PRACE NAUKOWE UNIWERSYTETU EKONOMICZNEGO WE WROCŁAWIU
RESEARCH PAPERS OF WROCŁAW UNIVERSITY OF ECONOMICS nr 507 • 2018

Taksonomia 30
Klasyfikacja i analiza danych – teoria i zastosowania

ISSN 1899-3192
e-ISSN 2392-0041

Mariusz Kubus
Politechnika Opolska
e-mail: m.kubus@po.opole.pl

ELIMINACJA ZMIENNYCH Z WYKORZYSTANIEM MARGINESU

FEATURE SELECTION WITH A USE OF MARGIN

DOI: 10.15611/pn.2018.507.12
JEL Classification: C01, C14, C52

Streszczenie: Ważnym etapem budowy klasyfikatora jest dobór zmiennych. W metodzie *k* najbliższych sąsiadów, wrażliwej na zmienne nieistotne, etap ten jest niezbędny do uzyskania większej dokładności klasyfikacji. Metody doboru zmiennych, które także wykorzystują najbliższe sąsiedztwo, dokonują lokalnej oceny mocy dyskryminacyjnej zmiennych i zarazem reprezentują podejście wielowymiarowe. Część z nich wykorzystuje pojęcie marginesu (*margin*), definiując za jego pomocą funkcję celu i formułując zadanie ważenia zmiennych jako zadanie optymalizacji. W artykule porównano trzy algorytmy z tej grupy metod ze względu na zdolność identyfikacji zmiennych nieistotnych, dokładność klasyfikacji oraz czas pracy. Zweryfikowano też dwie własne propozycje modyfikacji. W badaniach wykorzystano zbiory danych rzeczywistych z dołączonymi zmiennymi nieistotnymi, które reprezentowały różne rozkłady, niezależne od klas.

Słowa kluczowe: dobór zmiennych, najbliższe sąsiedztwo, margines.

Summary: Feature selection methods employing nearest neighborhood utilize a local assessment of the discriminative power of variables, and at the same time they represent a multidimensional approach. Some of them use the concept of margin, defining an objective function, and formulating the feature weighing task as an optimization problem. In this paper, three algorithms from this group of methods have been compared due to: the ability of identifying the irrelevant variables, classification accuracy, and computational time. Two modifications have also been verified. The real data sets with irrelevant variables from different distributions have been used in the experiments.

Keywords: feature selection, nearest neigbourhood, margin.

1. Wstęp

W dobie powstawania dużych baz danych wzrasta popularność metod eksploracyjnej analizy danych. Przetwarzanie takich zbiorów wymaga najczęściej etapu czyszczenia danych, co określić można jako ich przygotowanie do postaci wejściowej dla

wybranej metody analitycznej. W zadaniu dyskryminacji jest to zbiór uczący, a więc zbiór wielowymiarowych obserwacji w przestrzeni cech, ze znaną etykietą klasy. W sytuacji gdy celem analizy jest wydobycie wiedzy z danych, a zestaw zmiennych nie został wybrany w oparciu o wiedzę merytoryczną o badanym zjawisku, ważnym etapem jest dobór zmiennych do modelu. Zmienne, które nie mają wpływu na zmienną objaśnianą, nazywane dalej zmiennymi nieistotnymi, nie tylko zbędnie zwiększają wymiar przestrzeni cech (zwiększając wymogi co do wielkości próby), ale mogą obniżyć dokładność klasyfikacji. Jedną ze szczególnie wrażliwych metod dyskryminacji na zmienne nieistotne jest metoda *k* najbliższych sąsiadów kNN.

Metody selekcji zmiennych są obecnie dzielone na trzy podejścia [Guyon i in. 2006]. Dobór przed etapem uczenia i od niego niezależny (*filters*). Dobór zmiennych przez selekcję modeli (*wrappers*). To podejście ma ścisły związek z modelem i wymaga zastosowania jakiejś techniki przeszukiwania przestrzeni wszystkich podzbiorów zmiennych. Wreszcie wyróżnić można metody dyskryminacji, gdzie dobór zmiennych jest integralną częścią algorytmu uczącego (*embedded methods*). W problemach wysokowymiarowych, np. klasyfikacja genów czy tekstów, dużą popularnością cieszy się pierwsza grupa metod. Dobór dokonywany jest na podstawie kryteriów, które nie są związane z modelem. Można je podzielić ze względu na podejście jedno- lub wielowymiarowe oraz ze względu na globalny lub lokalny charakter oceny zmiennych. Najczęściej stosowana jest ocena globalna za pomocą statystyk testowych lub miar bazujących na entropii. Z kolei kryteria wielowymiarowe skonstruowane są zwykle tak, by maksymalizowały skorelowanie ze zmienną objaśnianą i jednocześnie minimalizowały skorelowanie predyktorów między sobą. Do najbardziej znanych należą pojemność informacji Hellwiga [1969] oraz korelacja grupowa, którą wykorzystał M. Hall [2000] w swym algorytmie CFS. Kryteria te oceniają podzbiory zmiennych, więc w praktyce stosowane są z heurystycznym przeszukiwaniem. Można zauważyć, że ich konstrukcja to agregacja wartości kryteriów jednowymiarowych. Zatem zmienne nieistotne indywidualnie, które są istotne w kontekście innych zmiennych (zob. [Kubus 2015]), nie będą przez nie właściwie ocenione. Zupełnie odmienny charakter mają wielowymiarowe kryteria wykorzystujące najbliższe sąsiedztwo. Ze swej natury wykorzystują lokalną ocenę zmiennych, co może być korzystne też w sytuacjach, gdy ważność zmiennych nie jest jednakowa w różnych regionach przestrzeni cech.

R. Gilad-Bachrach, A. Navot i N. Tishby [2004] sformułowali pojęcie marginesu (*margin*), definiując za jego pomocą funkcję celu i formułując zadanie ważenia zmiennych jako zadanie optymalizacji. W artykule dokonana będzie ocena konkurencyjności tych metod w stosunku do pionierskiego algorytmu oceny ważności zmiennych za pomocą najbliższego sąsiedztwa, jakim jest Relief [Kira, Rendell 1992; Kononenko 1994]. Pokażemy prostą modyfikację algorytmu Simba [Gilad-Bachrach i in. 2004]. Podjęta też będzie próba modyfikacji marginesu o komponent kary. Badania empiryczne przeprowadzone będą na danych rzeczywistych z dodatkowo dołączonymi zmiennymi nieistotnymi, które będą reprezentowały różne rozkłady, niezależne od klas.

2. Najbliższe sąsiedztwo i margines w problemie doboru zmiennych

Pionierską pracą wykorzystującą najbliższe sąsiedztwo w problemie doboru zmiennych do modelu dyskryminacyjnego był artykuł K. Kiry i L.A. Rendella [1992]. Opracowali oni algorytm Relief, który iteracyjnie aktualizuje wagi zmiennych na podstawie odległości od najbliższych sąsiadów losowo wybranego obiektu $\boldsymbol{x}$ ze zbioru uczącego U. I. Kononenko [1994] wprowadził modyfikację na przypadek dyskryminacji wielu klas oraz zaproponował uwzględniać k najbliższych sąsiadów. Swoją wersję nazwał ReliefF. Oznaczmy przez $NH_i(\boldsymbol{x})$ i-tego najbliższego sąsiada z tej samej klasy co obiekt $\boldsymbol{x}$ (*nearest hit*) oraz przez $NM_i^c(\boldsymbol{x})$ i-tego najbliższego sąsiada z innej klasy c (*nearest miss*). W tym przypadku wagi zmiennych X_j aktualizowane są wg formuły:

$$w_j \leftarrow w_j - \frac{1}{k}\sum_{i=1}^{k}\left|x_j - NH_i(\boldsymbol{x})_j\right| + \sum_{c\neq Y(\boldsymbol{x})}\left[\frac{P(c)}{1-P(Y(\boldsymbol{x}))}\cdot\frac{1}{k}\sum_{i=1}^{k}\left|x_j - NM_i^c(\boldsymbol{x})_j\right|\right], \quad (1)$$

gdzie $Y(\boldsymbol{x})$ oznacza klasę obiektu $\boldsymbol{x}$. Algorytm ReliefF ma dwa parametry: liczbę iteracji oraz liczbę najbliższych sąsiadów. Ich wybór rekomendowany jest w pracy [Kubus 2016]. Należy podkreślić, że algorytm nadaje wagi zmiennym, w wyniku czego otrzymujemy ranking. Otwartą kwestią pozostaje zadanie wyznaczenia wartości progowej, która ostatecznie decyduje, ile zmiennych zostawiamy do dalszej analizy. Można wyodrębnić trzy główne podejścia do rozwiązania takiego problemu. Pierwsze polega na arbitralnym ustaleniu wartości progowej. Jest to rozwiązanie najprostsze i nie niesie z sobą kosztownych obliczeń, ale rzadko optymalne. Drugie to analiza wykresu osypiska. W przypadku gdy nie ma wyraźnego uskoku, podejście to sprowadza się do wyboru arbitralnego. Ponadto nie jest wygodne w ocenie dokładności klasyfikacji uzyskanego modelu przez sprawdzanie krzyżowe czy też wielokrotny podział na próbę uczącą oraz testową. Wreszcie trzecie podejście wykorzystuje metodologię *wrappers*, tzn. selekcję zmiennych za pomocą selekcji modeli. Budowane są modele zagnieżdżone i ten, który uzyska najlepszą ocenę, wyznacza optymalny podzbiór zmiennych. Funkcję oceny może pełnić błąd klasyfikacji estymowany sprawdzaniem krzyżowym lub z wykorzystaniem zbioru walidacyjnego. W niniejszym opracowaniu zastosowano właśnie to podejście, gdzie wykorzystano klasyfikatory kNN. Ponadto dla skrócenia czasu obliczeń sprawdzano tylko 10 modeli zagnieżdżonych budowanych na podzbiorach zmiennych wyznaczonych przez 10 wybranych wartości progowych.

Nowe możliwości metodologiczne stwarza zdefiniowanie marginesu. Margines dla obiektu $\boldsymbol{x}\in U$ w podprzestrzeni cech $S\subset X=(X_1,...,X_p)$ określa się następująco:

$$D(\boldsymbol{x},S)=\frac{1}{2}\cdot\left(\left\|\boldsymbol{x}-NM(\boldsymbol{x})\right\|_{\boldsymbol{w}}-\left\|\boldsymbol{x}-NH(\boldsymbol{x})\right\|_{\boldsymbol{w}}\right), \quad (2)$$

gdzie *NM* oraz *NH* oznaczają najbliższego sąsiada innej lub tej samej klasy odpowiednio (*nearest miss*, *nearest hit*). Kryterium oceniające podzbiór zmiennych *S* (zależne od wektora wag zmiennych $\boldsymbol{w}$) można sformułować następująco:

$$F(S,\boldsymbol{w}) = \sum_{\boldsymbol{x}\in U} f(D(\boldsymbol{x},S)), \tag{3}$$

gdzie $f(D)$ jest różniczkowalną funkcją rosnącą, np. sigmoidalną. Przyjmując jednakowe wagi, wartość kryterium F zależy jedynie od podzbioru zmiennych S. Maksymalizacja F jest wtedy zadaniem optymalizacji kombinatorycznej. R. Gilad-Bachrach i in. [2004] zaproponowali w algorytmie G-flip następującą strategię przeszukiwania (tab. 1). W artykule proponujemy modyfikację kryterium (3) polegającą na wprowadzeniu komponentu kary za złożoność modelu:

$$F(S,\mathbf{w}) = F(S) = \sum_{\mathbf{x}\in U} f(D(\mathbf{x},S)) - card(S)\cdot \ln N, \tag{4}$$

gdzie N jest liczbą obiektów w zbiorze uczącym U.

Tabela 1. Algorytm G-flip

1. Ustal początkowo zbiór numerów zmiennych objaśniających $n(S)$ jako zbiór pusty.
2. Powtarzaj następujące kroki ustaloną liczbę razy.
   a) wyznacz permutację A numerów zmiennych $\{1,2,\ldots,p\}$,
   b) dla kolejnych i ze zbioru $\{1,2,\ldots,p\}$ oblicz $F_1 = F(n(S)\cup A(i))$ oraz $F_2 = F(n(S)\setminus A(i))$,
   c) jeśli $F_1 > F_2$, to aktualizuj zbiór $n(S) \leftarrow n(S)\cup\{A(i)\}$, w przeciwnym razie $n(S) \leftarrow n(S)\setminus\{A(i)\}$
   d) jeśli nie nastąpiła zmiana w zawartości zbioru $n(S)$, zakończ działanie algorytmu.

Źródło: opracowanie własne na podstawie [Gilad-Bachrach i in. 2004].

Z kolei w przypadku, gdy $S = \boldsymbol{X}$, wartość kryterium (3) zależy jedynie od wag. Zadanie maksymalizacji można wtedy rozwiązać, stosując gradientową zasadę największego wzrostu. Znalazła ona wyraz w opracowanym przez autorów algorytmie Simba (*Iterative Search Margin Based Algorithm*), różniącym się od algorytmu Relief jedynie formułą aktualizacyjną, która przyjmuje postać:

$$w_j \leftarrow w_j + \frac{1}{2}\frac{\partial f(D)}{\partial D}\left(\frac{\left(x_j - NM(\boldsymbol{x})_j\right)^2}{\left\|\boldsymbol{x} - NM(\boldsymbol{x})\right\|_{\boldsymbol{w}}} - \frac{\left(x_j - NH(\boldsymbol{x})_j\right)^2}{\left\|\boldsymbol{x} - NH(\boldsymbol{x})\right\|_{\boldsymbol{w}}}\right) w_j . \tag{5}$$

Prostą modyfikacją tego algorytmu (analogiczną do ReliefF) jest wyszukanie k najbliższych sąsiadów zamiast jednego i uśrednianie odległości.

3. Badania empiryczne

Celem badań empirycznych było porównanie opisanych algorytmów ze względu na zdolność identyfikacji zmiennych nieistotnych, dokładność klasyfikacji oraz czas pracy. Błędy klasyfikacji estymowano jako średnie błędy na próbach testowych przy 30 losowych podziałach zbioru danych w stosunku 1/3. Badanie przeprowadzono, wykorzystując dane rzeczywiste (repozytorium Uniwersytetu Kalifornijskiego [Lichman 2013]) i dołączając do nich zmienne nieistotne. Wybrano zbiory: *cardiotocographic* (2126, 21, 3), *ionosphere* (351, 33, 2), *parkinson* (195, 22, 2) oraz *sonar* (208, 60, 2). W nawiasach podano kolejno liczby: obiektów, zmiennych objaśniających oraz klas. Zmienne nieistotne generowano z rozkładów jednowymiarowych, jednakowych w klasach. Dla poszerzenia zakresu badania wybrano rozkład symetryczny, niesymetryczny oraz dwumodalny, mianowicie:

- standaryzowany rozkład normalny,
- rozkład wykładniczy z parametrem $\lambda = 2$,
- mieszanka rozkładów normalnych: $N(0;1)$ (1/3 obserwacji) oraz $N(6;0,5)$ (2/3 obserwacji).

Rozważano też trzy przypadki ze względu na liczbę zmiennych nieistotnych. W pierwszym do zbiorów wprowadzano około 10%p takich zmiennych, a w kolejnych odpowiednio połowę z liczby oryginalnych zmiennych p lub dwukrotnie więcej.

Modyfikacja algorytmu Simba polegająca na uwzględnieniu k najbliższych sąsiadów okazała się korzystna. Dla uzyskania porównywalności z algorytmem ReliefF parametr ten był wyznaczany identycznie, tzn. jako przybliżenie liczby $N^{1/4}$ [Kubus 2016]. Poprawiona wersja algorytmu wprowadzała mniej zmiennych nieistotnych przy niemal identycznym czasie pracy. Skutkowało to zmniejszeniem błędów klasyfikacji, co ilustruje tab. 2. W przypadku zmiennych nieistotnych o innych rozkładach uzyskano podobne wyniki.

Tabela 2. Porównanie błędów klasyfikacji w % (w nawiasie błędy standardowe) oryginalnego algorytmu Simba oraz modyfikacji Simba_k. Zmienne nieistotne generowano z rozkładu wykładniczego

Zbiory	*cardiotocographic*			*ionosphere*			*parkinson*			*sonar*		
Liczba nieistotnych	*p/10*	*p/2*	*2p*	*p/10*	*p/2*	*2p*	*p/10*	*p/2*	*2p*	*p/10*	*p/2*	*2p*
Simba	8,9 (0,2)	9,2 (0,3)	9,5 (0,3)	12,1 (0,5)	12,5 (0,6)	15,9 (0,7)	7,7 (0,8)	12,5 (0,9)	21,1 (1,2)	16,5 (0,8)	19,4 (1,0)	30,3 (1,1)
Simba_k	8,9 (0,2)	8,7 (0,2)	8,8 (0,3)	11,5 (0,5)	11,8 (0,6)	12,6 (0,6)	7,5 (0,6)	11,8 (1,0)	17,9 (1,0)	15,1 (0,8)	18,7 (1,1)	30,1 (1,4)

Źródło: obliczenia własne.

Tabele 3 i 4 przedstawiają liczby wprowadzanych zmiennych oraz czasy pracy algorytmów. Obliczenia przeprowadzono na komputerze z procesorem 2,1 GHz oraz 4,0 GB RAM. Algorytm G-flip dobrze identyfikował zmienne nieistotne, ale był

Tabela 3. Liczby zmiennych wprowadzanych do modelu: nieistotnych (*n*) oraz oryginalnych (*q*). Rozważono trzy rozkłady zmiennych nieistotnych oraz różne ich liczby *I* (jako % liczby oryginalnych zmiennych). Porównano trzy algorytmy doboru zmiennych: G-flip, Simba z *k* najbliższymi sąsiadami oraz ReliefF

Zbiory	*I*	G-flip						Simba_k						ReliefF					
		norm		*exp*		*mix*		*norm*		*exp*		*mix*		*norm*		*exp*		*mix*	
		n	*q*	*n*	*q*	*n*	*q*	*n*	*q*	*n*	*q*	*n*	*q*	*n*	*q*	*n*	*q*	*n*	*q*
ctg	*10%*	0	20	0	20	0	20	0	9	0	5	0	12	0	12	0	12	0	11
	50%	0	20	0	20	0	20	0	19	0	12	0	13	0	18	0	13	0	16
	200%	0	20	0	20	0	20	0	10	0	4	0	7	0	14	0	14	0	16
ionosphere	*10%*	0	33	0	33	1	32	0	4	0	10	1	32	0	33	0	32	0	32
	50%	2	33	1	33	3	32	0	7	0	20	1	16	0	33	1	33	0	28
	200%	2	33	5	32	3	33	0	5	1	11	1	20	0	33	0	32	1	32
parkinson	*10%*	0	21	0	21	0	21	0	11	0	21	0	15	0	15	0	10	0	11
	50%	0	21	0	21	0	21	0	22	3	19	0	5	0	19	0	9	0	17
	200%	0	21	0	21	0	21	3	17	5	11	0	10	0	14	0	18	0	13
sonar	*10%*	0	43	0	43	0	43	0	20	0	43	0	18	0	31	0	21	1	40
	50%	0	43	0	43	0	43	3	43	6	40	2	44	1	40	0	32	0	31
	200%	0	43	0	43	0	43	17	26	40	34	33	43	16	38	5	38	0	23

Źródło: obliczenia własne.

Tabela 4. Porównanie czasu pracy algorytmów w sek.

Zbiory	*I*	G-flip	Simba_k	ReliefF
ctg	*10%*	6 323	26	2,3
	50%	7 830	34	2,7
	200%	22 743	73	4,0
ionosphere	*10%*	124	4	0,7
	50%	207	5	0,8
	200%	390	9	0,8
parkinson	*10%*	31	2	0,5
	50%	46	2	0,6
	200%	186	4	0,6
sonar	*10%*	92	4	0,7
	50%	177	5	0,7
	200%	336	10	0,9

Źródło: obliczenia własne.

dosyć wolny. Czas pracy wzrastał szybko z liczbą zmiennych. Gdyby chcieć estymować błąd klasyfikacji przez 10-częściowe sprawdzanie krzyżowe, cała procedura wymagałaby tyle czasu, że eliminowałoby ją to z wielu zastosowań. Wprowadzenie regularyzacji w algorytmie G-flip (4) nie zapewniło usunięcia zmiennych nieistotnych w zbiorze *ionosphere*. W przypadku gdy zmienne nieistotne były z mieszanki rozkładów, algorytm wprowadzał od 1 do 3 takich zmiennych. Dla zmiennych nieistotnych o rozkładzie wykładniczym G-flip wprowadzał jedną taką zmienną, a w przypadku rozkładu normalnego także jedną, ale tylko wtedy, gdy ich liczba była duża, tzn. dwukrotnie przewyższała liczbę zmiennych oryginalnych.

Tabela 5 przedstawia błędy klasyfikacji klasyfikatora kNN, który stosowano po selekcji zmiennych. Liczbę sąsiadów wyznaczano wg sugestii G.G. Enasa i S.C. Choi [1986], to jest $k_N \approx N^{1/4}$. Ze względu na duży czas pracy algorytmu G-flip studium porównawcze ograniczono do algorytmów Simba_k oraz ReliefF stosowanych z k najbliższymi sąsiadami.

Tabela 5. Błędy klasyfikacji w % (w nawiasach błędy standardowe). Do oryginalnych zbiorów danych dodawano zmienne nieistotne (10%*p*, 50%*p* lub 200%*p*) z różnych rozkładów

Zbiory	*I*	Simba_k			ReliefF		
		norm	*exp*	*mix*	*norm*	*exp*	*mix*
ctg	*10%*	8,6 (0,2)	8,9 (0,2)	8,4 (0,2)	9 (0,2)	8,9 (0,2)	8,7 (0,2)
	50%	9 (0,2)	8,7 (0,2)	8,5 (0,2)	9 (0,2)	9,3 (0,2)	9,3 (0,2)
	200%	9,2 (0,3)	8,8 (0,3)	8,9 (0,3)	9,2 (0,2)	9,3 (0,2)	9,1 (0,2)
ionosphere	*10%*	11,8 (0,5)	11,5 (0,5)	11,9 (0,5)	12,6 (0,5)	12,8 (0,5)	12,9 (0,5)
	50%	12 (0,5)	11,8 (0,6)	12,4 (0,6)	13,1 (0,5)	13,2 (0,5)	13 (0,5)
	200%	12,4 (0,6)	12,6 (0,6)	13,9 (0,7)	13,3 (0,5)	14,1 (0,6)	13,4 (0,6)
parkinson	*10%*	6,9 (0,5)	7,5 (0,6)	7,9 (0,7)	7,1 (0,7)	7 (0,6)	6,8 (0,6)
	50%	12,4 (1)	11,8 (1)	12,8 (0,9)	8,3 (0,7)	9,3 (0,6)	8,8 (0,7)
	200%	19,4 (0,9)	17,9 (1)	17,9 (1,1)	11,6 (1)	12,7 (1)	13 (1,2)
sonar	*10%*	14,8 (0,7)	15,1 (0,8)	14,8 (0,9)	16,7 (0,8)	15,5 (0,7)	14,3 (0,8)
	50%	19,3 (0,8)	18,7 (1,1)	20,5 (1,1)	17,2 (0,9)	18,6 (1)	18,6 (0,8)
	200%	25,6 (1)	30,1 (1,4)	28 (0,9)	21,6 (0,9)	21,4 (1)	22,4 (1)

Źródło: obliczenia własne.

W przypadku zmiennych nieistotnych o rozkładzie normalnym algorytm ReliefF na ogół zostawiał więcej zmiennych, jednak nieco lepiej identyfikował zmienne nieistotne. W przypadkach gdy algorytm Simba_k nie rozpoznawał wszystkich zmiennych nieistotnych, ReliefF prowadził do mniejszych błędów klasyfikacji. W pozostałych przypadkach dokładniejsze klasyfikacje otrzymywano raz algorytmem Simba_k, raz ReliefF. Różnice były jednak często statystycznie nieistotne. Dla zmiennych nieistotnych o rozkładzie wykładniczym sytuacja się nieco zmienia. Re-

liefF zostawia więcej zmiennych w zbiorach *ionosphere* i *ctg*, co prowadzi nieraz do nieco większych błędów klasyfikacji. W pozostałych zbiorach algorytm ReliefF zostawia przeważnie mniej zmiennych w porównaniu z algorytmem Simba_k, który częściej też zostawia zmienne nieistotne. Ten efekt uwidacznia się w błędach klasyfikacji, które dla ReliefF są mniejsze. W przypadku zmiennych nieistotnych z rozkładu mieszanego ReliefF dawał zdecydowanie mniejsze błędy w zbiorach *parkinson* i *sonar*, gdy wprowadzano dużo zmiennych nieistotnych. Dla pozostałych zbiorów różnice błędów nie były istotne, gdy wprowadzano 2*p* zmiennych nieistotnych.

Należy też podkreślić, że ReliefF jest szybszy, nie pracował dłużej niż 4 sekundy, a często poniżej 1 sekundy, co ma znaczenie dla zakresu jego stosowania.

4. Podsumowanie

Przeprowadzone symulacje, polegające na wprowadzaniu do danych rzeczywistych zmiennych nieistotnych, pozwalają sformułować kilka wniosków. Algorytm G-flip stosunkowo dobrze rozpoznaje zmienne nieistotne, a z drugiej strony eliminuje mało zmiennych. Jest jednak wolny, więc można go rekomendować jedynie dla mniejszych zbiorów danych. Zmodyfikowany algorytm Simba z *k* najbliższymi sąsiadami prowadził na ogół do mniejszych błędów klasyfikacji w porównaniu z jego oryginalną wersją. Różnica była znacząca zwłaszcza w przypadku dużej liczby zmiennych nieistotnych. Najszybszym algorytmem, który jednocześnie dobrze identyfikuje zmienne nieistotne, okazał się ReliefF. Algorytm Simba_k potrafi być konkurencyjny ze względu na błąd klasyfikacji, ale nieraz nie rozpoznaje zmiennych nieistotnych, przez co klasyfikator kNN traci dokładność.

Należy podkreślić, że w algorytmach zwracających jedynie wagi zmiennych należało zaproponować sposób wyznaczenia wartości progowej. Wykorzystano tu podejście polegające na ocenie modeli zagnieżdżonych.

Literatura

Enas G.G., Choi S.C., 1986, *Choice of the smoothing parameter and efficiency of k-nearest neighbor classification*, Computer and Mathematics with Applications, vol. 12A(2), s. 235–244.

Gilad-Bachrach R., Navot A., Tishby N., 2004, *Margin based feature selection – theory and algorithms*, Proceedings of the 21st International Conference on Machine Learning, Banff, Canada, s. 43–50.

Guyon I., Gunn S., Nikravesh M., Zadeh L., 2006, *Feature Extraction: Foundations and Applications*, Springer, New York.

Hall M., 2000, *Correlation-based feature selection for discrete and numeric class machine learning*, Proceedings of the 17th International Conference on Machine Learning, Morgan Kaufmann, San Francisco.

Hellwig Z., 1969, *Problem optymalnego wyboru predykant*, Przegląd Statystyczny, nr 3–4.

Kira K., Rendell L.A., 1992, *The feature selection problem: Traditional methods and a new algorithm*, Proc. AAAI-92, MIT Press, s. 129–134.

Kononenko I., 1994, *Estimating attributes: Analysis and extensions of RELIEF*, Proceedings European Conference on Machine Learning, s. 171–182.

Kubus M., 2015, *Feature selection and the chessboard problem*, Acta Universitatis Lodziensis, Folia Oeconomica, Statistical Analysis in Theory and Practice, nr 1(311), s. 17–25 (DOI 11089/21).

Kubus M., 2016, *Lokalna ocena mocy dyskryminacyjnej zmiennych*, Prace Naukowe Uniwersytetu Ekonomicznego we Wrocławiu, nr 427, Taksonomia 27: *Klasyfikacja i analiza danych – teoria i zastosowania*, s. 143–152 (DOI: 10.15611/pn.2016.427.15).

Lichman M., 2013, *UCI Machine Learning Repository*, University of California, School of Information and Computer Science, Irvine, CA, http://archive.ics.uci.edu/ml.