Ćwiczenie 2 - Sztuczna Inteligencja

Task 2 - Artificial Intelligence

Systemy bazujące na regułach - wyliczanie reguł decyzyjnych wybranymi technikami (Rule based systems - exemplary rule generation techniques)

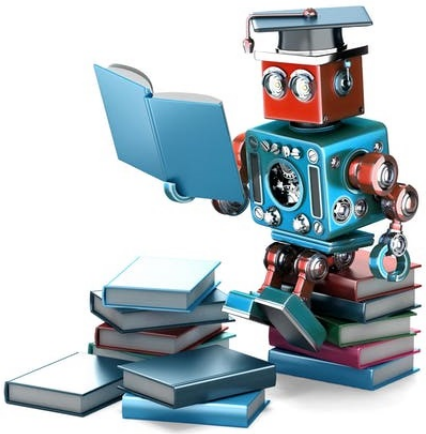

image source: https://theconversation.com/understanding-the-four-types-of-ai-from-reactive-robots-to-self-aware-beings-67616

Krótkie wprowadzenie teoretyczne (Short theoretical introduction)

Tematem ćwiczenia jest przetrenowanie wybranych technik generacji reguł decyzyjnych. Reguła decyzyjna to przepis na podejmowanie konkretnej, dyskretnej decyzji. Składa się z części warunkowej i decyzjnej. Przykładowa reguła to

$$JEZELI\ (pogoda = sloneczna)\ AND\ (czas = wolny) => (d = wizyta\ w\ parku)$$

jej część warunkowa to $(pogoda = sloneczna)\ AND\ (czas = wolny)$ reguła jest rzędu dwa (ma długość dwa) ponieważ posiada dwa deskryproty warunkowe, (dwie informacje, które są potrzebne do podjęcia decyzji $(d = wizyta\ w\ parku)$),

Deskryptory warunkwe łączymy znakiem koniunkcji AND lub alternatywy OR, w przypadku koniunkcji warunki muszą być spełnione jednocześnie, dla alternatywy jeden warunek musi być spełniony, żeby reguła zadziałała. Deskrytor decyzyjny zapisujemy po znaku implikacji $=>$.

Wyliczone reguły decyzyjne są klasyfikatorem, aby podjąć decyzję na nowym obiekcie (rozwiązać problem) należy wykryć, które z dostępnych reguł do niego pasują i dokonać głosowania pomiędzy regułami. Podejmowanie decyzji na podstawie reguł jest bardzo naturalne, zbieramy po prostu dane o problemie i rozwiązyjemy go.

Przykładem reguły sprzecznej w systemie z Tab. 2 jest:

$$JEZELI\ (czy\_obec\_spr? = NIE) => (czy\_isc? = TAK)$$

, ponieważ w systemie występuje też reguła

$$JEZELI\ (czy\_obec\_spr? = NIE) => (czy\_isc? = NIE)$$

występuje ten sam opis (warunek), a decyzja jest inna.
Prawidłową regułą jest

$$JEZELI\ (czy\_obec\_spr? = TAK) => (czy\_isc? = TAK)$$

W ramach przypomnienia poniżej przytaczamy opis teoretyczny z ćwiczenia pierwszego.

Przykładowym problemem klasyfikacji może być podejmowanie decyzji czy wybrać się na dany wykład czy nie. Zaprezentujmy poniżej system informacyjny (tablicę informacji o wykładach) - patrz Tab. 1 oraz system decyzyjny, na podstawie którego możemy automatycznie podejmować decyzję - patrz Tab. 2.

Tabela 1: System informacyjny o wykładach w pierwszym tygodniu semestru 6

	$czy\_obec\_spr?$	$czy\_interes\_tremat?$	$liczba\_godz\_w\_tyg.$
$wyklad\_przedmiot_1$	TAK	NIE	2
$wyklad\_przedmiot_2$	NIE	TAK	1
$wyklad\_przedmiot_3$	NIE	NIE	2
$wyklad\_przedmiot_4$	NIE	NIE	1
$wyklad\_przedmiot_5$	TAK	TAK	2
$wyklad\_przedmiot_6$	NIE_WIEM	NIE_WIEM	1

Tabela 2: System decyzyjny - pokazuje decyzje podejmowane przez konkrentego studenta w pierwszym tygodniu semestru 6

	$czy\_obec\_spr?$	$czy\_interes\_tremat?$	$liczba\_godz\_w\_tyg.$	czy_isc?
$wyklad\_przedmiot_1$	TAK	NIE	2	TAK
$wyklad\_przedmiot_2$	NIE	TAK	1	TAK
$wyklad\_przedmiot_3$	NIE	NIE	2	NIE
$wyklad\_przedmiot_4$	NIE	NIE	1	NIE
$wyklad\_przedmiot_5$	TAK	TAK	2	TAK
$wyklad\_przedmiot_6$	NIE_WIEM	NIE_WIEM	1	TAK

System informacyjny z Tab. 1, to zbiór opisanych obiektów.

System decyzyjny z Tab. 2, to zbiór rozwiazanych problemów, czyli lista wykładów, co do których mamy już podjętą decyzję.

Obiektami systemu informacyjnego/decyzyjnego są poszczególne wykłady $wyklad\_przedmiot\_1$ ... $wyklad\_przedmiot_6$.

Atrybutami systemów są informacje opisujące obiekty, czyli $czy\_obec\_spr?$, $czy\_interes\_temat?$, $liczba\_godz\_w\_tyg$..

Na podstawie systemu informacyjnego z Tab. 1, każdy ze studentów może podjąć indywidualne decyzję i system decyzyjny z Tab. 2 nie musi być za każdym razem identyczny.

Student rozwiązując problem obecności na wykładach, podejmuje decyzję dla każdego wykładu na podstawie jego opisu. W ten sposób tworzy bazę wiedzy (zbiór rozwiązanych problemów), która może być użyteczna do automatycznego podejmowania decyzji.

Aby odwołać się do systemu informacyjnego lub decyzyjnego używamy zapisu deskryptorowego,

np. $czy\_obec\_spr?(wyklad\_przedmiot_2) = NIE$, deskryptor mówi nam, że na wykładzie 2 obecność nie jest sprawdzana.

Jak zapiszemy ogólnie ($czy\_obec\_spr? = NIE$) odwołujemy się do wszystkich wykładów, na których obecność nie jest sprawdzana.

W naszych systemach wartość NIE_WIEM jest tak zwaną wartością nieznaną (MISSING VALUE).

W systemie decyzyjnym możemy znaleźć pewne reguły działania, np. JEŻELI ($czy\_obec\_spr? = TAK$) => ($czy\_isc? = TAK$), czyli gdy na wykładzie jest sprawdzana obecność definitywnie zaleca się przychodzenie na wykład. Gdy obecność nie jest obowiązkowa, wg. naszego systemu sprawa jest niejednoznaczna, ponieważ inne opisy (atrybuty) mogą mieć wpływ na decyzję o obecności.

Przejdźmy do zestawu zadań, które mają na celu demonstrację wybranych technik generacji reguł decyzyjnych.

Zadania do wykonania (Set of tasks to do)

1) Tworzymy w dowolnym miejscu na dysku katalog w formacie Imię.Nazwisko, w którym umieszczamy wszystkie pliki związane z ćwiczeniem (Create on the desktop folder in the format name.surname, collect all your files needed to complete the task).

3) Generujemy system decyzyjny za pomocą programu ds_generator.exe, w przypadku problemów z generatorem (wybieramy system z katalogu ds-additional z numerem miesiąca urodzenia (Generate your own decision system using ds_generator.exe, in case of problems choice the system with your moth birth number form the ds-additional folder).

4) Implementujemy w wybranym języku programowania następujące metody (Implement in chosen programming language the following methods):

• reguły sekwencyjnie pokrywające obiekty (sequential covering algorithm),

• reguły wyczerpujące metodą bazującą na macierzy nieodróżnialności (exhaustive set of rules with indiscernibility matrix usage),

• reguły LEM2 (LEM2 rules).

5) Przykładowy program demonstracyjny, ułatwiający pracę, w języku C++, znajdziemy na stronie http://wmii.uwm.edu.pl/∼artem w zakładce Dydaktyka/Sztuczna Inteligencja (Its the link to exemplary auxiliary program).

Algorytm z rodziny sekwencyjnie pokrywających (sequential covering)

Idea algorytmu pokrywającego obiekty (Algorithm idea)
Szukamy w obiektach systemu decyzyjnego, począwszy od pierwszego, a skończywszy na ostatnim reguł długości jeden, które są niesprzeczne. Po znalezieniu reguły niesprzecznej, dany obiekt wyrzucamy z rozważań, pamiętając o tym, że dalej bierze udział w sprawdzaniu sprzeczności i może wspierać inne reguły. (In the first step of algorithm, we are looking for rules of length one, which are correct. In case we get any, the objects covered by rules are removed from considerations. But we have to remember that they are usefull with conflict detections and they are included in support of rules)
Jeżeli po przeszukaniu wszystkich obiektów, pozostają obiekty nie wyrzucone z rozważań, szukamy w nich kombinacji niesprzecznej długości dwa i postępujemy analogicznie jak w przypadku reguł pierwszego rzędu. Wyszukiwanie reguł niesprzecznych jest kontynuowane do momentu wyeliminowania wszystkich obiektów niesprzecznych. Jeżeli w systemie pojawią się obiekty, które są sprzeczne na wszystkich deskryptorach, nie kreujemy z nich reguł. (In case there are objects with all conflicting single descriptors, we are trying to find the first not conflicting rule of higher order).

Przykładowe wyliczanie reguł pokrywających obiekty: (Toy example of rule generation)

Dany mamy system decyzyjny (for given decisio nsystem) (U, A, d), gdzie (where) $U = o_1, o_2, ..., o_7, o_8$, $A = a_1, a_2, ..., a_6$
d – atrybut decyzyjny (decision attribute)
Rząd I: (rules of length one)

z o_1 brak (lack of correct rules)
z o_2 $(a_6 = 2) \Rightarrow (d = 1)[3]$, wyrzucamy z rozważań obiekty (we leave from considerations the objects) o_2, o_4, o_6.
z o_3 brak
z o_5 brak
z o_7 brak
z o_8 $(a_5 = 4) \Rightarrow (d = 1)$, wyrzucamy z rozważań obiekt o_8.

Rząd II:

	a1	a2	a3	a4	a5	a6	d
o1	1	1	1	1	3	1	1
o2	1	1	1	1	3	2	1
o3	1	1	1	3	2	1	0
o4	1	1	1	3	3	2	1
o5	1	1	2	1	2	1	0
o6	1	1	2	1	2	2	1
o7	1	1	2	2	3	1	0
o8	1	1	2	2	4	1	1

z o_1 $(a_3 = 1) \wedge (a_4 = 1) \Rightarrow (d = 1)[2]$, wyrzucamy z rozważań obiekt o_1.
z o_3 $(a_3 = 1) \wedge (a_5 = 2) \Rightarrow (d = 0)$, wyrzucamy z rozważań obiekt o_3.
z o_5 $(a_5 = 2) \wedge (a_6 = 1) \Rightarrow (d = 0)[2]$, wyrzucamy z rozważań obiekt o_5.
z o_7 $(a_3 = 2) \wedge (a_5 = 3) \Rightarrow (d = 0)$, wyrzucamy z rozważań obiekt o_7.

Reguły wyczerpujące - z użyciem macierzy nieodróżnialności (exhaustive set of rules):

Dany mamy system decyzyjny (U, A, d), gdzie $U = o1, o2, .., o8$, $A = a_1, a_2, .., a_6$
d – atrybut decyzyjny

	a1	a2	a3	a4	a5	a6	d
o1	1	1	1	1	3	1	1
o2	1	1	1	1	3	2	1
o3	1	1	1	3	2	1	0
o4	1	1	1	3	3	2	1
o5	1	1	2	1	2	1	0
o6	1	1	2	1	2	2	1
o7	1	1	2	2	3	1	0
o8	1	1	2	2	4	1	1

Tworzymy dla niego macierz nieodróżnialności (we compute indiscernibility matrix) $\mu_A = [c_{ij}]_{8\times 8}$, gdzie $i, j = 1, 2, ..., 8$. Dla obiektów $x_1, \ldots, x_8$ zostawiamy w kolumnach tylko kratki o współrzędnych, które odpowiadają obiektom o innych decyzjach, gdzie

$$c_{ij} = \{a \in A : \ a(x_i) = a(x_j)\},$$

Komputerowi wystarczy tablica postaci (There is only need to use triangular part of matrix):

	o1	o2	o3	o4	o5	o6	o7	o8
o1			a1 a2 a3 a6		a1 a2 a4 a6		a1 a2 a5 a6	
o2			a1 a2 a3		a1 a2 a4		a1 a2 a5	
o3	a1 a2 a3 a6	a1 a2 a3		a1 a2 a3 a4		a1 a2 a5		a1 a2 a6
o4			a1 a2 a3 a4		a1 a2		a1 a2 a5	
o5	a1 a2 a4 a6	a1 a2 a4		a1 a2		a1 a2 a3 a4 a5		a1 a2 a3 a6
o6			a1 a2 a5		a1 a2 a3 a4 a5		a1 a2 a3	
o7	a1 a2 a5 a6	a1 a2 a5		a1 a2 a5		a1 a2 a3		a1 a2 a3 a4 a6
o8			a1 a2 a6		a1 a2 a3 a6		a1 a2 a3 a4 a6	

Tabela1: Macierz nieodróżnialności dla (indiscernibility matrix for) (U, A, d)

	o1	o2	o3	o4	o5	o6	o7
o1							
o2							
o3	a1 a2 a3 a6	a1 a2 a3					
o4			a1 a2 a3 a4				
o5	a1 a2 a4 a6	a1 a2 a4		a1 a2			
o6			a1 a2 a5		a1 a2 a3 a4 a5		
o7	a1 a2 a5 a6	a1 a2 a5		a1 a2 a5		a1 a2 a3	
o8			a1 a2 a6		a1 a2 a3 a6		a1 a2 a3 a4 a6

Tabela2: Macierz nieodróżnialności trójkątna dla (traingular part of insiscernibility matrix) (U, A, d)

Wyliczania reguły za pomocą Tabeli 1. Rozważana Tabela, pozwala, na szybkie wyszukiwanie niesprzecznych reguł. Szukanie reguł w obiekcie i-tym polega na szukaniu w kolumnie i-tej kombinacji deskryptorów, która w niej nie występuje. (It is easy to find rules using indiscernibility matrix, non conflicting rules for object i are the combinations of descriptors, which are missing in the $i-th$ column of matrix)

Zacznijmy od reguł I rzędu) (Lets start from rules of length one):
dla $o2$ mamy
$\underline{(a_6 = 2) => (d = 1)}$
dla $o4$ mamy
$\underline{(a_6 = 2) => (d = 1)}$
dla $o6$ mamy
$\underline{(a_6 = 2) => (d = 1)}$
dla o8 mamy
$(a_5 = 4) => (d = 1)$
pozostałe kolumny nie generują żadnej reguły. (there is no any rule with the rest columns)
Trzy podkreślone reguły zapisujemy jako regułę z supportem 3 (support oznacza liczbę obiektów systemu decyzyjnego do których pasuje reguła), czyli grupa reguł rzędu I jest postaci (Three underlined rules are represent as one rule with support 3):
$(a_6 = 2) => (d = 1)[3]$
$(a_5 = 4) => (d = 1)$
W kolejnym etapie modyfikujemy Tabelę 1 , zaznaczając deskryptory, z których nie będziemy mogli zbudować reguł wyższych rzędów. (czyli deskryptory: $(a_6 = 2), (a_5 = 4)$) Otrzymujemy (In the next step we mark used descriptors, to avoid its usage once again):
Teraz wybieramy reguły II rzędu, (szukamy kombinacji bez powtórzeń długości 2, które nie występują w danej kolumnie, pamiętając o pominięciu kombinacji zawierających reguły niższego rzędu) (Its time to find the rules of length 2)
z $o1$ mamy
$(a_3 = 1)$ & $(a_4 = 1) => (d = 1)$
$(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$
z $o2$ mamy
$(a_3 = 1)$ & $(a_4 = 1) => (d = 1)$
$(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$

	o1	o2	o3	o4	o5	o6	o7	o8
o1			a1 a2 a3 a6		a1 a2 a4 a6		a1 a2 a5 a6	
o2			a1 a2 a3		a1 a2 a4		a1 a2 a5	
o3	a1 a2 a3 a6	a1 a2 a3		a1 a2 a3 a4		a1 a2 a5		a1 a2 a6
o4			a1 a2 a3 a4		a1 a2		a1 a2 a5	
o5	a1 a2 a4 a6	a1 a2 a4		a1 a2		a1 a2 a3 a4 a5		a1 a2 a3 a6
o6			a1 a2 a5		a1 a2 a3 a4 a5		a1 a2 a3	
o7	a1 a2 a5 a6	a1 a2 a5		a1 a2 a5		a1 a2 a3		a1 a2 a3 a4 a6
o8			a1 a2 a6		a1 a2 a3 a6		a1 a2 a3 a4 a6	
		pomiń a6		pomiń a6		pomiń a6		pomiń a5

Tabela3

$(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$
z $o3$ mamy
$(a_3 = 1)$ & $(a_5 = 2) => (d = 0)$
$(a_4 = 3)$ & $(a_5 = 2) => (d = 0)$
$(a_4 = 3)$ & $(a_6 = 1) => (d = 0)$
$(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
z $o4$ mamy
$(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_4 = 3)$ & $(a_5 = 3) => (d = 1)$
z $o5$ mamy
$(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
z $o7$ mamy
$(a_3 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_4 = 2)$ & $(a_5 = 3) => (d = 0)$
Teraz reguły identyczne zapisujemy jako pojedyncze z odpowiednim supportem, otrzymujemy reguły postaci: (Support is combined)
$(a_3 = 1)$ & $(a_4 = 1) => (d = 1)[2]$
$(a_3 = 1)$ & $(a_5 = 3) => (d = 1)[3]$
$(a_4 = 1)$ & $(a_5 = 3) => (d = 1)[2]$
$(a_3 = 1)$ & $(a_5 = 2) => (d = 0)$
$(a_4 = 3)$ & $(a_5 = 2) => (d = 0)$
$(a_4 = 3)$ & $(a_6 = 1) => (d = 0)$
$(a_5 = 2)$ & $(a_6 = 1) => (d = 0)[2]$
$(a_4 = 3)$ & $(a_5 = 3) => (d = 1)$
$(a_3 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_4 = 2)$ & $(a_5 = 3) => (d = 0)$

Następnie korzystając z Tab. 3, wypisujemy kombinacje długości 3, które będą kandydatami na reguły: (consideeing Tab. 3, we are listed the condidates for rules of length 3)
z $o1$ mamy
$(a_1 = 1)$ & $(a_3 = 1)$ & $(a_4 = 1) => (d = 1)$
$(a_1 = 1)$ & $(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_1 = 1)$ & $(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_2 = 1)$ & $(a_3 = 1)$ & $(a_4 = 1) => (d = 1)$
$(a_2 = 1)$ & $(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_2 = 1)$ & $(4 = 1)$ & $(a_5 = 1) => (d = 1)$
$(a_3 = 1)$ & $(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$

$(a_3 = 1)$ & $(a_4 = 1)$ & $(a_6 = 1) => (d = 1)$
$(a_3 = 1)$ & $(a_5 = 3)$ & $(a_6 = 1) => (d = 1)$
$(a_4 = 1)$ & $(a_5 = 3)$ & $(a_6 = 1) => (d = 1)$
z $o2$ mamy
$(a_1 = 1)$ & $(a_3 = 1)$ & $(a_4 = 1) => (d = 1)$
$(a_1 = 1)$ & $(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_1 = 1)$ & $(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_2 = 1)$ & $(a_3 = 1)$ & $(a_4 = 1) => (d = 1)$
$(a_2 = 1)$ & $(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_2 = 1)$ & $(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_3 = 1)$ & $(a_4 = 1)$ & $(a_5 = 3) => (d = 1)$
z o3 mamy
$(a_1 = 1)$ & $(a_3 = 1)$ & $(a_5 = 2) => (d = 0)$
$(a_1 = 1)$ & $(a_4 = 3)$ & $(a_5 = 2) => (d = 0)$
$(a_1 = 1)$ & $(a_4 = 3)$ & $(a_6 = 1) => (d = 0)$
$(a_1 = 1)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
$(a_2 = 1)$ & $(a_3 = 1)$ & $(a_5 = 2) => (d = 0)$
$(a_2 = 1)$ & $(a_4 = 3)$ & $(a_5 = 2) => (d = 0)$
$(a_2 = 1)$ & $(a_4 = 3)$ & $(a_6 = 1) => (d = 0)$
$(a_2 = 1)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
$(a_3 = 1)$ & $(a_4 = 3)$ & $(a_5 = 2) => (d = 0)$
$(a_3 = 1)$ & $(a_4 = 3)$ & $(a_6 = 1) => (d = 0)$
$(a_4 = 3)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
z $o4$ mamy
$(a_1 = 1)$ & $(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_1 = 1)$ & $(a_4 = 3)$ & $(a_5 = 3) => (d = 1)$
$(a_2 = 1)$ & $(a_3 = 1)$ & $(a_5 = 3) => (d = 1)$
$(a_2 = 1)$ & $(a_4 = 3)$ & $(a_5 = 3) => (d = 1)$
$(a_3 = 1)$ & $(a_4 = 3)$ & $(a_5 = 3) => (d = 1)$
z $o5$ mamy
$(a_1 = 1)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
$(a_2 = 1)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
$(a_3 = 2)$ & $(a_4 = 1)$ & $(a_6 = 1) => (d = 0)$
$(a_3 = 2)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
$(a_4 = 1)$ & $(a_5 = 2)$ & $(a_6 = 1) => (d = 0)$
z o7 mamy
$(a_1 = 1)$ & $(a_3 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_1 = 1)$ & $(a_4 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_2 = 1)$ & $(a_3 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_2 = 1)$ & $(a_4 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_3 = 2)$ & $(a_4 = 2)$ & $(a_5 = 3) => (d = 0)$
$(a_3 = 2)$ & $(a_5 = 3)$ & $(a_6 = 1) => (d = 0)$
$(a_4 = 2)$ & $(a_5 = 3)$ & $(a_6 = 1) => (d = 0)$

rozważane kombinacje na pewno nie są sprzeczne, stąd teraz eliminujemy, tych kandydatów, którzy zawierają reguły niższego rzędu: zaznaczmy podkreśleniem, zawieranie reguł niższego rzędu: (and now we eliminating the rules, which containing existing shortest rules)

z $o1$ mamy

$(a_1 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ => $(d = 1)$

$(a_1 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_1 = 1)$ & $\underline{(a_4 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_4 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ & $(a_5 = 3)$ => $(d = 1)$

$\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ & $(a_6 = 1)$ => $(d = 1)$

$\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ & $(a_6 = 1)$ => $(d = 1)$

$\underline{(a_4 = 1)}$ & $\underline{(a_5 = 3)}$ & $(a_6 = 1)$ => $(d = 1)$

z $o2$ mamy

$(a_1 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ => $(d = 1)$

$(a_1 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_1 = 1)$ & $\underline{(a_4 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_4 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$\underline{(a_3 = 1)}$ & $\underline{(a_4 = 1)}$ & $(a_5 = 3)$ => $(d = 1)$

z o3 mamy

$(a_1 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 2)}$ => $(d = 0)$

$(a_1 = 1)$ & $\underline{(a_4 = 3)}$ & $\underline{(a_5 = 2)}$ => $(d = 0)$

$(a_1 = 1)$ & $\underline{(a_4 = 3)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$(a_1 = 1)$ & $\underline{(a_5 = 2)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$(a_2 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 2)}$ => $(d = 0)$

$(a_2 = 1)$ & $\underline{(a_4 = 3)}$ & $\underline{(a_5 = 2)}$ => $(d = 0)$

$(a_2 = 1)$ & $\underline{(a_4 = 3)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$\underline{(a_2 = 1)}$ & $\underline{(a_5 = 2)}$ & $(a_6 = 1)$ => $(d = 0)$

$\underline{(a_3 = 1)}$ & $(a_4 = 3)$ & $\underline{(a_5 = 2)}$ => $(d = 0)$

$\underline{(a_3 = 1)}$ & $\underline{(a_4 = 3)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$\underline{(a_4 = 3)}$ & $(a_5 = 2)$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

z $o4$ mamy

$(a_1 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_1 = 1)$ & $\underline{(a_4 = 3)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_3 = 1)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$(a_2 = 1)$ & $\underline{(a_4 = 3)}$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

$\underline{(a_3 = 1)}$ & $(a_4 = 3)$ & $\underline{(a_5 = 3)}$ => $(d = 1)$

z $o5$ mamy

$(a_1 = 1)$ & $\underline{(a_5 = 2)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$(a_2 = 1)$ & $\underline{(a_5 = 2)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$(a_3 = 2)$ & $\underline{(a_4 = 1)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$ ta jest prawidłowa

$(a_3 = 2)$ & $\underline{(a_5 = 2)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

$(a_4 = 1)$ & $\underline{(a_5 = 2)}$ & $\underline{(a_6 = 1)}$ => $(d = 0)$

z o7 mamy

$(a_1 = 1)$ & $\underline{(a_3 = 2)}$ & $\underline{(a_5 = 3)}$ => $(d = 0)$

$(a_1 = 1)$ & $\underline{(a_4 = 2)}$ & $\underline{(a_5 = 3)}$ => $(d = 0)$

$(a_2 = 1)$ & $\underline{(a_3 = 2)}$ & $\underline{(a_5 = 3)}$ => $(d = 0)$

$(a_2 = 1)$ & $\underline{(a_4 = 2)}$ & $\underline{(a_5 = 3)}$ => $(d = 0)$
$\underline{(a_3 = 2)}$ & $\underline{(a_4 = 2)}$ & $\underline{(a_5 = 3)}$ => $(d = 0)$
$\underline{(a_3 = 2)}$ & $(a_5 = 3)$ & $\underline{(a_6 = 1)}$ => $(d = 0)$
$\underline{(a_4 = 2)}$ & $\underline{(a_5 = 3)}$ & $(a_6 = 1)$ => $(d = 0)$

Mamy tylko jedną regułę trzeciego rzędu postaci, (There is only one rule of length 3)
$(a_3 = 2)$ & $(a_4 = 1)$ & $(a_6 = 1)$ => $(d = 0)$

Kolejne rzędy reguł wyliczamy analogicznie. W naszym przypadku nie ma potrzeby sprawdzać, czy istnieją reguły rzędu cztery, otrzymanie tylko jednej reguły rzędu III, może być tu warunkiem stopu. (Algorithm work analogously for higher lengths of rules) In our case is finished.

Ostatecznie nasz zbiór reguł exhaustive wyliczony zmodyfikowaną macierzą nieodróżnialności jest postaci (the final set of rules is as follows):
I
$(a_6 = 2)$ => $(d = 1)[3]$
$(a_5 = 4)$ => $(d = 1)$
II
$(a_3 = 1)$ & $(a_4 = 1)$ => $(d = 1)[2]$
$(a_3 = 1)$ & $(a_5 = 3)$ => $(d = 1)[3]$
$(a_4 = 1)$ & $(a_5 = 3)$ => $(d = 1)[2]$
$(a_3 = 1)$ & $(a_5 = 2)$ => $(d = 0)$
$(a_4 = 3)$ & $(a_5 = 2)$ => $(d = 0)$
$(a_4 = 3)$ & $(a_6 = 1)$ => $(d = 0)$
$(a_5 = 2)$ & $(a_6 = 1)$ => $(d = 0)[2]$
$(a_4 = 3)$ & $(a_5 = 3)$ => $(d = 1)$
$(a_3 = 2)$ & $(a_5 = 3)$ => $(d = 0)$
$(a_4 = 2)$ & $(a_5 = 3)$ => $(d = 0)$
III
$(a_3 = 2)$ & $(a_4 = 1)$ & $(a_6 = 1)$ => $(d = 0)$

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

Dany mamy system decyzyjny (U, A, d), gdzie $U = o1, o2, .., o7$, $A = a_1, a_2, .., a_5$
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. (The rules are created with usage of the most common attribute values in the system), starting form the most common value, there is high probability to have conflict in system, that is why we are looking for non conflicting conjunction, which are formed from the subsequently added most common values.)

Wszelkie konflikty rozwiązywane są hierarchicznie (wybierana jest wartość pierwsza od góry z lewej strony) (ties are resolved in hierarchic way, first conflicting value from tghe top and left side is chosen)

W praktyce wygląda to tak (in practice):
Patrzymy na koncept 1 (koncept jest synonimem klasy decyzyjnej), szukając deskryptora, który występuje najczęściej:
W wybranym koncepcie najczęściej występuje deskryptor (considering first decision class, the most common descriptor is)

$(a_2 = 1) \rightharpoonup$ powstaje z obiektów (it fits to objects) o_2, o_3, o_4

Nie tworzy jednak reguły ponieważ w koncepcie 2 mamy sprzeczność (is conflicting).
Skupiając uwagę na obiektach do których pasuje (considering in next step only object, which fits to descriptor) $(a_2 = 1)$, czyli (thus) o_2, o_3, o_4, szukam kolejnego najlepszego deskryptora, z największym pokryciem w klasie 1 (we are looking for next most common descriptor in class 1). Tym deskryptorem jest (this descriptor is) $(a_3 = 1) \rightharpoonup$ powstaje z obiektów (is created from) o_2, o_3, dokładam go do pierwszego deskryptora i tworzę koniunkcję (we are adding it to the firs descriptor as conjunction):

$(a_2 = 1) \wedge (a_3 = 1)$, jednak powstała koniunkcja dalej jest sprzeczna (this conjunction is still conflicting),

Z faktu, że powyższa reguła powstała z obiektów o_2, o_3, szukam w nich kolejnego najbardziej licznego deskryptora, tym razem jest nim $(a_1 = 1) \rightharpoonup$ powstaje z obiektu o_2, dokładam znaleziony deskryptor do budowanej reguły (we are looking for next most common descriptor considering respective objects), we are doing similar steps until the conflict is resolved:

$(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1)$, 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 (its final non conflicting rule), object o_2 is removed from considerations (is covered):

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

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 (We are looking for next rule from not covered object in class 1):

$(a_1 = 2) \Rightarrow (d = 1)[2]$, reguła ma support 2, ponieważ pasuje do dwóch obiektów, o_1 i o_3 (its correct rule, with support 2).

w koncepcie 1, został nam do rozważenia tylko obiekt o_4, z którego powstaje niesprzeczna reguła (and the last rule is as follows):

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

Następnie tworzymy reguły z konceptu 2 (we are computing the rules from class 2 in analogous way):

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:

$(a_1 = 3) \Rightarrow (d = 2)[2]$, pokrywa obiekty o_5, o_6.

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

$(a_1 = 1)$, $(a_2 = 1)$, $(a_3 = 1)$, $(a_4 = 3)$ tworzą sprzeczność, dopiero dołożenie deskryptora $(a_5 = 1)$, likwiduje sprzeczność i powstaje reguła:

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

W przypadku gry 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 (Final set of rules is as follows):

rule1 $(a_2 = 1) \wedge (a_3 = 1) \wedge (a_1 = 1) \wedge (a_4 = 3) \wedge (a_5 = 2) \Rightarrow (d = 1)$
rule2 $(a_1 = 2) \Rightarrow (d = 1)[2]$
rule3 $(a_1 = 4) \Rightarrow (d = 1)$
rule4 $(a_1 = 3) \Rightarrow (d = 2)[2]$
rule5 $(a_1 = 1) \wedge (a_2 = 1) \wedge (a_3 = 1) \wedge (a_4 = 3) \wedge (a_5 = 1) \Rightarrow (d = 2)$