Algorytm LEM1

Oznaczenia i definicje:

- U - uniwersum, tj. zbiór obiektów;
- A - zbiór atrybutów warunkowych;
- d - atrybut decyzyjny;
- $IND(B) = \{(x, y) \in U \times U : \forall_{a \in B} a(x) = a(y)\}$ - relacja nierozróżnialności, tj. zbiór wszystkich par obiektów, które mają takie same wartości na wszystkich atrybutach ze zbioru $B \subseteq A$;
- B^* - rodzina wszystkich klas nierozróżnialności wyznaczonych przez relację $IND(B)$, tzn. $B^* = U/IND(B)$;
- $\{d\}^*$ - rodzina wszystkich klas nierozróżnialności wyznaczonych przez relację $IND(\{d\})$, tzn. $\{d\}^* = U/IND(\{d\})$. Inaczej, rodzina wszystkich klas decyzyjnych;
- Zbiór $\{d\}$ *zależy* od zbioru B wtedy i tylko wtedy, gdy $B^* \leqslant \{d\}^*$ (tutaj " $\leqslant$ " oznacza, że każdy zbiór pierwszej rodziny jest podzbiorem pewnego zbioru drugiej rodziny, czyli $\forall_{X \in B^*} \exists_{Y \in \{d\}^*} X \subseteq Y$);
- B jest *globalnym pokryciem* zbioru $\{d\}$ wtedy i tylko wtedy, gdy $\{d\}$ zależy od B i nie istnieje $B' \subset T$ takie, że $\{d\}$ zależy od B';

Algorithm 1: LEM1

Data: zbiór A wszystkich atrybutów warunkowych, rodzina $\{d\}^*$;
Result: pojedyncze globalne pokrycie B;
begin
  $B := \emptyset$; $C := A$;
  if $A^* \leqslant \{d\}^*$ **then**
    foreach $a \in A$ **do**
      $D := C \setminus \{a\}$;
      if $D^* \leqslant \{d\}^*$ **then** $C := D$;
    $B := C$;
end

Opis algorytmu.
Algorytm LEM1 działa jedynie dla danych niesprzecznych, tzn. każde dwa

obiekty z U, należące do różnych klas muszą być rozróżnialne na atrybutach z A. Wtedy spełniony jest warunek $A^* \leqslant \{d\}^*$. Algorytm w każdym kroku sprawdza, czy możliwe jest usunięcie jednego atrybutu z rozpatrywanego zbioru atrybutów (tj. $D := C \setminus \{a\}$). Jeżeli tak (tj. $D^* \leqslant \{d\}^*$), to atrybut jest usuwany ze zbioru atrybutów. W efekcie otrzymywany jest najmniejszy podzbiór atrybutów warunkowych zachowujący rozróżnialność obiektów z różnych klas.

Przykład 1.
Dana jest tablica decyzyjna $(U, A \cup \{d\})$

obiekt	*temperatura*	*bol_glowy*	*oslabienie*	*nudnosci*	*grypa*
1	b._wysoka	tak	tak	nie	tak
2	wysoka	tak	nie	tak	tak
3	normalna	nie	nie	nie	nie
4	normalna	tak	tak	tak	tak
5	wysoka	nie	tak	nie	tak
6	wysoka	nie	nie	nie	nie
7	normalna	nie	tak	nie	nie

gdzie $U = \{1, \ldots, 7\}$,
$A = \{temperatura, bol\_glowy, oslabienie, nudnosci\}, d = grypa.$

Ilustracja działania algorytmu:
$B := \emptyset$; $C := A$;
$A^* = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}\}, d = \{\{1, 2, 4, 5\}, \{3, 6, 7\}\}$,
zatem $A^* \leqslant \{d\}^*$;
$D := C \setminus \{temperatura\} = \{bol\_glowy, oslabienie, nudnosci\}$,
stąd $D^* = \{\{1\}, \{2\}, \{3, 6\}, \{4\}, \{5, 7\}\}$, więc $D^* \not\leqslant \{d\}^*$;
$D := C \setminus \{bol\_glowy\} = \{temperatura, oslabienie, nudnosci\}$,
stąd $D^* = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}\}$, więc $D^* \leqslant \{d\}^*$,
zatem $C := D$;
$D := C \setminus \{oslabienie\} = \{temperatura, nudnosci\}$,
stąd $D^* = \{\{1\}, \{2\}, \{3, 7\}, \{4\}, \{5, 6\}\}$, więc $D^* \not\leqslant \{d\}^*$;
$D := C \setminus \{nudnosci\} = \{temperatura, oslabienie\}$,
stąd $D^* = \{\{1\}, \{2, 6\}, \{3\}, \{4, 7\}, \{5\}\}$, więc $D^* \not\leqslant \{d\}^*$,
zatem globalne pokrycie wynosi $B := \{temperatura, oslabienie, nudnosci\}$;

Na podstawie globalnego pokrycia konstruowane są reguły przy zastosowaniu techniki „dropping conditions". Na początku konstruowana jest reguła w oparciu o obiekt 1, tj.
$(temperatura, b.\_wysoka) \wedge (oslabienie, tak) \wedge (nudnosci, nie) \rightarrow (grypa, tak)$.
Reguła pokrywa tylko pierwszy obiekt (tj. 1 jako jedyny spełnia część warun-

kową reguły). Aby zwiększyć pokrycie reguły o kolejne obiekty z tej samej klasy decyzyjnej, rozpatrywane jest opuszczenie kolejnych warunków reguły. Warunek $(temperatura, b.\_wysoka)$ nie może być opuszczony, gdyż otrzymana w ten sposób reguła $(oslabienie, tak) \wedge (nudnosci, nie) \rightarrow (grypa, tak)$, oprócz obiektu 1, pokrywa obiekt z innej klasy decyzyjnej, tj. obiekt 7. Możliwy do usunięcia jest warunek $(oslabienie, tak)$, a także $(nudnosci, nie)$, chociaż nie zmienia to pokrycia reguły. Ostatecznie otrzymujemy regułę $(temperatura, b.\_wysoka) \rightarrow (grypa, tak)$ pokrywającą obiekt 1.
Dla pozostałych obiektów reguły generowane są w analogiczny sposób. Zbiór reguł wygenerowanych przez algorytm LEM1 składa się z następujących reguł (po dwukropku podane pokrycie reguły):
$(temperatura, b.\_wysoka) \rightarrow (grypa, tak) : 1,$
$(nudnosci, tak) \rightarrow (grypa, tak) : 2, 4,$
$(temperatura, wysoka) \wedge (oslabienie, tak) \rightarrow (grypa, tak) : 5,$
$(oslabienie, nie) \wedge (nudnosci, nie) \rightarrow (grypa, nie) : 3, 6,$
$(temperatura, normalna) \wedge (nudnosci, nie) \rightarrow (grypa, nie) : 3, 7.$

Algorytm LEM2

Oznaczenia i definicje:

- X – niepusta dolna lub górna aproksymacja klasy decyzyjnej;
- $t = (a, v)$ – warunek, gdzie a – atrybut warunkowy, v – wartość przyjmowana przez a;
- $[t]$ – blok, tj. zbiór obiektów spełniających warunek $t = (a, v)$;
- T - zbiór warunków t;
- $[T] = \bigcap_{t \in T}[t]$ - zbiór obiektów spełniających każdy z warunków $t \in T$;
- Zbiór X *zależy* od zbioru T wtedy i tylko wtedy, gdy $\emptyset \neq [T] \subseteq X$;
- T jest *minimalnym kompleksem* zbioru X wtedy i tylko wtedy, gdy X zależy od T i nie istnieje $T' \subset T$ takie, że X zależy od T';
- $\mathcal{T}$ - rodzina zbiorów T;
- $\mathcal{T}$ jest *lokalnym pokryciem* zbioru X wtedy i tylko wtedy, gdy:
  1. Każdy zbiór $T \in \mathcal{T}$ jest minimalnym kompleksem zbioru X;
  2. $\bigcup_{T \in \mathcal{T}}[T] = X$;

3. $\mathcal{T}$ jest minimalny, tj. składa się z najmniejszej możliwej liczby zbiorów T.

Algorithm 2: LEM2

Data: zbiór X;
Result: pojedyncze lokalne pokrycie $\mathcal{T}$ zbioru X;
begin
  $G := X; \mathcal{T} := \emptyset$;
  while $G \neq \emptyset$ **do**
    $T := \emptyset$; $T_G := \{t : [t] \cap G \neq \emptyset\}$;
    while $T = \emptyset$ **or** $[T] \not\subseteq X$ **do**
      wybierz $t \in T_G$ takie, że $|[t] \cap G|$ jest maksymalne; jeżeli jest więcej niż jedno t spełniające warunek, to wybierz spośród nich pierwsze znalezione t takie, że $|[t]|$ jest minimalne;
      $T := T \cup \{t\}$; $G := [t] \cap G$; $T_G = \{t : [t] \cap G \neq \emptyset\} \setminus T$;
    foreach $t \in T$ **do**
      if $[T \setminus \{t\}] \subseteq X$ **then** $T := T \setminus \{t\}$;
    $\mathcal{T} := \mathcal{T} \cup \{T\}$; $G := X \setminus \bigcup_{T \in \mathcal{T}} [T]$;
  foreach $T \in \mathcal{T}$ **do**
    if $\bigcup_{T' \in \mathcal{T} \setminus \{T\}} [T'] = X$ **then** $\mathcal{T} := \mathcal{T} \setminus \{T\}$;
end

Opis algorytmu.

Algorytm LEM2 na wejściu otrzymuje aproksymację rozpatrywanej klasy decyzyjnej. Jeżeli jest to dolna aproksymacja, to są generowane reguły pewne, jeżeli górna, to możliwe (Zamiast aproksymacji można rozpatrywać obszar brzegowy, wtedy generowane są reguły przybliżone). Algorytm przy generowaniu każdej reguły bierze pod uwagę tylko te warunki, które są spełnione przynajmniej przez jeden obiekt z rozpatrywanego zbioru (tj. $T_G := \{t : [t] \cap G \neq \emptyset\}$). W każdym kroku generowania reguły (zbiór T reprezentuje regułę) wybierany jest taki warunek, który jest spełniany przez największą liczbę obiektów (tj. $|[t] \cap G|$ jest maksymalne). Jeżeli jest więcej takich warunków spełniających podane kryterium, to spośród nich wybierany jest ten, który jest spełniony przez najmniejszą liczbę wszystkich obiektów z U (tj. $|[t]|$ jest minimalne). W tym przypadku jest to równoważne temu, że taki warunek jest spełniony przez najmniejszą liczbę obiektów spoza zbioru G. Reguła jest tworzona (tj. $T := T \cup \{t\}$), dopóki nie jest dodany ani jeden warunek do reguły (tj. $T = \emptyset$) lub te warunki które już są dodane do reguły są spełniane nie tylko przez obiekty z rozpatrywanej aproksymacji (tj. $[T] \not\subseteq X$).

Po wygenerowaniu każdej reguły, następuje jej przycinanie poprzez usunięcie zbędnych warunków (tj. **if** $[T\backslash\{t\}] \subseteq X$ **then** $T := T\backslash\{t\}$). Przy generowaniu kolejnej reguły, rozpatrywane są tylko te obiekty z danej aproksymacji, które nie spełniają wygenerowanych do tej pory reguł (tj. $G := X \setminus \bigcup_{T \in \mathcal{T}}[T]$), czyli nie spełniają reguł znajdujących się w zbiorze $\mathcal{T}$. Po wygenerowaniu wszystkich reguł, tzn. w sytuacji gdy nie ma już obiektów z rozpatrywanej aproksymacji niespełniających którejkolwiek z reguł (tj. $G \neq \emptyset$ nie jest spełnione), następuje przycinanie zbioru reguł poprzez usuwanie zbędnych reguł (tj. **if** $\bigcup_{T' \in \mathcal{T}\setminus\{T\}}[T'] = X$ **then** $\mathcal{T} := \mathcal{T} \setminus \{T\}$).

Przykład 2.
Dana jest tablica decyzyjna z Przykładu 1.
Załóżmy, że rozpatrywana jest dolna aproksymacja klasy decyzyjnej $\{1,2,4,5\}$, zatem $X = \{1,2,4,5\}$ (w tym przypadku obie aproksymacje pokrywają się).
$G := X; \mathcal{T} := \emptyset;$
pętla **while** $G \neq \emptyset$ (krok 1):
$T := \emptyset; T_G := \{(temperatura, b.\_wysoka), (temperatura, wysoka),$
$(temperatura, normalna), (bol\_glowy, tak), (bol\_glowy, nie), (oslabienie, tak),$
$(oslabienie, nie), (nudnosci, tak), (nudnosci, nie)\};$
pętla **while** $T = \emptyset$ **or** $[T] \nsubseteq X$ (krok 1):
Dla warunków (bol_glowy, tak) i $(oslabienie, tak)$ wyrażenie $|[t] \cap G|$ przyjmuje maksymalną wartość, tj. 3. Wybierany jest warunek (bol_glowy, tak), gdyż $|(bol\_glowy, tak)| = 3 < |(oslabienie, tak)| = 5$.
$T := \{(bol\_glowy, tak)\}; G := \{1,2,4\}; T_G = \{(temperatura, b.\_wysoka),$
$(temperatura, wysoka), (temperatura, normalna, (bol\_glowy, nie),$
$(oslabienie, tak), (oslabienie, nie), (nudnosci, tak), (nudnosci, nie)\}.$
Otrzymujemy $T :\neq \emptyset$ **and** $[T] \subseteq X$, więc wychodzimy z pętli wewnętrznej.
$\mathcal{T} := \{\{(bol\_glowy, tak)\}\}; G := \{1,2,4,5\} \setminus \{1,2,4\} = \{5\}.$
pętla **while** $G \neq \emptyset$ (krok 2):
$T := \emptyset; T_G := \{(temperatura, wysoka), (bol\_glowy, nie), (oslabienie, tak),$
$(nudnosci, nie)\};$
pętla **while** $T = \emptyset$ **or** $[T] \nsubseteq X$ (krok 1):
Wybierany jest warunek $(temperatura, wysoka)$, otrzymujemy
$T := \{(temperatura, wysoka)\}; G := \{5\}; T_G = \{(bol\_glowy, nie),$
$(oslabienie, tak), (nudnosci, nie)\}.$
Ponieważ $[T] = \{2,5,6\} \nsubseteq X$, więc
pętla **while** $T = \emptyset$ **or** $[T] \nsubseteq X$ (krok 2):
Wybierany jest warunek (bol_glowy, nie), otrzymujemy
$T := \{(temperatura, wysoka), (bol\_glowy, nie)\}; G := \{5\};$
$T_G = \{(oslabienie, tak), (nudnosci, nie)\}$
Ponieważ $[T] = \{5,6\} \nsubseteq X$, więc

pętla **while** $T = \emptyset$ **or** $[T] \not\subseteq X$ (krok 3):
Wybierany jest warunek $(oslabienie, tak)$, otrzymujemy
$T := \{(temperatura, wysoka), (bol\_glowy, nie), (oslabienie, tak)\}; G := \{5\}$;
$T_G = \{(nudnosci, nie)\}$
Ponieważ $[T] = \{5\} \subseteq X$, więc wychodzimy z pętli wewnętrznej.
Przycinając regułę reprezentowaną przez zbiór T (pętla **foreach** $t \in T$) otrzymujemy
$T = \{(temperatura, wysoka), (oslabienie, tak)\}$; $\mathcal{T} := \{\{(bol\_glowy, tak)\}, \{(temperatura, wysoka), (oslabienie, tak)\}\}; G := \{5\} \setminus \{5\} = \emptyset$, więc wychodzimy z pętli zewnętrznej.

Na podstawie otrzymanego lokalnego pokrycia konstruowane są reguły: $(bol\_glowy, tak) \rightarrow (grypa, tak), (temperatura, wysoka) \wedge (oslabienie, tak) \rightarrow (grypa, tak)$. Zbiór reguł wygenerowanych przez algorytm LEM2 składa się z następujących reguł (po dwukropku podane pokrycie reguły):
$(bol\_glowy, tak) \rightarrow (grypa, tak) : 1, 2, 4,$
$(temperatura, wysoka) \wedge (oslabienie, tak) \rightarrow (grypa, tak) : 5,$
$(temperatura, normalna) \wedge (bol\_glowy, nie) \rightarrow (grypa, nie) : 3, 7,$
$(bol\_glowy, nie) \wedge (oslabienie, nie) \rightarrow (grypa, nie) : 3, 6.$