Ćwiczenia z algorytmów wyliczania reguł decyzyjnych

March 2, 2025

Ćwiczenie 3 (1pkt)

Algorytmy wyliczania reguł decyzyjnych

Zadanie do wykonania

1. Czytamy teorię, analizujemy przykłady, w razie problemu ze zrozumieniem wyliczamy reguły na kartce.
2. Wczutyjemy plik SystemDecyzyjny.txt . Ostatni atrybut (kolumna) określa klasę próbki
3. Implementujemy algorytm w Python, wyliczając następujące reguły:
   - reguły LEM2 (1pkt).

Teoria do ćwiczeń z przykładami

Sposoby zapisu deskryptora:

$$(a = a(v))$$

$$(a = v)$$

$$(a, a(v))$$

$$(a, v)$$

Znaczenie: $(a = a(v))$, atrybut a ma wartość v

System Informacyjny: $(U, A)U$ - zbiór obiektów;

A - zbiór atrybutów warunkowych;
Przykład: $(U, A), U = ob1, ob2, ob3, A = a1, a2, a3$

	a_1	a_2	a_3
ob_1	1	2	3
ob_2	3	2	5
ob_3	10	2	17

System Decyzyjny: $(U, A, d)U$ - zbiór obiektów;
A - zbiór atrybutów warunkowych;
d - atrybut decyzyjny
$d \notin A$
Przykład System decyzyjny zapisujemy jako (U, A, d), przyjmijmy,
$U = ob1, ob2, ob3$
$A = a1, a2, a3$
$d \in D = 1, 2$
Przykładowy system decyzyjny zgodny z opisem powyżej, może wygladać następująco,

	a_1	a_2	a_3	d
ob_1	7	2	3	1
ob_2	3	3	5	2
ob_3	10	45	4	1

Zdefiniujmy reguły decyzyjne wzajemnie niesprzeczne

$$(a_1 = 1) \implies (d = 1)$$

$$(a_1 = 2) \wedge (a_2 = 7) \implies (d = 1)$$

$$(\text{pogoda} = \text{słoneczna}) \wedge (\text{żona} = \text{w pracy}) \wedge (\text{czas} = \text{wolny}) \implies (\text{decyzja} = \text{park})$$

$$(\text{pogoda} = \text{słoneczna}) \wedge (\text{żona} = \text{w domu}) \wedge (\text{czas} = \text{wolny}) \implies (\text{decyzja} = \text{dom})$$

Reguły decyzyjne wzajemnie sprzeczne

$$(a_1 = 1) \implies (d = 1)$$

$$(a_1 = 1) \implies (d = 2)$$

$$(\text{pogoda} = \text{słoneczna}) \wedge (\text{żona} = \text{w pracy}) \wedge (\text{czas} = \text{wolny}) \implies (\text{decyzja} = \text{park})$$

$$(\text{pogoda} = \text{słoneczna}) \wedge (\text{żona} = \text{w pracy}) \wedge (\text{czas} = \text{wolny}) \implies (\text{decyzja} = \text{dom})$$

Reguła z alternatywną decyzją

$$(\text{pogoda} = \text{słoneczna}) \wedge (\text{żona} = \text{w pracy}) \wedge (\text{czas} = \text{wolny}) \implies (\text{decyzja} = \text{park}) \vee (\text{decyzja} = \text{dom})$$

1 Przykładowe wyliczanie reguł LEM2 (Learn from Examples by Modules):

Dany mamy system decyzyjny (U, A, d), gdzie $U = o1, o2, .., o7$, $A = a1, a2, .., a5$
d – atrybut decyzyjny

	a_1	a_2	a_3	a_4	a_5	d
o_1	2	6	1	2	3	1
o_2	1	1	1	3	2	1
o_3	2	1	1	2	3	1
o_4	4	1	3	1	2	1
o_5	3	5	2	1	3	2
o_6	3	1	3	1	1	2
o_7	1	1	1	3	1	2

Idea algorytmu

Algorytm polega na tworzeniu pierwszej reguły przez sekwencyjny wybór "najlepszego" elementarnego warunku, przy zachowaniu ustalonych kryteriów. Przykłady treningowe pokryte przez regułę są usuwane z rozważań. Proces tworzenia reguł jest powtarzany iteracyjnie do momentu, gdy pozostają jakieś niepokryte obiekty w systemie treningowym.

Wszelkie konflikty rozwiązywane są hierarchicznie (wybierana jest wartość pierwsza od góry z lewej strony). W praktyce wygląda to tak:

Patrzymy na koncept 1 (koncept jest synonimem klasy decyzyjnej), szukając deskryptora, który występuje najczęściej: W wybranym koncepcie najczęściej występuje deskryptor $(a2 = 1) + powstajezobiektówo2, o3, o4$

Nie tworzy jednak reguły ponieważ w koncepcie 2 mamy sprzeczność. Skupiając uwagę na obiektach do których pasuje $(a2 = 1), czylio2, o3, o4$, szukam kolejnego najlepszego deskryptora, z największym pokryciem w klasie 1. Tym deskryptorem jest $(a3 = 1) + powstajezobiektówo2, o3$, dokładam go do pierwszego deskryptora i tworzę koniunkcję: W wybranym koncepcie najczęściej występuje deskryptor

$$(a_2 = 1) \rightarrow \text{powstaje z obiektów } o_2, o_3, o_4$$

Nie tworzy jednak reguły, ponieważ w koncepcie 2 mamy sprzeczność. Skupiając uwagę na obiektach, do których pasuje $(a_2 = 1)$, czyli o_2, o_3, o_4, szukam kolejnego najlepszego deskryptora, z największym pokryciem w klasie 1. Tym deskryptorem jest $(a_3 = 1) \rightarrow$ powstaje z obiektów o_2, o_3, dokładam go do pierwszego deskryptora i tworzę koniunkcję:

$$(a_2 = 1) \wedge (a_3 = 1),$$

jednak powstała koniunkcja dalej jest sprzeczna.

Z faktu, że powyższa reguła powstała z obiektów o_2, o_3, szukam w nich kolejnego najbardziej licznego deskryptora, tym razem jest nim $(a_1 = 1) \rightarrow$ powstaje z obiektu o_2, dokładam znaleziony deskryptor do budowanej reguły:

$$(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1),$$

sprzeczność nie została usunięta, stąd wybieramy kolejny deskryptor z obiektu o_2, dostajemy $(a_4 = 3)$, dołączamy do naszej reguły:

$$(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1) \wedge (a_4 = 3),$$

koniunkcja jest wciąż sprzeczna, dodajemy do niej kolejny deskryptor postaci $(a_5 = 2)$, dostajemy:

$$(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1) \wedge (a_4 = 3) \wedge (a_5 = 2),$$

ta kombinacja jest niesprzeczna, tworzymy z niej regułę postaci:

$$(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1) \wedge (a_4 = 3) \wedge (a_5 = 2) \implies (d = 1)$$

W koncepcie 1 powstała już reguła z obiektu o_2, stąd przy szukaniu kolejnej skupiamy uwagę na o_1, o_3, o_4, najczęstszym deskryptorem jest $(a_1 = 2)$, który pasuje do obiektów o_1, o_3, ten deskryptor nie jest sprzeczny, stąd powstaje reguła:

$$(a_1 = 2) \implies (d = 1)[2],$$

reguła ma support 2, ponieważ pasuje do dwóch obiektów, o_1 i o_3.

w koncepcie 1, został nam do rozważenia tylko obiekt o4, z którego powstaje niesprzeczna reguła:

$$(a_1 = 4) \implies (d = 1)$$

Następnie tworzymy reguły z konceptu 2: Czyli rozważamy obiekty o_5, o_6, o_7. Najbardziej licznym deskryptorem i pierwszym z brzegu jest $(a_1 = 3)$, do tego jest niesprzeczny, stąd tworzy regułę:

$$(a_1 = 3) \implies (d = 2)[2], \quad \text{pokrywa obiekty } o_5, o_6.$$

Ostatecznie tworzymy regułę z obiektu o_7: Widzimy, że deskryptory:

$$(a_1 = 1), \quad (a_2 = 1), \quad (a_3 = 1), \quad (a_4 = 3)$$

tworzą sprzeczność, dopiero dołożenie deskryptora $(a_5 = 1)$ likwiduje sprzeczność i powstaje reguła:

$$(a_1 = 1) \wedge (a_2 = 1) \wedge (a_3 = 1) \wedge (a_4 = 3) \wedge (a_5 = 1) \implies (d = 2)$$

W przypadku, gdy sprzeczność występuje na wszystkich $card\{A\}$ deskryptorach warunkowych, tworzymy regułę, która ma alternatywne decyzje. Takie obiekty należą do brzegu systemu decyzyjnego.

Nasze reguły LEM2 ostatecznie mają postać:

rule1 $(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1) \wedge (a_4 = 3) \wedge (a_5 = 2) \implies (d = 1)$

rule2 $(a_1 = 2) \implies (d = 1)[2]$

rule3 $(a_1 = 4) \implies (d = 1)$

rule4 $(a_1 = 3) \implies (d = 2)[2]$

rule5 $(a_1 = 1) \wedge (a_2 = 1) \wedge (a_3 = 1) \wedge (a_4 = 3) \wedge (a_5 = 1) \implies (d = 2)$