

Wnioskowanie Boolowskie i teoria zbiorów przybliżonych

4 Zbiory przybliżone

- Wprowadzenie do teorii zbiorów przybliżonych
- Złożoność problemu szukania reduktów

5 Wnioskowanie Boolowskie w obliczaniu reduktów i reguł decyzyjnych

- Metody wnioskowań Boolowskich w szukaniu reduktów
- Systemy decyzyjne oparte o zbiory przybliżone

6 Metoda drzew decyzyjnych

- Wprowadzenie
- Konstrukcja drzew decyzyjnych

7 Problem dyskretyzacji

- Przypomnienia podstawowych pojęć
- Problem dyskretyzacji
- Dyskretyzacja metodą wnioskowania Boolowskiego

Definition

Algebra Boola Jest to struktura algebraiczna

$$\mathcal{B} = (B, +, \cdot, 0, 1)$$

spełniająca następujące aksjomaty

- **Przemienność:**

$$(a + b) = (b + a) \text{ oraz } (a \cdot b) = (b \cdot a)$$

- **Rozdzielność:**

$$a \cdot (b + c) = (a \cdot b) + (a \cdot c), \text{ oraz} \\ a + (b \cdot c) = (a + b) \cdot (a + c)$$

- **Elementy neutralne:**

$$a + 0 = a \text{ oraz } a \cdot 1 = a$$

- **Istnienie elementu komplementarnego:**

$$a + \bar{a} = 1 \text{ and } a \cdot \bar{a} = 0$$

*) Czasem negacja jest dodana do sygnatury algebry Boola.

Przykłady Algebry Boola

- 1 Algebra zbiorów: $\mathbb{P}(X) = (2^X, \cup, \cap, -, \emptyset, X)$;
- 2 Rachunek zdań

$$(\{\text{zdania logiczne}\}, \vee, \wedge, \neg, \perp, \top)$$

- 3 Binarna algebra Boola $\mathcal{B}_2 = (\{0, 1\}, +, \cdot, 0, 1)$ to jest najmniejsza, ale najważniejsza algebra Boola w zastosowaniu.

x	y	$x + y$	$x \cdot y$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

x	$\neg x$
0	1
1	0

Przykłady zastosowań:

- projektowanie układów scalonych;
- rachunek zdań.

Właściwości algebr Boola

Łączność (ang. Associative law):

$$(x + y) + z = x + (y + z) \text{ oraz } (x \cdot y) \cdot z = x \cdot (y \cdot z)$$

Idempotence: $x + x = x$ oraz $x \cdot x = x$ (*dual*)

Działania z 0 i 1: $x + 1 = 1$ oraz $x \cdot 0 = 0$ (*dual*)

Pochłanianie (ang. Absorption laws):

$$(y \cdot x) + x = x \text{ oraz } (y + x) \cdot x = x$$
 (*dual*)

Inwolucja (ang. Involution laws): $\overline{\overline{x}} = x$

Prawa DeMorgana (ang. DeMorgan's laws):

$$\neg(x + y) = \neg x \cdot \neg y \text{ oraz } \neg(x \cdot y) = \neg x + \neg y$$
 (*dual*)

Prawa konsensusu (ang. Consensus laws):

$$(x + y) \cdot (\bar{x} + z) \cdot (y + z) = (x + y) \cdot (\bar{x} + z) \text{ oraz} \\ (x \cdot y) + (\bar{x} \cdot z) + (y \cdot z) = (x \cdot b) + (\bar{x} \cdot z)$$

Zasada dualności: Każda algebraiczna równość pozostaje prawdziwa jeśli zamieniamy operatory $+$ na \cdot , \cdot na $+$, 0 na 1 oraz 1 na 0 .

Funkcje Boolowskie

- Każde odwzorowanie $f : \{0, 1\}^n \rightarrow \{0, 1\}$ nazywamy (**zupelną**) **funkcją Boolowską**.
- Funkcje Boolowskie \equiv **formuły Boolowskie**:
 - Stałe, zmienne, operatory \neg , $+$ oraz \cdot .
 - Literały, mintermy (jednomiany), maxtermy, ...
 - Kanoniczna postać dyzjunkcyjna (DNF), np.
 $f = xy\bar{z} + x\bar{y}z + \bar{x}yz + xyz$;
 - Kanoniczna postać koniunkcyjna (CNF), np. $f = (x + y + z)(\bar{x} + y)$
- Term $t = x_{i_1} \dots x_{i_m} \bar{x}_{j_1} \dots \bar{x}_{j_k}$ nazywamy **implikantem** funkcji f jeśli

$$\forall \mathbf{a} \in \{0, 1\}^n \quad t(\mathbf{a}) = 1 \Rightarrow f(\mathbf{a}) = 1$$

- **Implikant pierwszy**: jest to implikant, który przestaje nim być po usunięciu dowolnego literału.
- **Kanoniczna postać Blake'a**: każdą funkcję Boolowską można przedstawić jako sumę wszystkich jej implikantów pierwszych:

$$f = t_1 + t_2 + \dots + t_k$$

Przykład

Wiele formuł reprezentuje tę samą funkcję;

$$\phi_1 = xy\bar{z} + x\bar{y}z + \bar{x}yz + xyz$$

$$\phi_2 = (x + y + z)(\bar{x} + y + z)(x + \bar{y} + z)(x + y + \bar{z})$$

$$\phi_3 = xy + xz + yz$$

$xy\bar{z}$ jest implikantem

xy jest implikantem pierwszym

x	y	z	f
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

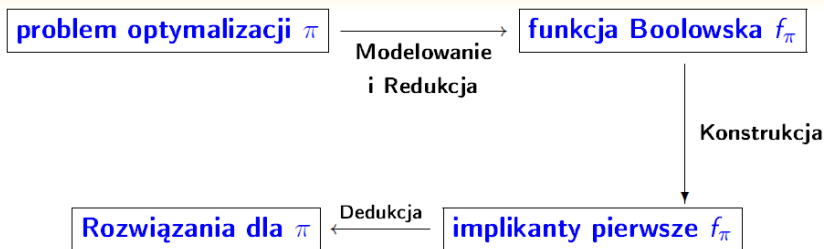
Funkcje monotoniczne

- Niech \prec oznacza relację częściowego porządku w $\{0, 1\}^n$
- Funkcja f jest monotoniczna (niemalejąca) wtw, gdy dla każdego $\mathbf{x}, \mathbf{y} \in \{0, 1\}^n$ jeśli $\mathbf{x} \prec \mathbf{y}$ to $f(\mathbf{x}) \leq f(\mathbf{y})$
- monotoniczne funkcje Boolowskie można zapisać bez użycia negacji.
- term $f' = x_{i_1} \cdot x_{i_2} \dots \cdot x_{i_k}$ nazywamy implikantem pierwszym funkcji monotonicznej f jeśli
 - $f'(\mathbf{x}) \leq f(\mathbf{x})$ dla każdego wektora \mathbf{x} (jest implikantem)
 - każda funkcja większa od f' nie jest implikantem
- Np. funkcja

$$f(x_1, x_2, x_3) = (x_1 + x_2)(x_2 + x_3)$$

posiada 2 implikanty pierwsze: $f_1 = x_2$ i $f_2 = x_1 \wedge x_3$

Wnioskowanie Boolowskie



- 1 **Modelowanie:** Kodowanie problemu za pomocą układu równań Boolowskich;
- 2 **Redukcja:** Sprowadzenie układu równań do pojedynczego równania postaci $f = 0$ lub $f = 1$
- 3 **Konstrukcja:** Znalazienie wszystkich implikantów pierwszych funkcji f (konstrukcja kanonicznej postaci Blake'a);
- 4 **Dedukcja:** Zastosowanie pewnej sekwencji wnioskowań transformujących implikanty pierwsze do rozwiązań problemu.

Boolean Reasoning – Example

Problem:

A, B, C, D are considering going to a party. Social constrains:

- If A goes than B won't go and C will;
- If B and D go, then either A or C (but not both) will go
- If C goes and B does not, then D will go but A will not.

Problem modeling:

$$\begin{aligned}A \rightarrow \bar{B} \wedge C &\iff A(B + \bar{C}) &= 0 \\ \dots &\iff BD(AC + \bar{A}\bar{C}) &= 0 \\ \dots &\iff \bar{B}C(A + \bar{D}) &= 0\end{aligned}$$

- **After reduction:**

$$f = A(B + \bar{C}) + BD(AC + \bar{A}\bar{C}) + \bar{B}C(A + \bar{D}) = 0$$

- **Blake Canonical form:**

$$f = B\bar{C}D + \bar{B}C\bar{D} + A = 0$$

- **Facts:**

$$BD \rightarrow C$$

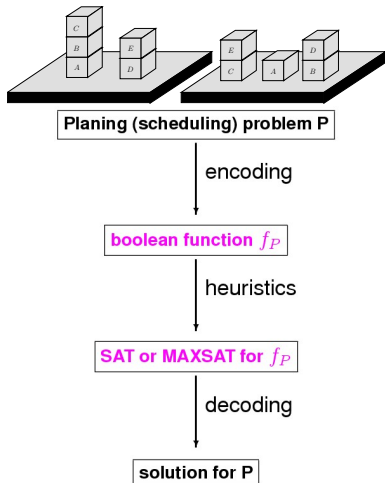
$$C \rightarrow B \vee D$$

$$A \rightarrow 0$$

- **Reasoning:** (theorem proving)
e.g., show that
"nobody can go alone."

Metoda BR w Sztucznej Inteligencji

- **Problem SAT:** czy równanie $f(x_1, \dots, x_n) = 1$ posiada rozwiązanie?
- SAT odgrywa ważną rolę w teorii złożoności (twierdzenie Cooka, dowodzenie NP-zupełności ...)
- Każdy SAT-solver może być używany do rozwiązywania problemu planowania.
- *Blocks world problem:* Po redukcji formuła boolowska nadal zawiera $O(tk^2)$ zmiennych i $O(tk^3)$ klauzuli (k, t - liczby bloków i kroków).
- Np. Dla $k = 15, t = 14$, mamy 3016 zmiennych i 50457 klauzuli



Szukanie reduktów: szkic metody

1 Konstrukcja funkcji odróżnialności:

- Zmienne boolowskie: atrybuty a_1, \dots, a_k ;
- Klauzula:

$$\phi_{u,v} = \bigvee \{a \in A : a(u) \neq a(v)\}$$

- Funkcja rozróżnialności:

$$\mathcal{F}_S(a_1, \dots, a_k) = \bigwedge_{x,y \in U: dec(x) \neq dec(y)} \phi_{x,y}(a_1, \dots, a_k)$$

- 2 przekształcenie funkcji rozróżnialności do postaci DNF
- 3 każdy implikant pierwszy odpowiada jednemu reduktowi;

Przykład tablicy decyzyjnej

Hurt.	Jakość obsługi	Jakość towaru	Obok autostrady?	Centrum?	decyzja
ID	a_1	a_2	a_3	a_4	dec
1	dobra	dobra	nie	nie	strata
2	dobra	dobra	nie	tak	strata
3	bdb	dobra	nie	nie	zysk
4	slaba	super	nie	nie	zysk
5	slaba	niska	tak	nie	zysk
6	slaba	niska	tak	tak	strata
7	bdb	niska	tak	tak	zysk
8	dobra	super	nie	nie	strata
9	dobra	niska	tak	nie	zysk
10	slaba	super	tak	nie	zysk
11	dobra	super	tak	tak	zysk
12	bdb	super	nie	tak	zysk
13	bdb	dobra	tak	nie	?
14	slaba	super	nie	tak	?

Macierz i funkcja odróżnialności

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$\begin{aligned} f = & (\alpha_1)(\alpha_1 \vee \alpha_4)(\alpha_1 \vee \alpha_2)(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_4) \\ & (\alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_3)(\alpha_4)(\alpha_2 \vee \alpha_3)(\alpha_2 \vee \alpha_4) \\ & (\alpha_1 \vee \alpha_3)(\alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_4) \end{aligned}$$

Szukanie reduktów

$$f = (\alpha_1)(\alpha_1 \vee \alpha_4)(\alpha_1 \vee \alpha_2)(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_4) \\ (\alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_3)(\alpha_4)(\alpha_2 \vee \alpha_3)(\alpha_2 \vee \alpha_4) \\ (\alpha_1 \vee \alpha_3)(\alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_4)$$

- usuwanie alternatyw regułą *pochłaniania* (t.j. $p \wedge (p \vee q) \equiv p$):

$$f = (\alpha_1)(\alpha_4)(\alpha_2 \vee \alpha_3)$$

- sprowadzanie funkcji f z postaci **CNF** (koniunkcja alternatyw) do postaci **DNF** (alternatywa koniunkcji)

$$f = \alpha_1\alpha_4\alpha_2 \vee \alpha_1\alpha_4\alpha_3$$

- Każdy składnik jest reduktem! Zatem mamy 2 redukty:

$$R_1 = \{A_1, A_2, A_4\} \text{ i } R_2 = \{A_1, A_3, A_4\}$$

Heurystyka Johnsona: algorytm zachłanny

- Atrybut jest ważniejszy jeśli częściej występuje w klauzulach;
- **Selekcja:** W każdym kroku wybierzmy atrybut, który najczęściej występuje w funkcji rozróżnialności;
- **Usuwanie:** Usuwamy z tej funkcji te klauzule, które zawierają wybrany atrybut;
- Powtarzamy **Selekcja** i **Usuwanie** dopóty, póki funkcja rozróżnialności zawiera jeszcze jakąś klauzulę.

Heurystyka Johnsona

- 1 Znaleźć atrybut, który występuje najczęściej w macierzy rozróżnialności;
 - 2 Usunąć wszystkie pola zawierające wybrany atrybut;
 - 3 Powtarzamy kroki 1 i 2 dopóki wszystkie pola macierzy są puste.
-

Przykład działania

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

- W macierzy nierozróżnialności z poprzedniego przykładu
 a_1 - występuje 23 razy a_2 - występuje 23 razy
 a_3 - występuje 18 razy a_4 - występuje 16 razy
- Jeśli wybieramy a_1 , to po usunięciu pól zawierających a_1 zostało 9 niepustych pól macierzy nierozróżnialności. Wśród nich:
 a_2 - występuje 7 razy a_3 - występuje 7 razy a_4 - występuje 6 razy
- Jeśli wybieramy tym razem a_2 to zostały 2 niepuste pola i wśród nich a_4 jest zdecydowanym faworytem.
- Możemy dostać w ten sposób redukt: $R_1 = \{a_1, a_2, a_4\}$.

Jak dobra jest heurystyka Johnsona?

- Niech $X = \{(u, v) \in U^2 : dec(u) \neq dec(v)\}$;
- Niech $S_{a_i} = \{(u, v) \in X : a_i(u) \neq a_i(v)\}$;
- Niech $C^* \subset A$ będzie minimalnym reduktem, lecz $C = \{a_1, \dots, a_k\}$ będzie reduktem znalezionym przez heurystykę Johnsona;
- Załóżmy, że heurystyka Johnsona musi **płacić** 1zł za każdym razem, gdy dodała a_i do C .
- Rozłóżmy ten koszt tym elementom, które zostały po raz pierwszy pokryte przez zbiór S_{a_i} ;
- Niech c_x = koszt ponoszony przez x . Jeśli x jest pokryty po raz pierwszy przez a_i , to

$$c_x = \frac{1}{|S_{a_i} - (S_{a_1} \cup \dots \cup S_{a_{i-1}})|}$$

Jak dobra jest heurystyka Johnsona?

- Heurystyka Johnsona ponosi łączny koszt $= |C|$. Mamy

$$|C| = \sum_{x \in X} c_x \leq \sum_{a \in C^*} \sum_{x \in S_a} c_x$$

- Gdybyśmy pokazali, że dla dowolnego atrybutu $a \in A$

$$\sum_{x \in S_a} c_x \leq H(|S_a|)$$

gdzie $H(n) = \sum_{i=1}^n \frac{1}{i}$, wówczas możemy oszacować dalej:

$$\begin{aligned} |C| &\leq \sum_{a \in C^*} \sum_{x \in S_a} c_x \leq |C^*| \cdot H(\max\{|S_a| : a \in A\}) \\ &\leq |C^*| \cdot H(|X|) \leq |C^*| \cdot \ln(|X| + 1) \end{aligned}$$

Lemat:

Dla dowolnego atrybutu $a \in A$

$$\sum_{x \in S_a} c_x \leq H(|S_a|)$$

Dowód:

- Niech $u_i = |S_a - (S_{a_1} \cup \dots \cup S_{a_{i-1}})|$, mamy:

$$u_0 = |S_a| \geq \dots \geq u_{i-1} \geq u_i \geq \dots \geq u_k = 0;$$

gdzie k jest najmniejszym indeksem t., że $S_a = S_{a_1} \cup \dots \cup S_{a_k}$

- Zatem

$$\begin{aligned} \sum_{x \in S_a} c_x &= \sum_{i=0}^k (u_{i-1} - u_i) \cdot \frac{1}{|S_{a_i} - (S_{a_1} \cup \dots \cup S_{a_{i-1}})|} \\ &\leq \sum_{i=0}^k (u_{i-1} - u_i) \cdot \frac{1}{u_{i-1}} \leq \sum_{i=0}^k (H(u_{i-1}) - H(u_i)) = H(|S_a|) \end{aligned}$$

Inne heurystyki do szukania implikantów pierwszych

- ILP (Integer Linear Program)
- Symulowane wyżarzanie (simulated annealing)
- Algorytmy genetyczne
- SAT solver
- ???

ILP

Dana jest funkcja $f : \{0, 1\}^n \rightarrow \{0, 1\}$ zapisana w postaci CNF za pomocą zmiennych x_1, \dots, x_n

- 1 Utwórz nowe zmienne $\{y_1, \dots, y_{2n}\}$ odpowiadające literałom $x_1, \bar{x}_1, \dots, x_n, \bar{x}_n$;
- 2 Dla każdej zmiennej x_i , utwórzmy nierówność $y_{2i-1} + y_{2i} \leq 1$
- 3 Zamień każdą klauzulę $\omega_i = (l_{i_1} \vee \dots \vee l_{i_{n_i}})$ na nierówność $y_{i_1} + \dots + y_{i_{n_i}} \geq 1$;
- 4 Utwórzmy układ nierówności z poprzednich punktów:

$$\mathbf{Ay} \geq \mathbf{b}$$

- 5 Zagadnienie programowania liniowego liczb całkowitych (ILP) jest definiowane jako problem szukania

$$\min \sum_{i=1}^n y_i \quad \text{przy ograniczeniu: } \mathbf{Ay} \geq \mathbf{b}.$$

Język deskryptorów (ang. description language)

Dla danego zbioru atrybutów A definiujemy język deskryptorów jako trójkę

$$\mathcal{L}(A) = (\mathbf{D}, \{\vee, \wedge, \neg\}, \mathbf{F})$$

gdzie

- \mathbf{D} jest zbiorem *deskryptorów*

$$\mathbf{D} = \{(a = v) : a \in A \text{ and } v \in Val_a\};$$

- $\{\vee, \wedge, \neg\}$ jest zbiorem standardowych operatorów logicznych;
- \mathbf{F} jest zbiorem formuł logicznych zbudowanych na deskryptorach z \mathbf{D} .
- Dla każdego zbioru atrybutów $B \subseteq A$ oznaczamy:
 $\mathbf{D}|_B = \{(a = v) : a \in B \text{ and } v \in Val_a\}$, czyli zbiór deskryptorów obciętych do B .
 $\mathbf{F}|_B$ zbiór formuł zbudowanych na $\mathbf{D}|_B$.

Semantyka w systemach informacyjnych

Semantyka:

Niech $\mathbb{S} = (U, A)$ będzie tablicą informacyjną. Każda formuła $\phi \in \mathbf{F}$, jest (semantycznie) skojarzona ze zbiorem $[[\phi]]_{\mathbb{S}}$ zawierającym obiekty spełniające ϕ .

Formalnie możemy indukcyjnie definiować semantykę jak następująco:

$$[[a = v]]_{\mathbb{S}} = \{x \in U : a(x) = v\} \quad (1)$$

$$[[\phi_1 \vee \phi_2]]_{\mathbb{S}} = [[\phi_1]]_{\mathbb{S}} \cup [[\phi_2]]_{\mathbb{S}} \quad (2)$$

$$[[\phi_1 \wedge \phi_2]]_{\mathbb{S}} = [[\phi_1]]_{\mathbb{S}} \cap [[\phi_2]]_{\mathbb{S}} \quad (3)$$

$$[[\neg\phi]]_{\mathbb{S}} = U \setminus [[\phi]]_{\mathbb{S}} \quad (4)$$

Każda formuła ϕ może być charakteryzowana przez:

- $length(\phi)$ = liczba deskryptorów w ϕ ;
- $support(\phi) = |[[\phi]]_{\mathbb{S}}|$ = liczba obiektów spełniających formułę

Reguły decyzyjne

Definition

Definicja reguł decyzyjnych Niech $\mathbb{S} = \{U, A \cup \{dec\}\}$ będzie tablicą decyzyjną. Regułą decyzyjną dla \mathbb{S} nazywamy formuły postaci:

$$\phi \Rightarrow \delta$$

gdzie $\phi \in \mathbf{F}_A$ i $\delta \in \mathbf{F}_{dec}$.

Formułę ϕ nazywamy *poprzednikiem* (lub założeniem) reguły r , a δ nazywamy *następnikiem* (lub tezą) reguły r .

Oznaczamy poprzednik i następnik reguły r przez $prev(r)$ oraz $cons(r)$.

Definition

Reguły atomowa: Są to reguły postaci:

$$\mathbf{r} \equiv (a_{i_1} = v_1) \wedge \dots \wedge (a_{i_m} = v_m) \Rightarrow (dec = k) \quad (5)$$

Reguły decyzyjne (c.d.)

Każda reguła decyzyjna r postaci (5) może być charakteryzowana następującymi cechami:

$length(r)$ = liczba deskryptorów występujących w założeniu reguły r

$[r]$ = nośnik reguły r , czyli zbiór obiektów z U spełniających założenie reguły r

$support(r)$ = liczba obiektów z U spełniających założenie reguły: $support(r) = card([r])$

$confidence(r)$ = wiarygodność reguły r : $confidence(r) = \frac{|[r] \cap DEC_k|}{|[r]|}$

Mówimy, że reguła r jest *niesprzeczna* z \mathbb{A} jeśli

$$confidence(r) = 1$$

Minimalne reguły

Definition

Minimalne niesprzeczne reguły: Niech $\mathbb{S} = (U, A \cup \{dec\})$ będzie daną tablicą decyzyjną. Niesprzeczną regułę

$$(a_{i_1} = v_1) \wedge \dots \wedge (a_{i_m} = v_m) \Rightarrow (dec = k)$$

nazywamy *minimalną niesprzeczną regułą decyzyjną* jeśli usunięcie któregokolwiek z deskryptorów spowoduje, że reguła przestaje być niesprzeczna z \mathbb{S} .

Ogólne metody

Klasyfikatory regułowe działają w trzech fazach:

- 1 Faza treningu: Generuje pewien zbiór reguł $RULES(\mathbb{A})$ z danej tablicy decyzyjnej \mathbb{A} .
- 2 Faza selekcji reguł: Szuka w $RULES(\mathbb{A})$ tych reguł, które są wspierane przez obiekt x . Oznaczamy zbiór tych reguł przez $MatchRules(\mathbb{A}, x)$.
- 3 Faza klasyfikacji: wyznacza klasę decyzyjną dla x za pomocą reguł z $MatchRules(\mathbb{A}, x)$ według następującego schematu:
 - 1 Jeśli $MatchRules(\mathbb{A}, x)$ jest pusty: odpowiedź dla x jest "NIEWIEM", tzn. nie mamy podstaw, aby klasyfikować obiekt x do którejkolwiek z klas;
 - 2 Jeśli $MatchRules(\mathbb{A}, x)$ zawiera tylko obiekty z k -tej klasy: wówczas $dec(x) = k$;
 - 3 Jeśli $MatchRules(\mathbb{A}, x)$ zawiera reguły dla różnych klas decyzyjnych: wówczas decyzja dla x określimy za pomocą pewnego, ustalonego schematu głosowania między regułami z $MatchRules(\mathbb{A}, x)$.

Metoda Oparta o Wnioskowanie Boolowskie

- Każda reguła powstaje poprzez skracanie opisu jakiegoś obiektu.
- Redukty lokalne
- Te same heurystyki dla reduktów decyzyjnych.

Przykład tablicy decyzyjnej

Hurt.	Jakość obsługi	Jakość towaru	Obok autostrady?	Centrum?	decyzja
ID	a_1	a_2	a_3	a_4	dec
1	dobra	dobra	nie	nie	strata
2	dobra	dobra	nie	tak	strata
3	bdb	dobra	nie	nie	zysk
4	slaba	super	nie	nie	zysk
5	slaba	niska	tak	nie	zysk
6	slaba	niska	tak	tak	strata
7	bdb	niska	tak	tak	zysk
8	dobra	super	nie	nie	strata
9	dobra	niska	tak	nie	zysk
10	slaba	super	tak	nie	zysk
11	dobra	super	tak	tak	zysk
12	bdb	super	nie	tak	zysk
13	bdb	dobra	tak	nie	?
14	slaba	super	nie	tak	?

Macierz i funkcja lokalnych odróżnialności

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$f_{u_3} = (\alpha_1)(\alpha_1 \vee \alpha_4)(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_2) = \alpha_1$$

Reguły:

$$(a_1 = \text{bdb}) \implies \text{dec} = \text{zysk}$$

Macierz i funkcja lokalnych odróżnialności

M	1	2	6	8
3	a_1	a_1, a_4	a_1, a_2, a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_4	a_1, a_2, a_3
7	a_1, a_2, a_3, a_4	a_1, a_2, a_3	a_1	a_1, a_2, a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	a_1, a_2, a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

$$\begin{aligned}
 f_{u_8} &= (\alpha_1 \vee \alpha_2)(\alpha_1)(\alpha_1 \vee \alpha_2 \vee \alpha_3)(\alpha_1 \vee \alpha_2 \vee \alpha_3 \vee \alpha_4)(\alpha_2 \vee \alpha_3) \\
 &\quad (\alpha_1 \vee \alpha_3)(\alpha_3 \vee \alpha_4)(\alpha_1 \vee \alpha_4) \\
 &= \alpha_1(\alpha_2 \vee \alpha_3)(\alpha_3 \vee \alpha_4) = \alpha_1\alpha_3 \vee \alpha_1\alpha_2\alpha_4
 \end{aligned}$$

Reguły:

- $(a_1 = \text{dobra}) \wedge (a_3 = \text{nie}) \implies \text{dec} = \text{strata}$
- $(a_1 = \text{dobra}) \wedge (a_2 = \text{super}) \wedge (a_4 = \text{nie}) \implies \text{dec} = \text{strata}$