B
  - l'entrée $x_0 = (x_1, ..., x_n) \in X_1 \times ... \times X_n$

• Calculer la compatibilité avec R
degré d'appartenance de x_0 au sous-ensemble flou prémisse de la règle (défini sur le produit cartésien)

$$\begin{aligned} a_R &= f_A(x_0) \\ &= \min(f_{A_1}(x_1), ..., f_{A_n}(x_n)) \end{aligned}$$

• Calculer le résultat de la règle R
avec la (fausse) implication floue de Mamdani

$$\begin{aligned} f_{resR} : Y &\to [0, 1] \\ y &\mapsto \min(a_R, f_B(y)) \end{aligned}$$

Traitement

- On considère
  - un ensemble de règles
    $R_1 =$ *Si* V_1 *est* A_1 *et ... et* V_n *est* A_n, *alors* W *est* B
    $R_2 =$ *Si* V_1 *est* A'_1 *et ... et* V_n *est* A'_n, *alors* W *est* B'
    $R_3 =$ *Si* V_1 *est* A''_1 *et ... et* V_n *est* A''_n, *alors* W *est* B''
  - l'entrée $x_0 = (x_1, ..., x_n) \in X_1 \times ... \times X_n$
- Etape 2 : combinaison des résultats de toutes les règles
  - $\Rightarrow$ sous-ensemble flou décrivant la commande à appliquer

Etape 2 : combinaison des règles

- Calcul de la compatibilité avec R

$$\begin{aligned} a_R &= f_A(x_0) \\ &= \min(f_{A_1}(x_1), ..., f_{A_n}(x_n)) \end{aligned}$$

- Calcul du résultat de la règle R

$$\begin{aligned} f_{resR} : Y &\rightarrow [0,1] \\ y &\mapsto \min(a_R, f_B(y)) \end{aligned}$$

- Combinaison des sef
  obtenus avec chacune des règles, par l'opérateur max

$$\begin{aligned} f_{res} : Y &\rightarrow [0,1] \\ y &\mapsto \max_R f_{resR}(y) \end{aligned}$$

Etapes 1 et 2 : exemple

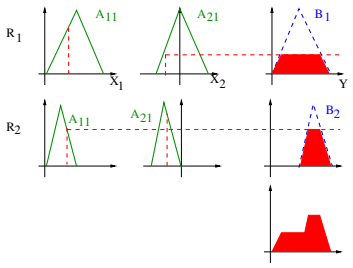
R : si V_1 est A_1 et V_2 est A_2, alors W est B

- Compatibilité avec la règle R :
  $a_R = \min(f_{A_1}(x_1), ..., f_{A_n}(x_n))$
- Résultat de la règle R

$$\begin{aligned} f_{resR} : Y &\rightarrow [0,1] \\ y &\mapsto \min(a_R, f_B(y)) \end{aligned}$$

- Combinaison des règles

$$\begin{aligned} f_{res} : Y &\rightarrow [0,1] \\ y &\mapsto \max_R f_{resR}(y) \end{aligned}$$

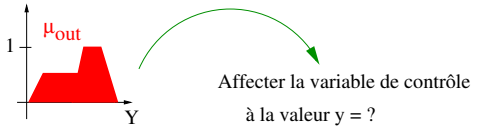


Traitement

- On considère
  - un ensemble de règles
    $R_1 =$ *Si* V_1 *est* A_1 *et ... et* V_n *est* A_n, *alors* W *est* B
    $R_2 =$ *Si* V_1 *est* A'_1 *et ... et* V_n *est* A'_n, *alors* W *est* B'
    $R_3 =$ *Si* V_1 *est* A''_1 *et ... et* V_n *est* A''_n, *alors* W *est* B''
  - l'entrée $x_0 = (x_1, ..., x_n) \in X_1 \times ... \times X_n$
- Etape 2 : combinaison des résultats de toutes les règles
  - $\Rightarrow$ sous-ensemble flou décrivant la commande à appliquer
- Etape 3 : défuzzification
  - $\Rightarrow$ on en déduit une valeur précise

Etape 3 : défuzzification

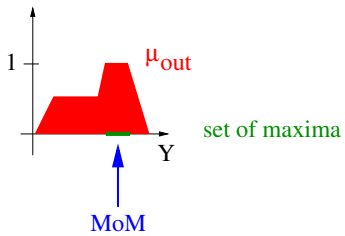
- Transformer le sous-ensemble flou obtenu en une valeur précise



- Méthodes principales
  - maximum : choisir y qui maximise f_{res}
  - moyenne des maxima
  - centre de gravité

Etape 3 : moyenne des maxima

- Définition : moyenne des y qui maximisent f_{res}

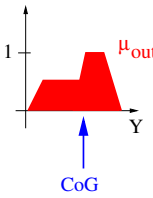


- Difficultés
  - suppose que l'on ait un intervalle
  - peut conduire à un contrôleur discontinu : le résultat dépend seulement de la règle qui a le plus haut degré de compatibilité

Etape 3 : centre de gravité

- Définition : centre de gravité de l'aire sous la courbe

$$\text{CoG} = \frac{\int_{y \in Y} y f_{res}(y) dy}{\int_{y \in Y} f_{res}(y) dy}$$



- Caractéristiques
  - conduit le plus souvent à un comportement continu
  - dépend de plusieurs règles, pondérées selon leurs degrés de compatibilité
  - coût de calcul élevé
  - difficile à justifier sémantiquement

Contrôleur de Takagi-Sugeno

- Base de règles de forme différente : conclusion précise

R : *si* V_1 *est* A_1 *et ... et* V_n *est* A_n, *alors* $y = g(x_1, ..., x_n)$

- $g : X_1 \times ... \times X_n \to Y$
- souvent g est linéaire : $g(x_1, ..., x_n) = a_0 + a_1 x_1 + ... a_n x_n$

Modèle de Takagi-Sugeno

Règles de type Takagi-Sugeno [Takagi & Sugeno 85]

R_1 : IF V_1 is $A_{1,1}$ and ... and V_k is $A_{1,k}$ THEN $y_1 = g_1(x_1, \ldots, x_k)$
...
R_p : IF V_1 is $A_{p,1}$ and ... and V_k is $A_{p,k}$ THEN $y_p = g_p(x_1, \ldots, x_k)$

avec $g_i : X_1 \times \cdots \times X_k \longrightarrow Y$ pour tout $i = 1, ..., p$
en général : $g_i(x_1, \ldots, x_k) = p_{i,0} + p_{i,1} x_1 + \ldots + p_{i,k} x_k$ avec $p_{i,j} \in \mathbb{R}$

- Variables linguistiques : (V_1, X_1, T_1), ..., (V_k, X_k, T_k), (U, Y, T_Y)
- Observation de données précises : $(x_1, \ldots, x_k) \in X_1 \times \cdots \times X_k$
- Conclusion précise : $y \in Y$
  - moyenne pondérée des y_i

Comparaison

Contrôleur de Mamdani	Contrôleur de Takagi-Sugeno
▶ Prémisses et conclusions linguistiques	▶ Conclusion arithmétique
▶ Avantages • quand le modèle est imprécis, voire inconnu • facile à appréhender par un non-spécialiste	▶ Avantages • preuve formelle de stabilité • extension des concepts de l'automatique linéaire au cas non-linéaire
▶ Inconvénients • manque de preuves formelles (stabilité, optimalité)	▶ Inconvénients • nécessite la connaissance d'un bon modèle

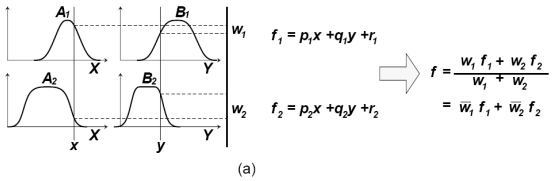
Approches neuro-floues

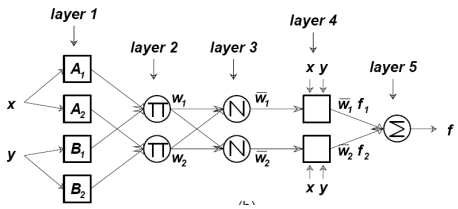
Idée des approches neuro-floues

Combiner une représentation à base de règles (SIF) avec un réseau de neurones

- Système neuro-flou coopératif :
  - réseau de neurones pour apprendre des sous-parties du SIF
  - une fois fait, on ne garde que le système flou
- Système neuro-flou combiné
  - les 2 approches se combinent
  - un réseau détermine les entrées ou les sorties d'un système flou
- Système neuro-flou hybride
  - pour déterminer les paramètres du système flou
  - réseau + approches d'optimisation (descente de gradient,...)

Exemple de système neuro-flou hybride : ANFIS





[Jang 94]

Adaptive Network based Fuzzy Inference System (pour TS)

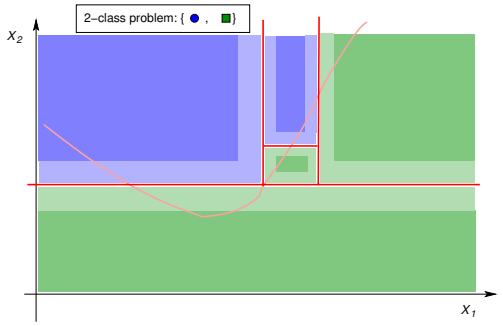
Construction / mise au point de SIF

- Selon le type, spécificités des approches :
  - modèle de Mamdani : arbres de décision flous, construction de règles, ...
  - modèle de Takagi-Sugeno : approches neuro-floues

Conclusions sur les SIF et les approches neuro-floues

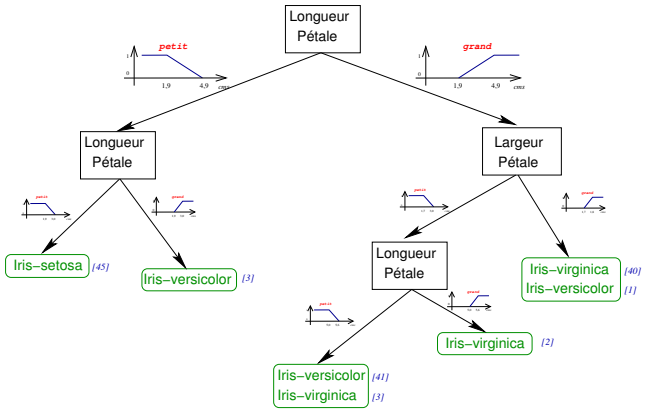
- Un système d'inférence flou est un approximateur universel
  - on peut trouver un SIF pour représenter toute fonction
  - modèle interprétable
- Système neuro-flou :
  - combiner un SIF et une représentation par réseau de neurones
  - plusieurs approches existent : ANFIS, NEFCON, NEFCLASS,...
  - utilisé en contrôle flou (cf. cours à venir)
  - conception pour hardware

Arbres de décision flous : frontières floues

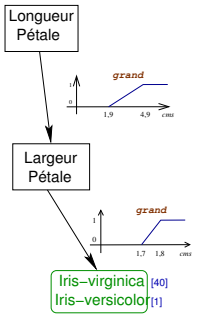


- Amélioration : "fuzzification" des frontières
  - arbres de décision flous

Exemple d'arbre de décision flou

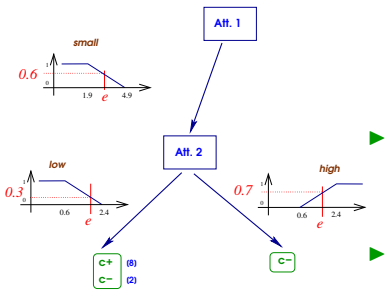


Classification avec un arbre de décision flou



SI
la longueur du pétale est grande
ET
la largeur du pétale est grande
ALORS
0.98 | Iris–Virginica
+ 0.02 | Iris–Versicolor

Classification d'un exemple e



- Deux chemins sont utilisés :
  - chemin 1 : degré $\top(0.6, 0.3)$
    - c_+ (0.8)
    - c_- (0.2)
  - chemin 2 : degré $\top(0.6, 0.7)$
    - c_- (1)
- Ensemble des chemins :
  - c_+ : $0.8 * \top(0.6, 0.3)$
  - c_- : $\perp\big(0.2 * \top(0.6, 0.3), \top(0.6, 0.7)\big)$
- Au final :
  - sous-ensemble flou sur l'ensemble des classes
  - exemple : $\{0.24|c_+, 0.6|c_-\}$
- Paramètre : choix du couple t-norme / t-conorme

Extension d'une approche d'apprentissage

- Approche classique
  - validation formelle : théorie de l'information
- Caractérisation des données à prendre en compte
  - hypothèses ?
- Comment réaliser une extension ?
  - qu'est-ce qui doit être étendu ?
  - comment est-ce que cela doit être étendu ?
  - quelles propriétés peuvent être conservées ?
- Quelles spécificités de l'algorithme ?
  - comment évaluer l'impact de l'extension ? la valider ?
  - quels apports ?
- Etude de cas : les arbres de décision

Entropie d'événements flous

- Extension de l'entropie de Shannon aux événements flous
  - $v_1, \ldots, v_m$: m valeurs de l'attribut A
  - $c_1, \ldots, c_K$: K valeurs pour la classe C

$$H_E(C|A) = -\sum_{j=1}^{m} p^*(v_j) \sum_{k=1}^{K} p^*(c_k|v_j) \log(p^*(c_k|v_j))$$

  - avec la probabilité d'un événement flou e d'un ensemble dénombrable X muni d'une probabilité p, définie par (Zadeh, 65) :

$$p^*(e) = \sum_{x \in X} \mu_e(x) p(x)$$

- Mesure du lien entre A et la classe C