STUDIA INFORMATICA 2001
Volume 22 Number 4 (46)

Marek SIKORA, Paweł PROKSA
Politechnika Śląska, Instytut Informatyki

WYZNACZANIE PROGÓW TOLERANCJI I APROKSYMACYJNYCH REGUŁ DECYZYJNYCH

Streszczenie. W artykule zaprezentowano metodę poszukiwania progów tolerancji za pomocą algorytmu genetycznego i różnych postaci funkcji przystosowania. Prace inspirowane były metodą poszukiwania progów tolerancji przedstawioną w [1]. Dla wyznaczonych progów tolerancji zaproponowano algorytmy generowania, skracania i filtracji reguł decyzyjnych, wykorzystujące funkcje oceniające reguły. Przedstawiono wyniki eksperymentów z różnymi zbiorami danych.

CALCULATION OF TOLERANCE THRESHOLDS AND APPROXIMATE DECISION RULES

Summary. In this article a method of serching tolerance thresholds is presented. The method base on genetic algorithm and various fitness functions. An inspiration for this research was drawn from a method presented by Stepaniuk and Krętowski in [1]. For calculated tolerance thresholds the methods of generating, shortening and filtering decision rules were proposed. The methods base on decision rules quality functions. The article includes results of experiments conducted for various data sets.

1. Wstęp

Teoria zbiorów przybliżonych umożliwia wyznaczanie reguł decyzyjnych opisujących zależności, jakie w tablicy decyzyjnej występują pomiędzy wartościami atrybutów warunkowych a wartością atrybutu decyzyjnego [2]. Zbiór reguł o tej samej wartości atrybutu decyzyjnego traktowany jest jako opis klasy decyzyjnej i może być wykorzystany do automatycznej klasyfikacji obiektów nieznanych do odpowiadających im klas decyzyjnych jedynie na podstawie znajomości atrybutów warunkowych. Klasyczna teoria zbiorów

przybliżonych opiera się na relacji nierozróżnialności (która jest relacją równoważności) ustalającej podział zbioru obiektów. Jednakże w wielu przypadkach, kiedy analizuje się tablicę decyzyjną, wygodniej jest zamiast nierozróżnialności obiektów badać ich podobieństwo. Przykładem mogą być zbiory danych zawierające dane niepewne lub nie poddane wcześniejszej dyskretyzacji atrybuty typu ciągłego. O tym, czy dwa obiekty są do siebie podobne, decyduje zdefiniowana dla badanej tablicy decyzyjnej relacja podobieństwa (tolerancji), która nie musi być przechodnia i w związku z tym wyznacza pokrycie zbioru obiektów znajdujących się w tablicy decyzyjnej. Przy ustalonej formie relacji tolerancji oraz ustalonych miarach odległości obiektów przed wyznaczeniem zbioru reguły decyzyjnych, konieczne jest dobranie dla danej tablicy decyzyjnej odpowiednich progów tolerancji [3]. W niniejszym artykule przedstawiamy zmodyfikowaną metodę wyznaczania quasi-optymalnych progów tolerancji opisaną w [1]. Przedstawiona metoda pozwoliła na uzyskiwanie mniejszej ilości wyznaczanych reguły decyzyjnych oraz zbliżonej lub większej testowej dokładności klasyfikacji otrzymanej przez wyznaczone zbiory reguł decyzyjnych.

Zbiory przybliżone wyznaczane przez relację tolerancji nie gwarantują, iż wszystkie uzyskane reguły decyzyjne będą dokładne. Reguły niedokładne nazywa się regułami aproksymacyjnymi. W zastosowaniach algorytmów uczących się reguł decyzyjnych w dziedzinie znanej jako *data mining* chodzi głównie o wyznaczanie takich reguł, które opisują silne, nieznane wcześniej zależności występujące w danych. Aby znaleźć w zbiorze reguł, reguły najsilniejsze, używa się różnych funkcji mających za zadanie oceniać jakość danej reguły. Wartość funkcji oceniającej regułę może być użyta jako kryterium budowania reguły [4][5][6][7]. W pracy proponujemy metodę wyznaczania reguły decyzyjnej wykorzystującą wiersz macierzy rozróżnialności oraz funkcję oceniającą regułę. Na koniec zwracamy uwagę na to, iż prosty algorytm filtracji reguł pozwala na ograniczenie ilości stosowanych w klasyfikacji reguł decyzyjnych bez znacznej straty testowej dokładności klasyfikacji.

2. Podstawowe definicje i przyjęte założenia

Niech $DT=(U,A\cup\{d\})$ będzie tablicą decyzyjną, gdzie U jest zbiorem obiektów, A zbiorem atrybutów warunkowych oraz d jest atrybutem decyzyjnym. Relacja $\tau \subseteq U\times U$ jest relacją podobieństwa (tolerancji), czyli jest zwrotna i symetryczna.

Dla każdego atrybutu $a\in A$ zdefiniowana jest funkcja odległości $\delta_a: D_a\times D_a\rightarrow[0,\infty]$, gdzie D_a jest zbiorem wartości przyjmowanych przez atrybut a w DT o własnościach $\forall x,y\in U$ $\delta_a(a(x),a(x))=0$ i $\delta_a(a(x),a(y))=\delta_a(a(y),a(x))$.

Dla każdego atrybutu $a\in A$ ustalony jest próg tolerancji (podobieństwa) ε_a.

Przy ustalonym atrybucie $a \in A$ i ε_a relacja $\tau_a(\varepsilon_a)$ zdefiniowana jest następująco:

$$\forall x,y \in U \; \langle x,y \rangle \in \tau_a(\varepsilon_a) \Leftrightarrow \delta_a(a(x),a(y)) \leq \varepsilon_a$$

W przypadku kiedy rozpatrywany jest podzbiór atrybutów $B \subseteq A$, to zgodnie z przyjętymi założeniami dla każdego atrybutu $a_i \in B$ $i=1,2,..,n$ ustalony jest próg tolerancji ε_{a_i}, wtedy relacja $\tau_B(\varepsilon_{a_1},\varepsilon_{a_2},...,\varepsilon_{a_n})$ definiowana jest następująco:

$$\forall x,y \in U \; \langle x,y \rangle \in \tau_B(\varepsilon_{a_1},\varepsilon_{a_2},...,\varepsilon_{a_n}) \Leftrightarrow \; \forall a_i \in B \; [\, \delta_{a_i}(a_i(x),a_i(y)) \leq \varepsilon_{a_i} \,].$$

Relacja tolerancji przedstawiona powyżej jest w tzw. postaci koniunkcyjnej.

Zbiór obiektów podobnych do obiektu $x \in U$ ze względu na zbiór atrybutów $B \subseteq A$ określa wartość funkcji niepewności $I_B: U \rightarrow 2^U$ $I_B(x)$ zdefiniowana następująco:

$$\forall y \in U \; y \in I_B(x) \Leftrightarrow \langle x,y \rangle \in \tau_B(\varepsilon_{a_1},\varepsilon_{a_2},...,\varepsilon_{a_n})$$

Innymi słowy, zbiór $I_B(x)$ jest zbiorem tolerancji elementu x.

Przy wprowadzonych definicjach przyjmuje się, że klasa decyzyjna $X_v \subseteq U$ jest B-definiowalna, jeżeli jest sumą mnogościową zbiorów będących wartościami funkcji niepewności dla pewnych obiektów z U. Nie dla każdego zbioru atrybutów $B \subseteq A$ klasa decyzyjna będzie B-definiowalna, można wtedy jednak podać przybliżoną B-definicję takiej klasy. Aby móc to zrobić, konieczne jest określenie miary zawierania się dwóch zbiorów, pozwala na to inkluzja przybliżona, która jest uogólnieniem pojęcia przybliżonej funkcji przynależności [8]. Inkluzja przybliżona jest funkcją określoną następująco $v: 2^U \times 2^U \rightarrow [0,1]$, w pracach [8][9] rozważane są warunki, jakie powinna spełniać inkluzja przybliżona, w szczególności funkcja ta powinna być monotoniczna ze względu na swój drugi argument. W artykule przyjmujemy następującą postać inkluzji przybliżonej:

$$v(X,Y) = \begin{cases} \dfrac{card\ (X \cap Y)}{card\ (X)} & \text{dla} \quad X \neq \varnothing \\ 1, & \text{dla} \quad X = \varnothing \end{cases},$$

którą nazywa się standardową inkluzją przybliżoną i oznacza się v_{SRI} [3]. W świetle przyjętych założeń dla tablicy decyzyjnej $DT=(U,A \cup \{d\})$, zbioru $B \subseteq A$, B-dolne i B-górne przybliżenie zbioru w $X \subseteq U$ może zostać zdefiniowane następująco [3]:

$$\underline{B}X = \{\, x \in U: v_{SRI}(I_B(x),X) = 1 \,\}, \quad \overline{B}X = \{\, x \in U: v_{SRI}(I_B(x),X) > 0 \,\}.$$

Aby wyznaczyć zbiory $I_B(x)$, konieczne jest wybranie konkretnej postaci miary podobieństwa pomiędzy obiektami ze zbioru U. W zależności od tego, jakiego typu jest atrybut, możemy funkcję odległości zdefiniować w różny sposób, w artykule rozpatrujemy następujące postaci miar podobieństwa:

1. *Jeśli $a \in A$ jest atrybutem typu rzeczywistego, to*

$$\delta_a(a(x),a(y)) = \frac{|a(x)-a(y)|}{\max D_a - \min D_a};$$

2. *Jeśli $a \in A$ jest atrybutem symbolicznym, to*

$$\delta_a(a(x),a(y)) = \frac{1}{k}\sum_{i=1}^{k}(v_{SRI}(X_{a(x)},X_{v_i}) - v_{SRI}(X_{a(y)},X_{v_i}))^2 .$$

Gdzie przez $maxD_a$ *i* $minD_a$ oznacza się odpowiednio największą i najmniejszą wartość atrybutu $a \in A$ występującą w *DT*, a $X_{v_1}, X_{v_2}, .., X_{v_k}$ są klasami decyzyjnymi. Dodatkowo, dla każdego $a \in A$, $x \in U$, $X_{a(x)}=\{u \in U: a(u)=a(x)\}$. Obie funkcje są znane i często wykorzystywane [3].

Kiedy wybrana jest miara podobieństwa obiektów oraz dla każdego obiektu $x \in U$ i każdego atrybutu $a \in A$ znane są progi tolerancji ε_a, możliwe staje się wyznaczenie zbioru wszystkich minimalnych reguł decyzyjnych w tablicy decyzyjnej [2], proces ten jest równoznaczny z wyznaczeniem reduktów relatywnych dla każdego obiektu ze zbioru *U* [2]. Zbiór takich reduktów wyznacza się analizując uogólnioną macierz rozróżnialności modulo d [10].

Definicja 1. *Macierz rozróżnialności modulo d (uogólniona)*

Niech $DT=(U,A\cup\{d\})$, $U=\{x_1,x_2,..,x_n\}$ *uogólnioną macierzą rozróżnialności tablicy* ***DT*** *nazywamy macierz kwadratową* $M_d(DT)=\{c_{ij}: 1\le i,j\le n\}$ *o elementach zdefiniowanych następująco:*

$$c_{ij}=\{a \in A:(\langle x_i,x_j\rangle \notin \tau_a(\varepsilon_a)) \wedge (d(x_i)\neq d(x_j)\} = \{a \in A: (x_j \notin I_a(x_i)) \wedge (d(x_i)\neq d(x_j))\}.$$

$c_{ij}=\varnothing$ *jeśli* $d(x_i)=d(x_j)\}$.

Definicja 2. *Redukt relatywny dla obiektu [3]*

Niech dany jest $DT=(U,A\cup\{d\})$. *Podzbiór* $B\subseteq A$ $B=\{a_1,..,a_k\}$ *nazywamy reduktem relatywnym dla obiektu* $x \in U$, *wtedy i tylko wtedy, gdy:*

1. $\{y \in I_B(x): d(y)\neq d(x)\}=\{y \in I_A(x): d(y)\neq d(x)\}$;
2. *Dla dowolnego* $C\subset B$ *pierwszy warunek nie jest prawdziwy.*

Definicja 3. *Funkcja odróżnialności modulo d dla obiektu [10]*

Niech dane są, $DT=(U,A\cup\{d\})$, $U=\{x_1,x_2,..,x_n\}$, $A=\{a_1,a_2,..,a_m\}$ *oraz* $M(DT)$. *Funkcją odróżnialności obiektu* $x_i \in U$, *nazywamy funkcję Boolowską zmiennych* $a_1^*,a_2^*,..,a_m^*$ *odpowiadających odpowiednio atrybutom* $a_1,a_2,..,a_m \in A$ *zdefiniowaną następująco:*

$$f_{x_i}^d(a_1^*,a_2^*,..,a_m^*) = \wedge\{\vee(c_{ij} : 1 \le j \le n)\}$$

gdzie $\vee(c_{ij})$ *jest alternatywą wszystkich zmiennych* $a^* \in \{a_1^*,a_2^*,..,a_m^*\}$, *takich że* $a \in c_{ij}$.

Twierdzenie 1

Dla danej tablicy decyzyjnej $DT=(U,A\cup\{d\})$, *zbioru* $\{a_1,a_2,..,a_k\}\subseteq A$, *zmiennych boolowskich* $a_1^*,a_2^*,..,a_k^*$ *odpowiadających atrybutom* $a_1,a_2,..,a_k$ *następujące warunki są równoważne [3]:*

1. $\{a_1,a_2,..,a_k\}$ *jest reduktem dla obiektu* $x\in U$ *w systemie* DT,
2. $a_1^*\wedge a_2^*\wedge..\wedge a_k^*$ *jest implikantem pierwszym funkcji Boolowskiej* $f^d_{x_i}$.

Podane wyżej definicje i twierdzenie pozwalają wyznaczyć zbiór wszystkich reduktów relatywnych dla każdego obiektu należącego do zbioru U. Zbiór wszystkich reduktów relatywnych dla obiektu $x\in U$ w tablicy decyzyjnej $DT=(U,A\cup\{d\})$ oznaczymy przez $RED(DT,d)$. Zbiór minimalnych reguł decyzyjnych dla obiektu $x\in U$ takiego, że $d(x)=v$ definiowany jest następująco:

$$RUL(DT,x)=\{(a_1,V_{a_1})\wedge..\wedge(a_k,V_{a_k})\rightarrow(d,d(x))) : a_i\in B\ i=1,..,k\ B\in RED(DT,d)\}$$

Zbiór wszystkich minimalnych reguł decyzyjnych w tablicy decyzyjnej $DT=(U,A\cup\{d\})$ definiuje się jako $RUL(DT)=\bigcup_{x\in U} RUL(DT,x)$.

Pojedyncza reguła decyzyjna jest zatem postaci $(a_1,V_{a_1})\wedge..\wedge(a_k,V_{a_k})\rightarrow(d,v)$, dowolne (a,V_a) nazywane jest deskryptorem oraz $V_a\subseteq D_a$ a v jest pewną wartością atrybutu decyzyjnego $(v\in D_d)$. Kiedy wyznaczana jest reguła dla konkretnego obiektu $x\in U$, zakres deskryptora V_a wyznacza się na podstawie znajomości progu tolerancji ε_a, w sposób następujący $V_a=\{a(y)\in D_a: y\in I_a(x)\}$.

Znane są algorytmy wyznaczania wszystkich minimalnych reguł decyzyjnych dla danej tablicy decyzyjnej, jednakże ze względu na ich złożoność obliczeniową oraz fakt, iż zazwyczaj pożądane są reguły z małą ilością deskryptorów warunkowych, opracowano metody heurystyczne wyznaczania prawie minimalnych reguł decyzyjnych [3][11].

Weryfikacją jakości wyznaczonych reguł decyzyjnych jest ich testowa dokładność klasyfikacji. W artykule przyjęliśmy, że dla każdego obiektu testowego u wyznaczany jest stopień zaufania tego obiektu do każdej klasy decyzyjnej, zgodnie ze wzorem:

$$confidence(X_v,u)=\sum_{r\in RUL_{X_v}(DT),dist(r,u)\le\varepsilon}(1\text{-}dist(r,u))confidence(r)$$

gdzie $RUL_{X_v}(DT)$ jest zbiorem reguł opisujących klasę decyzyjną X_v; $dist(r,u)$ jest odległością obiektu testowego u od reguły r, w szczególności jeżeli $\varepsilon=0$, to klasyfikacja odbywa się przez reguły pasujące, czyli rozpoznawane przez obiekt testowy u; $confidence(r)$ jest miarą odzwierciedlającą stopień zaufania do reguły r, inaczej mówiąc siłę reguły r, może to być np. dokładność reguły (*accuracy*). Funkcja decyzyjna f przyporządkowująca

dowolnemu obiektowi wartość atrybutu decyzyjnego (klasę decyzyjna, do jakiej obiekt przynależy) określona jest wzorem:

$$f(u)=max\{confidence(X_v,u): v \in V_d\},$$

gdzie u-obiekt testowy.

Teraz przejdziemy do omówienia metod wyznaczania progów tolerancji oraz wyznaczania reguł decyzyjnych z wykorzystaniem funkcji odróżnialności dla obiektu bez wyznaczania implikantów pierwszych tej funkcji.

3. Wyznaczanie progów tolerancji

W przeciwieństwie do postaci relacji tolerancji oraz miar podobieństwa arbitralny wybór wartości progów tolerancji nie jest rzeczą oczywistą ze względu na zazwyczaj dużą liczbę możliwych wartości progu dla danego $a \in A$. Głównym celem jest wyznaczenie wektora progów tolerancji $(\varepsilon_{a_1}, \varepsilon_{a_2}, .., \varepsilon_{a_n})$, $a_i \in A$, tak aby zbiór wyznaczonych reguł decyzyjnych miał jak największą testową dokładność klasyfikacji.

Jeżeli dane są $DT=(U,A \cup \{d\})$ i δ_a dla każdego $a \in A$ to z DT można zbudować nową tablicę $DT`=(U`,A` \cup \{D\})$, gdzie $U`=\{<x_i,y_j>: (<x_i,y_j> \in U \times U) \wedge (i \leq j)\}$

$$A`=\{a`: U` \to R^+ : a`(<x,y>)=\delta_a(a(x),a(y))\},$$

$$D(<x,y>)=\begin{cases} 0 & dla \quad d(x)=d(y) \\ 1 & wpp. \end{cases}$$

Narzucającym się sposobem znalezienia wektora epsilonów jest rozpatrzenie wszystkich możliwych wektorów $(\varepsilon_{a_1}, \varepsilon_{a_2}, .., \varepsilon_{a_n})$, $a_i \in A$, każdorazowe wyznaczenie reguł decyzyjnych i sprawdzenie ich zdolności klasyfikacyjnych, jednakże jest to proces bardzo złożony obliczeniowo. Łatwo zauważyć, że sortując obiekty ze zbioru $U`$ rosnąco ze względu na wartości dowolnego $a` \in A`$, otrzymamy wszystkie możliwe wartości, dla których zmienia się moc zbioru $I_a(x)$, $x \in U$ [3]. Są to wszystkie sensowne wartości ε_a, jakie należałoby rozpatrzyć. W przypadku kiedy $card(U)=n$, $card(A)=k$ i założymy, że dla każdego $a \in A$, $card(V_a)=n$, to należałoby sprawdzić $\frac{1}{2}\prod_{a \in A}(card(V_a)^2 - card(V_a)) + 1$ wektorów epsilonów [3]. Można zauważyć, iż modyfikując zbiór $U`$ w sposób następujący:

1. $U`=\{x\} \times U$
2. $U`=X_v \times U$, *gdzie* $X_v=\{x \in U, d(x)=v\}$

Wyznaczymy progi tolerancji odpowiednio dla każdego obiektu oraz dla każdej klasy decyzyjnej osobno, jednakże wtedy relacja $\tau_A(\varepsilon_{a_1}, \varepsilon_{a_2}, ..., \varepsilon_{a_n})$ może nie być symetryczna. Taką relację nazywa się słabą relacją tolerancji.

W pracy [1] przedstawiono heurystyczną metodę poszukiwania wektora progów tolerancji. Jako strategię przeszukiwania zbiorów rozwiązań wybrano algorytm genetyczny. Pojedynczy osobnik reprezentowany jest jako wektor progów ε. Dokładnej dla każdego $a \in A$, sensowne wartości ε_a ponumerowano, w ten sposób otrzymano ciąg liczb całkowitych, każda liczba odpowiadała jednej z możliwych wartości progu epsilon. Pojedynczy osobnik był zatem wektorem liczb całkowitych, np. zakładając, że w tablicy decyzyjnej są trzy atrybuty warunkowe, osobnik (2,3,2) oznacza, iż wybrano drugi próg epsilon dla atrybutu a_1, trzeci dla a_2 i drugi dla a_3. Taka reprezentacja osobnika prowadzi do sytuacji, iż przeszukiwany jest pewien ograniczony rozmiarem populacji i prawdopodobieństwem, z jakim występuje mutacja (powinno ono być małe) zbiór możliwych wektorów progów epsilon. Inaczej - tylko mutacja może wprowadzić do populacji nowe, nie wylosowane do populacji początkowej, wartości progów epsilon.

W pracy [1] wybierano do początkowej populacji tylko te osobniki, których wartość funkcji przystosowania była większa od zadanego progu, pozwalało to poszukiwać lepszego rozwiązania wśród rozwiązań już w pewnym stopniu dobrych. Z naszych doświadczeń wynika, iż proces wybierania osobników wystarczająco dobrych jest procesem czasochłonnym, zwłaszcza jeżeli w tablicy znajdują się nie poddane dyskretyzacji atrybuty o wartościach rzeczywistych.

Pierwsza nasza modyfikacja polegała zatem na znalezieniu jednej wartości ε, takiej samej dla wszystkich atrybutów warunkowych, takiej że wartość funkcji przystosowania wektora złożonego z tych samych wartości ε była wysoka. Osobniki do populacji początkowej tworzone były w ten sposób, że ustalony wcześniej wektor takich samych epsilonów podlegał mutacji (losowej zmianie). W ten sposób szybciej uzyskiwaliśmy populację początkową, składającą się z dobrych osobników.

Aby umożliwić pojawianie się nowych wartości progów epsilon w wyniku krzyżowania, przyjęliśmy binarną reprezentację pojedynczego osobnika. Każdemu atrybutowi przyporządkowujemy pewną liczbę bitów w binarnym wektorze stanowiącym osobnika, niezbędną do zakodowania wszystkich możliwych dla tego atrybutu numerów progów epsilon. Pojawia się tu pewien problem z interpretacją niektórych ciągów bitów, które mogą się pojawić w wyniku mutacji lub krzyżowania. Załóżmy, że atrybut a ma 10 możliwych wartości progu epsilon. W związku z tym musimy zarezerwować dla niego cztery bity. Jednak tylko ciągi bitów odpowiadające liczbom z zakresu od 0 do 9 dają się zinterpretować jako progi epsilon możliwe dla atrybutu a. Ciąg 1111 nie posiada interpretacji, jednak w wyniku mutacji

lub krzyżowania może się on pojawić. Problem rozwiązaliśmy w sposób następujący: załóżmy, że atrybut a posiada n_a możliwych wartości progu epsilon, w związku z tym zarezerwowano dla niego k bitów, przy czym $2^{k-1} < n_a \leq 2^k$. Każdą k-bitową liczbę b odpowiadającą w binarnej reprezentacji osobnika atrybutowi a przeliczamy na dopuszczalny numer progu epsilon wg wzoru $\frac{b}{2^k - 1} * n_a$. Taka reprezentacja osobnika daje szansę na rozpatrzenie wszystkich możliwych wektorów epsilonów.

W algorytmie wykorzystaliśmy standardową operację krzyżowania oraz operację mutacji [12], obie operacje zachodziły z zadanym prawdopodobieństwem.

Najistotniejszą zmianą było jednak użycie innych funkcji przystosowania osobnika. W pracy [3] jako funkcję przystosowania osobnika wykorzystano funkcję o postaci:

$$w\ \gamma_B(d) + (1-w) v_{SRI}(R_d, R_{I_A}) \tag{1}$$

gdzie:

$$\gamma_B(d) = \frac{card(POS_B(d))}{card(U)},$$

$POS_B(d)$ jest B-obszarem pozytywnym [2] tablicy decyzyjnej $DT=(U,A\cup\{d\})$ oraz: $R_d=\{<x,y> \in U\times U : d(x)=d(y)\}$ $R_{I_A} = \{< x, y > \in U\times U : y \in I_A(x)\}$ i $0<w\leq 1$.

Jesteśmy zainteresowani znalezieniem takich wartości progów tolerancji, aby jak najwięcej obiektów mających te same wartości atrybutu decyzyjnego było z sobą w relacji przy równoczesnym ograniczeniu do minimum przypadków, kiedy w relacji są obiekty mające różne wartości atrybutu decyzyjnego. Taką właściwość odzwierciedla przedstawiona powyżej funkcja, którą dla uproszczenia zapisu przyjęliśmy nazywać *Fun1*. Ponieważ w przeprowadzanych przez nas eksperymentach napotykaliśmy kłopoty z odpowiednim ustaleniem wartości wagi w, której wartość okazywała się istotna dla działania całego algorytmu genetycznego, zastosowaliśmy inne funkcje oceniające dany wektor progów epsilon. Teoretycznie z postaci funkcji *Fun1* oraz na podstawie znajomości zbioru treningowego wartość wagi w może być ustalona na drodze obliczeń. Ustalając arbitralnie, o ile powinien wzrosnąć drugi składnik funkcji *Fun1*, kiedy zmaleje jej pierwszy składnik, tak aby wartość całej funkcji nie uległa zmianie, można wyliczyć wartość wagi w pozwalającą spełnić ten warunek.

Innymi możliwymi do zastosowania funkcjami oceniającymi stopień przystosowania osobnika w populacji mogą być odpowiednio adaptowane do tego celu funkcje oceniające jakość reguły. Funkcje oceniające jakość reguły pozwalają ocenić daną regułę ze względu na jej dokładność oraz typowość zależności, jaką reguła przedstawia (pokrycie) [5]. Ponieważ dokładność reguły zazwyczaj maleje, kiedy rośnie jej pokrycie oraz odwrotnie, a dzieje się tak również w przypadku składników funkcji *Fun1*, można do oceny wektora progów tolerancji

wykorzystać zaadaptowane funkcje oceniające regułę. W artykule rozpatrzono trzy takie funkcje: zaproponowaną przez R. S. Michalskiego [7], zastosowaną w programie IREP [6], oraz funkcję wykorzystywaną w teście niezależności χ^2 (statystykę) znaną jako funkcja Pearsona [5]. Wartości wszystkich trzech funkcji da się wyliczyć na podstawie analizy tablicy kontyngencji pozwalającej opisać zachowanie reguły w stosunku do zbioru treningowego. Aby funkcje te stosować do oceny wektora progów tolerancji, wystarczy zmodyfikować tablice kontyngencji, na podstawie której obliczana będzie ich wartość.

Jeżeli dane są $DT=(U,A\cup\{d\})$, $A=\{a_1,..,a_k\}$ koniunkcyjna postać relacji tolerancji oraz miary podobieństwa obiektów, to można skonstruować tablicę decyzyjną $DT`=(U`,A`\cup\{D\})$. Dla tablicy $DT`$ oraz dowolnego wektora progów tolerancji $(\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n})$ można wyznaczyć macierz przedstawioną w tabeli 1.

Tabela 1

Macierz, na podstawie której wyznaczamy wartości funkcji oceny jakości progów tolerancji

$n_{R_d R_{IA}}$	$n_{R_d \neg R_{IA}}$	n_{R_d}
$n_{\neg R_d R_{IA}}$	$n_{\neg R_d \neg R_{IA}}$	$n_{\neg R_d}$
$n_{R_{IA}}$	$n_{\neg R_{IA}}$	

gdzie: $R_{IA}=R_{I_A}$

$n_{R_d}=n_{R_d R_{IA}}+n_{R_d \neg R_{IA}}$ liczba par obiektów mających tę samą wartość atrybutu decyzyjnego;

$n_{\neg R_d}=n_{\neg R_d R_{IA}}+n_{\neg R_d \neg R_{IA}}$ liczba par obiektów mających różne wartości atrybutu decyzyjnego;

$n_{R_{IA}}=n_{R_d R_{IA}}+n_{\neg R_d R_{IA}}$ liczba par obiektów będących w relacji $\tau_A(\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n})$;

$n_{\neg R_{IA}}=n_{R_d \neg R_{IA}}+n_{\neg R_d \neg R_{IA}}$ liczba par obiektów nie będących w relacji $\tau_A(\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n})$;

$n_{R_d R_{IA}}$ liczba par obiektów mających taką sama wartość atrybutu decyzyjnego i będących w relacji $\tau_A(\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n})$. Wielkości $n_{\neg R_d R_{IA}}, n_{R_d \neg R_{IA}}, n_{\neg R_d \neg R_{IA}}$ mają znaczenie podobne jak $n_{R_d R_{IA}}$.

Funkcje oceniające dany wektor progów tolerancji $(\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n})$ mają następującą postać:

1. $q^{Michalski}((\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n}))=\frac{n_{R_d R_{IA}}}{n_{R_d}}*w+\frac{n_{R_d R_{IA}}}{n_{R_{IA}}}*(1-w)\quad w\in(0,1]$;

2. $q^{IREP}((\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n}))=\frac{n_{R_d R_{IA}}+n_{\neg R_{IA}}-n_{R_d \neg R_{IA}}}{card(U`)}$;

3. $q^{Pearson}((\varepsilon_{a_1},\varepsilon_{a_2},..,\varepsilon_{a_n}))=\frac{\left(n_{R_d R_{IA}} n_{\neg R_d \neg R_{IA}}-n_{R_d \neg R_{IA}} n_{\neg R_d R_{IA}}\right)^2}{n_{R_{IA}} n_{\neg R_{IA}} n_{R_d} n_{\neg R_d}}$.

Takich funkcji używaliśmy jako funkcji oceniających przystosowanie osobnika w populacji. Chcieliśmy, aby wyznaczone progi tolerancji były na tyle duże, by uzyskiwane reguły miały szerokie deskryptory oraz wysoką zdolność klasyfikacji obiektów testowych, wyniki eksperymentów zaprezentowano w ostatnim rozdziale artykułu.

4. Wyznaczanie aproksymacyjnych reguł decyzyjnych

Kiedy zostaną ustalone progi tolerancji, reguły decyzyjne wyznaczone dla tablicy decyzyjnej nie zawsze są dokładne, wyznaczając wszystkie redukty dla każdego obiektu w tablicy decyzyjnej uzyska się zbiór wszystkich minimalnych reguł decyzyjnych. Inne metody wyznaczania reguł decyzyjnych jak chociażby szukanie wzorców decyzyjnych [13], rodzina programów AQ [7] czy wyznaczanie jednego lub zadanej ilości reduktów relatywnych dla obiektu [11], które uzyskują podobną testową dokładność klasyfikacji bez wyznaczania wszystkich możliwych reguł. Proponujemy wyznaczanie jednej reguły dla każdego obiektu z tablicy decyzyjnej, analizując odpowiedni wiersz uogólnionej macierzy rozróżnialności modulo d. Kryterium oceniającym tworzoną regułę jest funkcja oceniająca regułę [4][5], może to być jedna z trzech funkcji przedstawionych w rozdziale 3 (tablica kontyngencji oczywiście jest wtedy inna). Dla uproszczenia zapisu algorytm prezentowany poniżej nazwaliśmy *RMatrix*

Algorytm generacji reguły decyzyjnej na podstawie wiersza macierzy rozróżnialności

dane: *DT=(U, A∪{d}), progi tolerancji*

deklaracje:

r - *reguła decyzyjna; u – obiekt, dla którego wyznaczana jest reguła; q – funkcja oceniająca regułę*

ranking[] - *tablica o rozmiarze card(A), każdy element to struktura zawierająca dwa pola :- atrybut- atrybut warunkowy; licznik- liczba całkowita*

początek

twórz nową regułę r o decyzji, jaką ma obiekt u , nie zawierającą żadnego deskryptora warunkowego

$r_{najlepsza} := r$

$i := 1$

dla każdego *atrybutu* $a \in A$

ranking[i].atrybut $:= a$

ranking[i].licznik := liczba wystąpień atrybutu a w wierszu $M_a(DT)$ *odpowiadającym obiektowi u*

$i := i + 1$

sortuj tablicę ranking[] malejąco względem pola „licznik"

dla $i := 1, \ldots, card(A)$

$a :=$ *ranking[i].atrybut*

konstruuj deskryptor p postaci ($a, I_a(u)$ *)*

dodaj deskryptor p do reguły r

```
        jeżeli q(r) > q(r_najlepsza)
                r_najlepsza := r
    zwróć jako wynik r_najlepsza
koniec
```

Można oczywiście wyznaczyć tym algorytmem tylko te reguły, które wystarczą do pokrycia zbioru treningowego U.

Kiedy reguły decyzyjne zawierają dużo deskryptorów warunkowych, można próbować skracać takie reguły, wykorzystując następujący algorytm skracania reguły:

Algorytm skracania reguł decyzyjnych

```
dane:
    r    - reguła decyzyjna podlegająca skracaniu;
    q    - funkcja oceniająca regułę;
    qmin        - minimalna ocena reguły, poniżej której niemożliwe jest dalsze skracanie reguły
deklaracje:
    d- deskryptor
początek
    μ_max := q(r)
    :pętla
        μ := qmin
        dla każdego deskryptora d zawartego w r      /* znajdź deskryptor d_max */
            usuń deskryptor d z reguły r             /* którego usunięcie */
            jeżeli q(r) > μ                          /* powoduje najmniejszy */
                μ := q(r)                            /* spadek jakości reguły */
                d_max := d
            wstaw deskryptor d do reguły r
        jeżeli μ ≥ μ_max                             /* jeżeli po usunięciu d_max jakość */
            usuń deskryptor d_max z reguły r         /* reguły nie będzie gorsza niż */
            μ_max := μ                               /* do tej pory, to usuń d_max */
            goto pętla
        w przeciwnym razie
            goto koniec
    :koniec
    zwróć jako wynik r
koniec
```

Z eksperymentów przez nas przeprowadzanych wynika, że niezależnie od metody wyznaczania otrzymane reguły decyzyjne można odfiltrowywać, tzn. usuwać z nich reguły słabe lub niepotrzebne do klasyfikacji, w tym artykule wykorzystywaliśmy prosty algorytm filtracji polegający na tym, iż początkowo wyznaczany był zbiór wszystkich reguł, a następnie

poczynając od reguły o najlepszej ocenie wyznaczany był zbiór reguł wystarczający do pokrycia zbioru U, algorytm przedstawia się następująco:

Algorytm filtracji reguł:

dane

q – wybrana funkcja oceniająca regułę; RUL – zbiór reguł; U –obiekty treningowe

deklaracje

RUL - przefiltrowny zbiór reguł

początek

RUL`:=∅

posortuj malejąco reguły według wartości funkcji oceniającej q, RUL:= $< rule_1,..,rule_{card(RUL)} >$

dopóki *card(U)>0*

U:=U \ { u : dist(u,rule$_1$)=0 };

RUL`:=RUL`∪{rule$_1$};

RUL=RUL \ ({rule$_1$}∪{r∈RUL: r została wyznaczona z obiektu u, dla którego dist(u, rule$_1$)=0});

przenumeruj reguły w zbiorze RUL

zwróć jak wynik *RUL`*

koniec

Powyższy algorytm można potraktować jako zmodyfikowany algorytm generowania quasi-minimalnego pokrycia zbioru U, obiekt generator, od którego rozpoczyna się wyznaczanie reguł, determinowany jest jako obiekt, z którego wyznaczona była najlepsza reguła znajdująca się w wejściowym zbiorze reguł.

Wyniki eksperymentów przedstawiono w następnym rozdziale.

5. Eksperymenty z danymi

Do przetestowania działania przedstawionych funkcji oceniających wektor progów tolerancji wybraliśmy trzy zbiory danych znane jako Lymphography (Instytut Onkologii Lubljana), Australian-Credit (Statlog), Ionosphere (Statlog) zawierające odpowiednio atrybuty typu: tylko dyskretnego, dyskretnego i ciągłego, tylko ciągłego.

Zastosowany algorytm genetyczny miał następujące parametry: populacja początkowa 60 osobników, krzyżowanie zachodziło z prawdopodobieństwem P=0.6, mutacja P=0.05, wartość funkcji przystosowania osobnika w każdej populacji była skalowana. Do skalowania używaliśmy funkcji skalującej Boltzmana, temperatura początkowa dla tej funkcji ustalona została na 6.45, a końcowa na 1.45, skok temperatury wynosił 0.2. Do następnej populacji bez krzyżowania i mutacji przechodził zawsze najlepszy osobnik.

Tabela 2

LYMPHOGRAPHY

Reguły – z wszystkich reduktów

funkcja	michalski w=0.2	michalski w=0.5	irep	peason	fun1 w=0.5	fun1 w=0.2	rclacja równow.
$POS_A(d)$	1	0,83	0,38	0,6	0,99	0,87	1
$v_{SRI}(R_d,R_{IA})$	0,024	0,098	0,25	0,18	0,02	0,088	
dokładność klasyfikacji	**0,79**	**0,81**	**0,82**	**0,79**	**0,78**	**0,76**	**0,75**
liczba reguł	**796**	**431**	**189**	**202**	**903**	**486**	**3700**
confidence(r)	michalski w=0.2	michalski w=0.5	irep	χ^2	accuracy	accuracy	accuracy

Tabela 3

CREDIT

Reguły - z wszystkich reduktów

funkcja	michalski w=0.2	michalski w=0.5	irep	pearson	fun1 w=0.5	fun1 w=0.2
$POS_A(d)$	1	0,08	0,08	0,08	1	0,05
$v_{SRI}(R_d,R_{IA})$	0,004	0,45	0,45	0,45	0,004	0,57
dokładność klasyfikacji	**0,82**	**0,85**	**0,85**	**0,86**	**0,84**	**0,78**
liczba reguł	**3026**	**450**	**450**	**450**	**3184**	**632**
confidence(r)	michalski w=0.2	michalski w=0.5	irep	χ^2	accuracy	accuracy

Tabela 4

IONOSPHERE

Reguły - Dziesięć reduktów (możliwie najkrótszych)

funkcja	michalski w=0.2	michalski w=0.5	irep	χ^2	fun1 w=0.5	fun1 w=0.8
POS	1	0,98	0,98	0,98	0,98	1
$v_{SRI}(R_d,R_{IA})$	0,02	0,03	0,03	0,03	0,03	0,02
dokładność klasyfikacji	**0,86**	**0,95**	**0,95**	**0,96**	**0,95**	**0,85**
confidence(r)	michalski w=0.2	michalski w=0.5	irep	χ^2	accuracy	accuracy

Po wyznaczeniu progów tolerancji wyznaczane były reguły decyzyjne i obliczana była średnia dokładność klasyfikacji dla zbioru testowego (średnia arytmetyczna dokładności klasyfikacji klas decyzyjnych). Pierwsze dwie tabele prezentują wyniki uzyskane przy użyciu

metodologii testowania 10-fold cross-validation. Dla danych ze zbioru Ionosphere do nauki i testowania użyto oryginalnych zbiorów ionosphere.train i ionosphere.test.

W klasyfikacji brały udział tylko reguły dokładnie rozpoznawane przez obiekt testowy, czyli takie, dla których odległość obiektu testowego od reguły była równa zero.

Przedstawione wyniki sugerują, że wszystkie prezentowane funkcje pozwalają uzyskiwać podobną testową dokładność klasyfikacji i nie można wśród nich wskazać zdecydowanie lepszej. Za stosowaniem funkcji IREP lub Pearsona może przemawiać to, iż dla tych funkcji nie jest konieczne ustalanie wartości wagi w oraz to, iż wyznaczane reguły mają szersze deskryptory, dzięki czemu jest ich mniej. Szerokie deskryptory powodują oczywiście to, że reguły mogą być niedokładne.

Algorytm skracania reguł zastosowano do uzyskanych powyżej zbiorów reguł, najlepsze wyniki przedstawiono poniżej.

Tabela 5

Skracanie reguł

Zbiór danych	Funkcja skracającą (sterująca skracaniem)	Maksymalny procentowy akceptowany spadek wartości funkcji skracającej	Funkcja użyta do wyznaczenia progów tolerancji	Dokładność klasyfikacji	Liczba reguł
LYMHO	accuracy	10	irep	0,84	139
CREDIT	accuracy	20	pearson	0,85	39

W pozostałych przypadkach skracanie nie powodowało zwiększenia dokładności klasyfikacji lub zmniejszenia zbioru reguł. Po przeanalizowaniu postaci reguł nasuwa się wniosek, iż reguł, w których jest mało (w stosunku do ilości atrybutów w tablicy treningowej) deskryptorów warunkowych, nie opłaca się poddawać skracaniu, gdyż mocno traci się wtedy na dokładności klasyfikacji. Reguły dłuższe należy próbować skracać.

Poniżej przedstawiono wyniki uzyskane algorytmem *RMatrix* oraz wyniki pokazujące efektywność działania algorytmu filtracji. W przypadku przedstawionego przez nas algorytmu filtracji okazało się, iż zmniejszał on znacznie liczbę reguł koniecznych do pokrycia zbioru treningowego (bardziej niż zwykłe pokrycie), jednak tak uzyskany zbiór reguł miał mniejszą (czasem znacznie) zdolność klasyfikacji. Wprowadziliśmy zatem taką modyfikację do algorytmu, aby funkcja oceniająca regułę używana w tym algorytmie filtracji obliczana była nieco inaczej. Dokładniej, jeżeli w algorytmie używaliśmy funkcji oceniającej q (np. Pearsona), to w zmodyfikowanej wersji algorytmu wartości funkcji q dla reguły r obliczaliśmy według wzoru:

$q(r)_{new}=q(r)_{unique}/q(r)_{normal}$

gdzie $q(r)_{new}$ nowa wartość funkcji, $q(r)_{unique}$ wartość funkcji q uwzględniająca tylko te obiekty ze zbioru treningowego, które unikalnie rozpoznają regułę r, $q(r)_{normal}$ wartość funkcji q obliczana w sposób typowy.

Tabela 6

Wyznaczanie reguły decyzyjnej z wiersza macierzy rozróżnialności

	funkcja nadzorująca tworzenie reguły	skracanie	funkcja klasyfikująca	dokładność klasyfikacji	liczba reguł
LYMPHO	irep	-	accuracy	0,82	26
CREDIT	pearson	accuracy 10%	accuracy	0,87	63

W tabeli przedstawiono najlepsze uzyskane wyniki. Przy ustalonych progach tolerancji wyznaczano reguły używając do sterowania procesem tworzenia reguły różnych funkcji oceniających regułę. Ciekawym wynikiem jest to, iż zarówno dla zbioru LYMPHOGRPAHY, jak i CREDIT najlepsze wyniki uzyskano wtedy, gdy funkcją nadzorującą proces tworzenia reguły była dokładnie ta sama funkcja jak nadzorująca wyznaczanie progów tolerancji.

Po filtracji zbioru reguł konieczna okazała się zmiana sposobu klasyfikacji. Klasyfikacja odbywała się przez najbliższe reguły, dokładniej w klasyfikacji brały udział wszystkie reguły, gdyż maksymalną odległość obiektu testowego od reguły ustalono na jeden. Najlepsze wyniki uzyskano, kiedy funkcją oceniającą siłę reguły w procesie klasyfikacji była dokładność reguły. Uzyskane poniżej wyniki należy porównywać z najlepszymi wynikami otrzymanymi przez odpowiadające metody wyznaczania reguł bez używania algorytmu filtracji prezentowanymi we wcześniejszych tabelach.

Filtracja pozwalała na dalsze ograniczanie zbioru reguł bez znacznego spadku dokładności klasyfikacji, w przedstawionych przypadkach algorytm filtracji działał lepiej, niż gdyby wyliczać reguły z pokrycia bez wyznaczania obiektu generatora (tzn. startując z pierwszego obiektu treningowego).

Dla zbioru IONSPHERE algorytmy *RMatrix* oraz skracania i filtracji reguł nie działały dobrze, po skracaniu i filtracji liczba reguł się co prawda zmniejszała, jednak uzyskiwana dokładność klasyfikacji była rzędu 0,86, a więc znacząco spadła.. Stosowanie filtracji do zbioru reguł wyznaczonych algorytmem *RMatrix* dawało i w tym przypadku lepsze wyniki niż wyznaczanie tym algorytmem reguł wystarczających do pokrycia zbioru treningowego.

Tabela 7

Filtracja reguł

	algorytm wyznaczania reguł	skracanie	sposób obliczania wartości q(r)	czy reguły wyznaczano z pokrycia zbioru treningowego	dokładność klasyfikacji	liczba reguł
LYMPHO	redukty	nie	q_{normal}	nie	0,76	29
	rmatrix	nie	q_{new}	nie	0,79	9
	rmatrix	nie	-	tak	0,77	12
CREDIT	redukty	nie	q_{new}	nie	0,85	23
	rmatrix	accuracy 10%	-	nie	0,87	63
	rmatrix	accuracy 10%	-	tak	0,87	12
	rmatrix	accuracy 10%	q_{new}	nie	0,85	5

6. Wnioski

Na podstawie przeprowadzonych eksperymentów nasuwają się następujące wnioski:

- przedstawione funkcje oceniające jakość wektora progów tolerancji mogą być stosowane z równym powodzeniem jak funkcja Fun1, używając funkcji IREP i Pearsona nie trzeba dobierać wartości wagi *w*, dodatkowo uzyskuje się zbiory tolerancji o większej mocy, a przez to mniej reguł o szerszych deskryptorach. Reguły takie są niedokładne (aproksymacyjne), ale charakteryzują się większym pokryciem;
- nie jest opłacalne skracanie już krótkich reguł decyzyjnych, gdyż powoduje to spadek dokładności klasyfikacji, skracanie długich reguł decyzyjnych może być opłacalne;
- z przeprowadzanych eksperymentów wynika, że należałoby zaimplementować program skracający reguły z wybranej klasy decyzyjnej lub kiedy reguła ma więcej niż zadaną liczbę deskryptorów warunkowych;

- wyznaczanie jednej reguły z wiersza macierzy rozróżnialności (algorytm *RMatrix*) nie zawsze daje dobre wyniki, nasze eksperymenty sugerują, że wyniki te są tym gorsze, im więcej jest atrybutów ciągłych w zbiorze treningowym;
- algorytm filtracji zbioru reguł (zwłaszcza po modyfikacji) daje dobre wyniki i bardziej opłaca się go stosować niż „proste" wyznaczanie pokrycia zbioru treningowego;
- w przypadku kiedy klasyfikujący zbiór reguł jest mały, o sile głosującej reguły powinna decydować dokładność reguły.

Na koniec warto wspomnieć, iż przeprowadzaliśmy eksperymenty z wyznaczaniem progów tolerancji, używając również innych niż algorytm genetyczny strategii poszukiwania rozwiązania. Były to eksperymenty polegające na: wyznaczaniu jednej wartości progu tolerancji dla wszystkich atrybutów takiej samej; szukanie progów tolerancji metodą wspinaczki (ustalając z góry, ile rozpatrujemy możliwych wartości progu epsilon np. od 0 do 1 z krokiem 0.05); łączyliśmy też obie metody. Otrzymywane wyniki nie były lepsze niż te uzyskiwane za pomocą algorytmu genetycznego.

Lepsze wektory progów tolerancji można naszym zdaniem uzyskać stosując bardziej wyszukane metody, jakie oferują algorytmy genetyczne (np. inne metody krzyżowania), na szybkość znajdowania rozwiązań może wpłynąć uprzednie dokonanie dyskretyzacji niesprzecznej (nie zmieniającej obszaru pozytywnego tablicy decyzyjnej) zbioru atrybutów warunkowych, co spowoduje uzyskanie mniejszej ilości możliwych wektorów progów tolerancji.

LITERATURA

1. Stepaniuk J., Krętowski M.: Decision System Based on Tolerance Rough Sets, Proceedings IIS 1995, June 5-9, Augustów, Poland, ICS Polish Academy of Sciences.
2. Pawlak Z.: Rough Sets, Theoretical Aspects of Reasoning about Data, Kluwer Academic Pub. 1991.
3. Stepaniuk J.: Knowledge Discovery by Application of Rough Set Models, Prace IPI PAN, Warszawa 1999.
4. Sikora M., Proksa P.: Funkcje oceny jakości reguł w algorytmie generowania aproksymacyjnych reguł decyzyjnych, ZN Pol. Śl. Studia Informatica Vol. 21, No 3(41), Gliwice 2000.
5. Bruha I.: Quality of decision rules: Definitions and classification schemes for multiple rules. Machine Learning and Statistics:The Interface, John Wiley and Sons, 1995.

6. Furnkranz J., Widmer G.: Incremental Reduced Error Pruning, Proceedings of the Eleventh International Conference of Machine Learning, New Brunswick, NJ, 1994.
7. Kaufman K. A., Michalski R. S.: Learning in Inconsistent World, Rule Selection in STAR/AQ18, Machine Learning and Inference Laboratory, Report, P99-2, 1999.
8. Skowron A., Stepaniuk J.: Tolerance approximation speces, Fundamenta Informaticae 27, 1996.
9. Polkowski L., Skowron A.: "Adaptive Decision-Making by Systems of Cooperating Intelligent Agents Organized on Rough Mereological Principles" Intelligent Automation and Soft Computing", 2(2),1996.
10. Skowron A., Rauszer C.: The discernibility matrices and functions in information systems, In: Inteligent Decision Support, Handbook of Applications and Advances of the Rough Set Theory, Słowiński R. (ed.), Kluwer, Dordrecht, 1992.
11. Nguyen S. H., Nguyen H. S.: Same efficient algorithms for rough set methods, Proceedings Information Processing and Managment of Uncertainty on Knowledge Based Systems (IPMU-96), July 1-5, Granada, Spain, Universidad de Granada, vol.III, 1996.
12. Goldberg E. D.: Algorytmy genetyczne i ich zastosowania, WNT, Warszawa 1998.
13. S.H.Nguyen, A.Skwron, P.Synak: Discovery of Data Patterns with Applications to Decomposition and Classification Problems. In Polkowski L., Skowron A.: Rough Sets in Knowlege Discovery 1, Methodology and Application. Physica-Verlag, Heidelberg 1998.

Recenzent: Dr hab. Jarosław Stepaniuk, prof. Pol. Białostockiej

Wpłynęło do Redakcji 6 lutego 2001 r.

Abstract

In this article we present a method of searching tolerance thresholds. An inspiration for our research was drawn from a method presented by Stepaniuk and Krêtowski. They used genetic algorithm for searching the best thresholds. We introduce some modifications to their approach. First, we proposed binary form for vectors representing set of tolerance thresholds. Second, we adapted the decision rule quality functions as fitness functions. We considered three functions – introduced by R. S. Michalski, used in IREP program and function known as Pearson function. For the tolerance relation we proposed a approximate rules generating algorithm. Our algorithm base on a row of disernibility matrix and is controled by rule quality function. We also proposed simple algorithm of shortening approximate rules and algorithm of

filtering rules set. We conducted experiments with three data sets available in the „Machine Learning Databases" repository: „Lymphography", „Credit screening" and "Ionosphere". As the result of experiments we formulated following main conclusions: 1) Rule quality functions used by us to asses quality of tolerance thresholds work very well. We got larger tolerance sets. For such tolerance relation we got less decision rules and these rules were more general. 2) Filtering of rules set gives better results then generating cover.