Title: Klasyfikacja danych z wykorzystaniem zmodyfikowanego podejścia opartego na dynamicznym programowaniu

Author: Beata Zielosko, Krzysztof Żabiński

Citation style: Zielosko Beata, Żabiński Krzysztof. (2017). Klasyfikacja danych z wykorzystaniem zmodyfikowanego podejścia opartego na dynamicznym programowaniu. “Studia Informatica” (Vol. 38, Nr 2 (2017), s. 45-54)

STUDIA INFORMATICA 2017
Volume 38 Number 2 (131)

Beata ZIELOSKO, Krzysztof ŻABIŃSKI
Uniwersytet Śląski w Katowicach, Instytut Informatyki

KLASYFIKACJA DANYCH Z WYKORZYSTANIEM ZMODYFIKOWANEGO PODEJŚCIA OPARTEGO NA DYNAMICZNYM PROGRAMOWANIU

Streszczenie. Artykuł stanowi kontynuację badań dotyczących zmodyfikowanego algorytmu dynamicznego programowania dla optymalizacji reguł decyzyjnych względem pokrycia. Praca przedstawia wyniki eksperymentalne dotyczące regułowego klasyfikatora, dla zbiorów danych umieszczonych w Repozytorium Uczenia Maszynowego.

Słowa kluczowe: rozszerzenie algorytmu dynamicznego programowania, reguły decyzyjne, pokrycie, klasyfikator regułowy

DATA CLASSIFICATION BASED ON MODIFIED DYNAMIC PROGRAMMING APPROACH

Summary. The article is a continuation of research connected with a modified dynamic programming algorithm for optimization of decision rules relative to coverage. The paper contains experimental results for rule based classifier using data sets from UCI Machine Learning Repository.

Keywords: dynamic programming approach, decision rules, coverage, rule based classifier

1. Wprowadzenie

Reguły decyzyjne stanowią jedną z popularnych form reprezentacji wiedzy, m.in. ze względu na czytelność opisu i łatwość interpretacji. Innym, ważnym zastosowaniem reguł jest konstrukcja klasyfikatorów [11].

Istnieje wiele algorytmów konstruowania reguł decyzyjnych, np. algorytmy oparte na procedurze sekwencyjnego pokrywania [8], wnioskowaniu Boolowskim [7], różne rodzaje algorytmów zachłannych [5, 12] czy algorytmy dynamicznego programowania dla optymalizacji reguł decyzyjnych [13].

Istnieją także różne miary jakości reguł [1, 8, 10]. W niniejszej pracy stosowana jest miara pokrycia, która pozwala na odkrywanie ważnych wzorców w danych.

Istnieje także wiele narzędzi, które pozwalają pracować z regułami decyzyjnymi, np. narzędzia przedstawione w pracach [3, 4, 6, 9].

Artykuł ten stanowi kontynuację badań dotyczących zmodyfikowanego algorytmu dynamicznego programowania dla optymalizacji reguł decyzyjnych względem pokrycia.

Idea algorytmu została zaproponowana w pracy [13]. Tam zostały przedstawione wyniki badań dotyczące przybliżonych reguł decyzyjnych, istotne z punktu widzenia reprezentacji wiedzy. Algorytm pozwala uzyskać reguły decyzyjne, których wartość pokrycia jest bliska wartościom optymalnym, natomiast rozmiar skierowanego grafu acyklicznego, na podstawie którego opisywane są reguły, jest mniejszy. W niniejszej pracy zostaną przedstawione wyniki badań dotyczące m.in. porównania klasyfikatorów regułowych dla algorytmu dynamicznego programowania [5] oraz zmodyfikowanego algorytmu dynamicznego programowania [13] dla dokładnych reguł decyzyjnych.

Artykuł składa się z 5 rozdziałów. W rozdziale 2 zostały przedstawione podstawowe pojęcia. Rozdział 3 prezentuje zmodyfikowany algorytm dynamicznego programowania dla optymalizacji reguł względem pokrycia. Rozdział 4 zawiera wyniki eksperymentów przeprowadzonych na danych umieszczonych w UCI Machine Learning Repository [2]. Rozdział 5 stanowi krótkie podsumowanie.

2. Podstawowe pojęcia

Tablica decyzyjna jest definiowana jako $T = (U, A \cup \{dec\})$ [7], gdzie $U=\{r_1,\ldots,r_m\}$ jest niepustym, skończonym zbiorem obiektów (wierszy), $A = \{f_1,\ldots,f_n\}$ jest niepustym, skończonym zbiorem atrybutów, $f:U\rightarrow V_f$ jest funkcją dla dowolnego $f\in A$, V_f jest zbiorem wartości atrybutu f. Elementy zbioru A są nazywane atrybutami warunkowymi, $dec\in A$ jest wyróżnionym atrybutem, nazywanym atrybutem decyzyjnym. Tablica decyzyjna T jest spójna, tj. nie występują w niej identyczne wiersze o różnych decyzjach.

Tabela uzyskana z T przez usunięcie pewnych wierszy określana jest jako *podtabela*. Niech T będzie niepustą tablicą decyzyjną, $f_{i1},\ldots,f_{im} \in \{f_1,\ldots,f_n\}$ i $a_1,\ldots,a_m$, będą wartościami atrybutów warunkowych, d będzie wartością atrybutu dec. Przez $T(f_{i1},a_1)\ldots(f_{im},a_m)$ jest

oznaczana podtabela T, która zawiera wiersze posiadające wartości $a_1,\ldots,a_m$, na przecięciu z kolumnami $f_{i1},\ldots,f_{im}$. Takie niepuste podtabele (razem z tablicą decyzyjną T) są nazywane *separowalnymi podtabelami T*.

Minimalna wartość atrybutu decyzyjnego przypisana do maksymalnej liczby wierszy w T nazywana jest *najbardziej wspólną decyzją dla T*. Powiemy, że atrybut $f_{i,\, i\in 1,\ldots,n}$, nie jest stały, jeśli posiada przynajmniej dwie różne wartości. Zbiór takich atrybutów jest oznaczany jako $E(T)$. Wartość atrybutu $f_{i,\ i\in 1,\ldots,n}$, przypisana do największej liczby wierszy w tablicy decyzyjnej jest nazywana najczęstszą wartością atrybutu f_i.

W przypadku zmodyfikowanego algorytmu dynamicznego programowania zbiór $E(T)$ zawiera jeden atrybut o minimalnej liczbie wartości oraz pozostałe atrybuty, dla których została określona najczęstsza wartość [13].

Wyrażenie

$$f_{i1} = a_1 \wedge \ldots \wedge f_{im} = a_m \rightarrow d \qquad (1)$$

jest nazywane regułą decyzyjną dla T, jeśli $f_{i1},\ldots, f_{im} \in \{f_1,\ldots,f_n\}$, i $a_1,\ldots,a_m$ są wartościami atrybutów warunkowych oraz d jest wartością atrybutu decyzyjnego. Możliwe, że $m = 0$, wówczas reguła decyzyjna dla T ma postać $\rightarrow d$.

Niech $r=(b_1,\ldots,b_n)$ będzie wierszem tabeli T. Powiemy, że reguła (1) jest *zrealizowana dla r*, jeśli $a_1 = b_{i1},\ldots,a_m = b_{im}$. Jeśli $m=0$, wówczas reguła $\rightarrow d$ jest zrealizowana dla dowolnego wiersza z T. Reguła (1) jest *prawdziwa dla T*, jeśli każdy wiersz, dla którego reguła jest zrealizowana, ma przypisaną decyzję taką jak d. Należy zauważyć, że reguła (1) jest prawdziwa dla T wtedy i tylko wtedy, jeśli tabela $T' = T(f_{i1},a_1)\ldots(f_{im},a_m)$ jest zdegenerowana i każdy wiersz z T ma przypisaną decyzję d. Reguła, która jest prawdziwa dla T i zrealizowana dla r, jest nazywana *regułą decyzyjną dla T i r*.

Niech τ będzie regułą decyzyjną dla T i τ będzie równe (1). *Pokrycie* reguły τ, oznaczane jako $c(\tau)$, to liczba wierszy w T, dla których τ jest zrealizowana i które mają przypisaną decyzję d.

3. Zmodyfikowany algorytm dynamicznego programowania

Zmodyfikowany algorytm dynamicznego programowania, dla danej tablicy decyzyjnej T, tworzy skierowany graf acykliczny $\Delta^*(T)$. Wierzchołkami tego grafu są separowalne podtabele tabeli T. Podział tabeli na podtabele (podczas konstruowania grafu) zostaje zakończony, kiedy tabela jest zdegenerowana (wszystkie wiersze mają przypisaną tę samą decyzję). Na podstawie grafu $\Delta^*(T)$ można opisać zbiór reguł przypisanych do każdego wiersza tabeli T. W porównaniu do algorytmu prezentowanego w pracy [5], nie są rozważane wszystkie atry-

buty ze zbioru $E(T)$, lecz tylko jeden atrybut o minimalnej liczbie wartości i wszystkie jego wartości oraz dla pozostałych atrybutów uwzględniania jest tylko jedna, najczęstsza wartość. Przez $E^*(T,fi)$ jest oznaczany zbiór wartości atrybutu f_i. Jeśli f_i jest atrybutem o minimalnej liczbie wartości, wówczas $E^*(T,fi)$ jest zbiorem wszystkich wartości f_i w T. W przeciwnym przypadku $E^*(T,fi)$ zawiera tylko najczęstszą wartość f_i w T. Poniżej został przedstawiony pseudokod algorytmu, który konstruuje graf $\Delta^*(T)$.

```
Dane wejściowe: Tablica decyzyjna T.
Dane wyjściowe: Graf Δ*(T).

BEGIN
Skonstruuj graf zawierający pojedynczy wierzchołek T, który nie jest
oznaczony jako ``przetworzony'';
   WHILE wszystkie wierzchołki grafu nie są oznaczone jako ``przetworzone''
      DO
      Wybierz wierzchołek (tabelę) Θ, który nie został jeszcze ``prze-
      tworzony'';
         IF Θ jest zdegenerowana THEN
            badany wierzchołek jest oznaczany jako ``przetworzony'';
         END
         IF Θ nie jest zdegenerowana THEN
            Dla każdego atrybutu f_i∈E(Θ), rysowana jest wiązka krawędzi
            wychodzących z wierzchołka Θ.  Niech E*(Θ,fi)={b1,…,bt},
            wówczas rysowanych jest t krawędzi wychodzących z wierzchoł-
            ka Θ. Krawędzie te są oznaczane odpowiednio parami
            (fi,b1),…,(fi,bt). Krawędzie te wchodzą do wierzchołków
            Θ(fi,b1),…,Θ(fi,bt). Jeśli jakieś wierzchołki
            Θ(fi,b1),…,Θ(fi,bt) nie istnieją w grafie, wówczas są doda-
            wane do grafu. Każdemu wierszowi r tabeli Θ zostaje przypi-
            sany zbiór atrybutów E*(Θ,r)⊆ E(Θ). Wierzchołek Θ jest
            oznaczany jako ``przetworzony'';
         END
   END
Zwróć Graf Δ*(T);
END
```

W dalszej części rozdziału zostanie opisana procedura optymalizacji grafu $\Delta^*(T)$ względem pokrycia. Wynikiem pracy tej procedury jest graf G, który zawiera ten sam zbiór wierzchołków i krawędzi jak graf $\Delta^*(T)$. Różnicą jest to, że dowolny wiersz r, w każdym niekońcowym wierzchołku Θ w grafie G, ma przypisany zbiór atrybutów $E_G(\Theta,r) \subseteq E^*(\Theta,r)$. Jest także możliwe, że $G=\Delta^*(T)$.

Poniżej, dla każdego wierzchołka Θ w grafie G i dla każdego wiersza r z Θ zostanie opisany zbiór reguł decyzyjnych $Rul_G(\Theta,r)$. W tym celu należy przechodzić w grafie G, zaczynając od końcowych wierzchołków grafu, do wierzchołka T.

Niech Θ będzie końcowym wierzchołkiem w grafie G. Wówczas

$$Rul_G(\Theta,r) = \{\rightarrow d\}.$$

Niech teraz Θ będzie niekońcowym wierzchołkiem w grafie G takim, że dla każdego potomka Θ' wierzchołka Θ i dla każdego wiersza r' tabeli Θ', zbiór reguł $Rul_G(\Theta',r')$ został zdefiniowany. Niech $r=(b_1,\ldots,b_n)$ będzie wierszem tabeli Θ. Dla dowolnego atrybutu $f_i \in E_G(\Theta,r)$, jest definiowany zbiór reguł przypisanych do $Rul_G(\Theta,r,f_i)$:

$$Rul_G(\Theta,r,f_i) = \{f_i = b_i \wedge \alpha \rightarrow d : \alpha \rightarrow d \in Rul_G(\Theta(f_i,b_i),r)\}.$$

Wówczas

$$Rul_G(\Theta,r) = \cup_{fi \in EG(\Theta,r)} Rul_G(\Theta,r,f_i).$$

3.1. Procedura optymalizacji względem pokrycia

Poniżej zostanie opisana procedura optymalizacji względem pokrycia c. Niech $G=\Delta^*(T)$. W wyniku pracy procedury optymalizacji grafu G względem pokrycia każdy wiersz r każdego wierzchołka Θ w grafie G będzie miał przypisaną liczbę $Opt_G^c(\Theta,r)$ – maksymalne pokrycie reguły decyzyjnej ze zbioru $Rul_G(\Theta,r)$, i dla każdego wiersza r w każdym niekońcowym wierzchołku Θ grafu G zostanie zmieniony zbiór $E_G(\Theta,r)$ przypisany do r w Θ. Uzyskany graf jest oznaczany przez G^c.

Niech Θ będzie końcowym wierzchołkiem w grafie G. Wówczas do każdego wiersza r tabeli Θ jest przypisywana liczba $Opt_G^c(\Theta,r)$, która jest równa liczbie wierszy w Θ, które mają przypisaną decyzję d.

Niech Θ będzie niekońcowym wierzchołkiem w grafie G i wszyscy potomkowie Θ zostali przetworzeni. Niech $r=(b_1,\ldots,b_n)$ będzie wierszem tabeli Θ. Do wiersza r w tabeli Θ jest przypisywana liczba

$$Opt_G^c(\Theta,r) = \max\{Opt_G^c(\Theta(f_i,b_i),r): fi \in E_G(\Theta,r)\}$$

i zbiór $$E_G^c(\Theta,r) = \{fi \in E_G(\Theta,r), Opt_G^c(\Theta(f_i,b_i),r) = Opt_G^c(\Theta,r)\}.$$

Przykład procedury optymalizacji względem pokrycia dla przybliżonych reguł decyzyjnych został przedstawiony w pracy [13].

4. System reguł jako klasyfikator

Niech T będzie tablicą decyzyjną z atrybutami warunkowymi $f_1,\ldots,f_n$ i atrybutem decyzyjnym d. Niech S będzie *kompletnym systemem reguł*, czyli takim, w którym każda reguła jest prawdziwa dla T i dla każdego wiersza z T istnieje reguła, która jest dla niego zrealizowana. System S może zostać użyty jako klasyfikator, w celu predykcji wartości atrybutu decyzyjnego d dla n-tki $a=(a_1,\ldots a_n)$ wartości atrybutów warunkowych $f_1,\ldots,f_n$, w przypadku gdy a nie jest wierszem z T.

Jeżeli S nie zawiera reguł, które są zrealizowane dla a, wówczas wartość atrybutu d dla a jest równa najbardziej wspólnej decyzji dla T.

Niech S zawiera reguły, które są zrealizowane dla a i p niech będzie minimalną wartością d taką, że liczba reguł z S, które są zrealizowane dla a i mają p z prawej strony jest maksymalna. Wówczas wartość d dla n-tki a jest równa p.

Kompletny system dokładnych reguł decyzyjnych stosowany jako klasyfikator może być przeuczony, tzn. może mieć dobrą jakość klasyfikacji dla wierszy z T i niską jakość klasyfikacji dla n-tek, które nie są wierszami w T. W celu poprawy tej sytuacji można zastosować procedurę przycinania reguł decyzyjnych przedstawioną w poniższym rozdziale.

4.1. Procedura przycinania dokładnych reguł decyzyjnych

Niech β będzie liczbą rzeczywistą taką, że $0 \leq \beta < 1$. Przez $P(T)$ jest oznaczana liczba nieuporządkowanych par wierszy o różnych decyzjach z T.

Niech niezdegenerowana tablica decyzyjna T będzie podzielona na trzy niezdegenerowane podtabele $T1$, $T2$, $T3$. Opisany w poprzednim rozdziale zmodyfikowany algorytm dynamicznego programowania stosowany jest dla uzyskania kompletnego systemu reguł decyzyjnych S dla $T1$.

Rozważana jest dowolna reguła z S

$$f_{i1} = a_1 \wedge \ldots \wedge f_{im} = a_m \rightarrow t. \tag{2}$$

Dla $j \in \{1,\ldots,m\}$, badana jest *podreguła*

$$f_{ij} = aj \wedge \ldots \wedge f_{im} = a_m \rightarrow t' \tag{3}$$

reguły (2), gdzie t' jest najbardziej wspólną decyzją dla podtabeli $T1(f_{ij},a_j)\ldots(f_{im},a_m)$ i można wyznaczyć niedokładność $P(T1(f_{ij},a_j)\ldots(f_{im},a_m))/P(T1)$ podreguły (3) względem $T1$.

Niech $\beta_1 < \beta_2 < \ldots < \beta q$ będą wszystkimi różnymi niedokładnościami dla wszystkich podreguł reguł z systemu S. Dla każdego $k \in \{1,\ldots,q\}$, konstruowany jest system reguł decyzyjnych S_k. Dla dowolnej reguły (2) z S, podreguła (3) reguły (2) z maksymalnym $j \in \{1,\ldots,m\}$, dla którego niedokładność wynosi najwyżej β_k, jest dodawana do S_k.

Dla każdego $k \in \{1,\ldots,q\}$, system reguł decyzyjnych S_k jest stosowany jako klasyfikator dla tabeli $T2$ i obliczana jest liczba błędnych zaklasyfikowań. Jest to liczba wierszy w $T2$, dla których decyzja wyznaczona przez system S_k nie jest równa decyzji dołączonej do rozważanego wiersza. Wybierane jest minimalne $k_0 \in \{1,\ldots,q\}$, dla którego system reguł S_{k0} posiada minimalną liczbę błędnych zaklasyfikowań. System ten S_{k0} jest traktowany jako ostateczny klasyfikator. Jest on stosowany do tabeli $T3$ i obliczana jest jego jakość klasyfikacji (liczba błędnych zaklasyfikowań dla wierszy z tabeli $T3$).

5. Wyniki eksperymentów

Celem przeprowadzonych eksperymentów było porównanie wyników klasyfikatora regułowego utworzonego na podstawie zmodyfikowanego algorytmu dynamicznego programowania z wynikami klasyfikatora utworzonego za pomocą „klasycznego" algorytmu dynamicznego programowania. W procesie konstrukcji klasyfikatora została zastosowana przedstawiona w poprzednim rozdziale procedura przycinania reguł decyzyjnych.

Przedstawione w pracy [13] wyniki pokazały, że zmodyfikowany algorytm pozwala uzyskać wartości pokrycia reguł decyzyjnych bliskie wartościom optymalnym, co jest istotne z punktu widzenia reprezentacji wiedzy. Wyniki te dotyczyły przybliżonych reguł decyzyjnych. Niniejsza praca dotyczy optymalizacji dokładnych reguł decyzyjnych względem pokrycia i klasyfikacji danych.

Eksperymenty zostały przeprowadzone na tablicach decyzyjnych z Repozytorium Uczenia Maszynowego [2]. Jeśli tablica zawierała atrybuty warunkowe, które posiadały unikalną wartość dla każdego wiersza, to atrybuty takie zostały usunięte. Jeśli w tablicach występowały identyczne wiersze o różnych decyzjach, to grupa takich wierszy została zastąpiona jednym wierszem o najbardziej wspólnej decyzji dla grupy. Jeśli w tablicach występowały brakujące wartości, to każda taka wartość została zastąpiona najbardziej wspólną wartością dla danego atrybutu.

Tabela 1 przedstawia średni błąd klasyfikacji dla 2-krotnej walidacji krzyżowej (dla każdej tablicy decyzyjnej eksperymenty zostały powtórzone 40 razy). Każda tablica decyzyjna została losowo podzielona na trzy części: trenująca-30%, walidacyjna-20% i testowa-50%.

Klasyfikator został skonstruowany na podstawie części trenującej. Następnie, reguły zostały przycięte, uwzględniając minimalny błąd klasyfikacji dla zbioru walidacyjnego. Dla części trenującej zbioru danych konstruowane były dokładne reguły decyzyjne. Następnie reguły te były przycinane i uzyskano β-reguły decyzyjne dla rosnących wartości β. Została wybrana taka wartość β, dla której uzyskano minimalny błąd klasyfikacji na zbiorze walidacyjnym. Uzyskany zbiór reguł został zastosowany jako klasyfikator dla części testowej i został wyznaczony błąd klasyfikacji. Jest to liczba wierszy z części testowej tablicy decyzyjnej, które nie zostały poprawnie sklasyfikowane, podzielona przez liczbę wszystkich wierszy z części testowej tablicy decyzyjnej. Kolumny Mdp i Std-Mdp przedstawiają odpowiednio błąd klasyfikacji oraz odchylenie standardowe dla zmodyfikowanego algorytmu dynamicznego programowania. Kolumny Dp i Std-Dp przedstawiają, odpowiednio, błąd klasyfikacji oraz odchylenie standardowe dla „klasycznego" algorytmu dynamicznego programowania. Ostatni wiersz tabeli 1 przedstawia średni błąd klasyfikacji dla badanych tablic decyzyjnych.

Tabela 1

Średni błąd klasyfikacji

Tablica decyzyjna	Wiersze	Attr	Mdp	Std- Mdp	Dp	Std-Dp
Balance-scale	625	4	0,23	0,03	0,23	0,04
Breast-cancer	266	9	0,28	0,06	0,31	0,08
Cars	1728	6	0,13	0,02	0,17	0,02
Hayes-roth-data	69	5	0,44	0,11	0,35	0,10
Soybean-small	47	35	0,07	0,05	0,06	0,04
			0,23	0,05	0,23	0,06

Wyniki zawarte w tabeli 1 pokazują, że z punktu widzenia klasyfikacji zmodyfikowany algorytm jest porównywalny z „klasycznym" algorytmem dynamicznego programowania.

Tabela 2 zawiera wyniki, dotyczące pokrycia dokładnych reguł decyzyjnych. Kolumny Mdp oraz Dp przedstawiają średnią wartość pokrycia reguł konstruowanych odpowiednio przez zmodyfikowany oraz "klasyczny" algorytm dynamicznego programowania.

Tabela 2

Średnie pokrycie reguł decyzyjnych

Tablica decyzyjna	Mdp	Dp
Balance-scale	3,07	4,21
Breast-cancer	6,15	9,53
Cars	325,58	332,76
Hayes-roth-data	2,32	6,52
Soybean-small	12,53	12,53

Tabela 3 przedstawia porównanie liczby wierzchołków i krawędzi w skierowanym grafie acyklicznym. Kolumny CompW i CompK przedstawiają, odpowiednio, liczbę wierzchołków/krawędzi w grafie skonstruowanym przez „klasyczny" algorytm dynamicznego programowania podzieloną przez liczbę wierzchołków/krawędzi w grafie skonstruowanym przez zmodyfikowany algorytm dynamicznego programowania.

Tabela 3

Skierowany graf acykliczny

Tablica decyzyjna	Wiersze	Attr	CompW	CompK
Balance-scale	625	4	1,85	4,23
Breast-cancer	266	9	2,42	6,55
Cars	1728	6	8,77	17,55
Hayes-roth-data	69	5	1,69	2,66
Soybean-small	47	35	1,19	2,69

Wyniki badań zawarte w tabelach 2 i 3 pokazują, iż zazwyczaj, dla dokładnych reguł decyzyjnych, zmodyfikowany algorytm dynamicznego programowania pozwala konstruować reguły o dobrym pokryciu. Natomiast rozmiar skierowanego grafu acyklicznego jest mniejszy (w przypadku zbioru Cars liczba krawędzi zmniejszyła się ponad 17 razy), co ma znaczenie z punktu widzenia skalowalności proponowanego podejścia.

6. Podsumowanie

W pracy została przedstawiona budowa klasyfikatora regułowego z wykorzystaniem procedury przycinania reguł decyzyjnych. Dokładne reguły decyzyjne zostały skonstruowane za pomocą zmodyfikowanego algorytmu dynamicznego programowania. Uzyskane wyniki dotyczące klasyfikacji są porównywalne z wynikami dla klasyfikatora regułowego konstruowanego z wykorzystaniem klasycznego algorytmu dynamicznego programowania.

W pracy [13] zostało pokazane, iż zaproponowany algorytm pozwala konstruować przybliżone reguły decyzyjne o wartościach pokrycia bliskich wartościom optymalnym. Wnioski te zostały potwierdzone także dla dokładnych reguł decyzyjnych, co jest istotne z punktu widzenia reprezentacji wiedzy. Kolejnym krokiem badań będzie utworzenie klasyfikatora regułowego opartego na heurystykach zachłannych.

BIBLIOGRAFIA

1. Ann A., Cercone N.: Rule quality measures improve the accuracy of rule induction: an experimental approach. [w:] Raś Z. W., Ohsuga S. (red.): ISMIS 2000, LNCS, Vol. 1932, Springer, 2000, s. 119÷129.
2. Asuncion A., Newman D.J.: UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2007, http://www.ics.uci.edu/~mlearn/ (dostęp grudzień 2016).
3. Bazan J., Szczuka M.: The rough set exploration system. [w:] Peters J. F., Skowron A. (red.): Transactions on Rough Sets III, LNCS, Vol. 3400, Springer, 2005, s. 37÷56.
4. Malara W., Sikora M., Wróbel Ł.: Program do indukcji i oceny reguł klasyfikacyjnych, zintegrowany z pakietem R. Studia Informatica, Vol. 34, No. 2B, Wydawnictwo Politechniki Śląskiej, Gliwice 2013, s. 339÷352.
5. Moshkov M., Zielosko B.: Combinatorial Machine Learning – A Rough Set Approach. Studies in Computational Intelligence, Vol. 360, Springer, Heidelberg 2011.
6. Nowak-Brzezińska A.: KBEXPLORATOR a inne narzędzia eksploracji regułowych baz wiedzy. Studia Informatica, Vol. 36, No. 1, Wydawnictwo Politechniki Śląskiej, Gliwice 2015, s. 75÷88.
7. Pawlak Z., Skowron A.: Rough sets and Boolean reasoning. Information Science, Vol. 177, 2007, s. 41÷73.

8. Sikora M., Wróbel L.: Data-driven adaptive selection of rule quality measures for improving rule induction and filtration algorithms. Int. J. General Systems, Vol. 42(6), 2013, s. 594÷613.
9. Simiński R.: Biblioteka KBExpertLib dla języka Java – właściwości funkcjonalne i badania wydajnościowe. Studia Informatica, Vol. 37, No. 1, Wydawnictwo Politechniki Śląskiej, Gliwice 2016, s. 125÷134.
10. Stańczyk U.: Selection of decision rules based on attribute ranking. Journal of Intelligent and Fuzzy Systems, Vol. 29(2), 2015, s. 899÷915.
11. Stefanowski J., Vanderpooten D.: Induction of decision rules in classification and discovery-oriented perspectives. Int. J. Intell. Syst., Vol. 16(1), 2001, s. 13÷27.
12. Zielosko B., Moshkov M.: Approximate algorithm for β-decision rule optimization. Studia Informatica, Vol. 32, No. 2A(96), Wydawnictwo Politechniki Śląskiej, Gliwice 2011, s. 329÷335.
13. Zielosko B.: Optimization of Approximate Decision Rules Relative to Coverage. BDAS 2014, CCIS, Vol. 424, Springer, 2014, s. 170÷179.

Abstract

The article is a continuation of research connected with a modified dynamic programming algorithm for optimization of decision rules relative to coverage [13]. The idea of construction of classifier using pruning of decision rules was presented. The paper contains experimental results for rule based classifiers using modified dynamic programming algorithm and "standard" dynamic programming algorithm. The results are comparable. Presented algorithm allows us to obtain exact decision rules with good enough coverage, what is important from the point of view of knowledge representation. The size of the constructed directed acyclic graph is smaller than the size of the graph constructed by the "standard" dynamic programming algorithm. This fact is important from the point of view of scalability.

Adresy

Beata ZIELOSKO: Uniwersytet Śląski w Katowicach, Instytut Informatyki, ul. Będzińska 39, 41-200 Sosnowiec, Polska, beata.zielosko@us.edu.pl.
Krzysztof Żabiński: Uniwersytet Śląski w Katowicach, Instytut Informatyki, ul. Będzińska 39, 41-200 Sosnowiec, Polska, krzysztof.kamil.zabinski@gmail.com.