PRACE NAUKOWE AKADEMII EKONOMICZNEJ WE WROCŁAWIU
Nr 1134 — 2006
Nowoczesne technologie informacyjne w zarządzaniu

Janina Anna Jakubczyc

KONTEKSTOWY KLASYFIKATOR ZŁOŻONY

1. Wstęp

Pojedyncze klasyfikatory często nie dają satysfakcjonujących wyników, głównie z powodu trzech ograniczeń [Dietterich 2000], a mianowicie: statystycznego, obliczeniowego oraz reprezentacyjnego. Jednym z proponowanych rozwiązań są klasyfikatory złożone, których idea polega na definiowaniu wielu zadań uczenia w ramach jednego pliku uczącego. Klasyfikator taki otrzymuje się, manipulując plikiem uczącym lub mechanizmem sterowania w algorytmie uczącym ([Kolen, Pollack 1991; Dietterich 2000; Kwok, Certer 1990]). Celem niniejszego artykułu jest zdefiniowanie różnych zadań uczenia poprzez manipulację etykietami przykładów, z zastosowaniem kontekstów danych uczących.

Jego kolejne rozdziały to wprowadzenie do problematyki klasyfikatorów złożonych ze wskazaniem możliwych modyfikacji ich tworzenia, propozycja rozwiązań i wyniki wstępnych badań oraz podsumowanie wyników.

2. Zasady tworzenia klasyfikatorów złożonych

Klasyfikator złożony jest zbiorem klasyfikatorów, kwalifikującym nowe przypadki przez łączenie decyzji pojedynczych klasyfikatorów w określony sposób. Przesłanką łączenia klasyfikatorów jest to, że klasyfikator złożony może osiągnąć wyższą trafność klasyfikacyjną niż pojedynczy. Jest to stosunkowo nowa koncepcja, chociaż pomysł agregacji opinii zespołu ekspertów nowy nie jest. Teoretyczne podstawy tej metody stanowi twierdzenie Jury sformułowane przez Condorceta, cytowane m.in. w pracy ShiXina [2003, s. 26]. Natomiast w odniesieniu do klasyfikatorów, teoretyczne rozważania można znaleźć m.in. w pracy Hansena i Salamona [1990].

Manipulacja materiałem uczącym obejmuje: sposoby próbkowania danych uczących, dobór atrybutów klasyfikacyjnych (metod selekcji i ekstrakcji atrybutów

jest bardzo dużo, a interesujący ich przegląd można znaleźć w literaturze (por. [Rousu 2001; Ricci, Aha 1997]) oraz manipulację etykietami klas.

Próbkowanie polega na wygenerowaniu różnych podzbiorów uczących, dla których generowane są różne klasyfikatory. Stosuje się trzy sposoby tworzenia plików uczących. Pierwszy, o nazwie *bootstrap*, polega na generowaniu – przez losowanie ze zwracaniem – pewnej liczby zbiorów uczących zawierających taką samą liczbę przykładów jak wejściowy plik uczący. W wyniku zastosowania takiego mechanizmu doboru można otrzymać różną strukturę zbiorów uczących, a więc można utworzyć różne klasyfikatory, które wspólnie będą opisywały dane uczące. Słabością tego sposobu jest to, że nie wiadomo, kiedy i które w ten sposób powstałe pliki uczące są odpowiednie. Należy więc utworzyć sporo takich plików i zastosować do każdego z nich algorytm uczący, a na podstawie jego wyniku wybrać właściwe. Poszerzeniem tego jest wprowadzenie zaburzenia losowego wartości atrybutów wejściowych dla wybranej liczby przykładów uczących [Raviv, Intrator 1996].

Drugim sposobem generowania plików uczących jest zastosowanie weryfikacji krzyżowej (*cross-validation*). Pliki tworzy się na podstawie różnych wariantów plików uczących przy zadanej liczbie podziału k, dzielącej plik uczący na k części. Następnie usuwa się jedną z nich i stosuje się algorytm uczenia do pozostałych k–1 podzbiorów. Liczba powstałych klasyfikatorów jest więc równa przyjętej liczbie podziału.

Kolejną metodą tworzenia plików uczących jest AdaBoost (*adaptive boosting*) [Schapire 1998; 1999]. Jej idea polega na tym, by wszystkie przykłady zostały poprawnie zaklasyfikowane w kolejnych iteracyjnych wywołaniach algorytmu klasyfikacyjnego. Do tego celu wykorzystuje się zbiór wag przypisywanych przykładom uczącym, większe wagi nadaje się przykładom, które nie zostały poprawnie zaklasyfikowane. Po każdej iteracji zmieniany jest układ wag. Klasyfikator tworzony jest dla każdej iteracji. AdaBoost rozwija się i obecnie występuje w trzech wariantach, agresywnym, konserwatywnym i inwersyjnym [Kuncheva 2005].

Kolejnym sposobem tworzenia klasyfikatorów jest manipulowanie etykietami przykładów nazwane przez Dietterichа i Bakiriego korektą błędów kodów wyjść [Dietterich, Bakirri 1995; Adeva, Baresi, Calvo 2000]. Warunkiem zastosowania tego pomysłu jest duża liczba klas i przykładów uczących. Pierwszym krokiem jest losowy podział klas na dwa podzbiory, którym przypisywane są dwie nowe etykiety, np. I i II. Za pomocą algorytmu uczącego generowany jest klasyfikator opisujący klasę I i II. W następnym kroku w ramach każdej klasy uruchamiany jest algorytm, który uczy się pierwotnych klas zawartych w wygenerowanym podzbiorze uczącym. Tę dwustopniową procedurę uruchamia się pożądaną liczbę razy. W tym przypadku mamy do czynienia z klasyfikatorem hierarchicznym, w którym najpierw określana jest przynależność do klasy I lub II, a następnie do klasy zgodnej z przykładami uczącymi zawartymi w pierwotnym pliku uczącym za pomocą wygenerowanych klasyfikatorów. Sposób ten, jak wynika z przeprowadzo-

nych badań [Ricci, Aha 1997; Dietterich, Bakiri 1995; Adeva, Beresi Calvo 2005; Kuncheva 2005; Masulli, Valentini 2004], jest szczególnie polecany w trudnych problemach klasyfikacyjnych.

Implementacja klasyfikatora złożonego wymaga zdefiniowania schematu działania, który określi, jakie klasyfikatory biorą udział w generowaniu wyników klasyfikacji oraz jak je lub ich wyniki ze sobą łączyć. Podstawowym schematem jest głosowanie, spotykane w trzech formach: głosowania większościowego, głosowania ważonego oraz głosowania większościowego ważonego [ShiXin 2003]. W schemacie głosowania większościowego wszystkie pojedyncze klasyfikatory są równe. Wybierany jest klasyfikator, który uzyska najwięcej głosów. Natomiast w głosowaniu ważonym klasyfikatorom składowym nadaje się wagi, które mogą zależeć np. od trafności klasyfikacyjnej w przeszłości lub jakiejś heurystyki. Schemat ważonego głosowania większościowego różni się od poprzedniego sposobem generowania wag, będących wynikiem procesu uczenia, w którym sprawdzana jest trafność poszczególnych klasyfikatorów.

Schematów głosowania jest znacznie więcej i obszar ten intensywnie się rozwija. Zaczynają powstawać jako nieodłączny element wybranego podejścia do tworzenia klasyfikatorów złożonych, i tak np. w pracy [Xu, Krzyżak, Suen, 1997] proponuje się, jako możliwe rozwiązania, trzy schematy w klasyfikatorach złożonych tworzonych na podstawie selekcji atrybutów klasyfikacyjnych. Są nimi: liniowa kombinacja wyników składowych klasyfikatora złożonego, „zwycięzca bierze wszystko", wnioskowanie dowodowe.

3. Propozycja rozszerzenia możliwości tworzenia klasyfikatorów złożonych

Z dotychczasowych rozważań nasuwa się wniosek, że mechanizmem do tworzenia zróżnicowanych plików uczących jest randomizacja. Wyjątkiem jest podejście AdaBoost, w którym w kolejnych iteracjach generuje się wagi wskazujące błędy klasyfikacji oraz manualny dobór atrybutów klasyfikacyjnych, bazujący na dziedzinie zadania klasyfikacyjnego.

Pomimo wykazanej skuteczności klasyfikatorów w przypadku generowania zbiorów uczących napotyka się następujące kwestie:

a) liczba prób (losowań), którą należy przeprowadzić, by mieć gwarancję lepszego opisu,
b) interpretacja różnic pomiędzy różnymi plikami danych uczących,
c) interpretacja różnych wyników,
d) zrozumienie działania klasyfikatora (kiedy i dlaczego jeden jest lepszy od drugiego),
e) sposób łączenia klasyfikatorów składowych (który z nich wybrać i w jakiej sytuacji).

Przedstawione kwestie są ważne, gdyż są one wynikiem braku możliwości celowego działania przy tworzeniu opisu danych, a więc zdania się na przypadek. Pozbawienie możliwości interpretacji i rozumienia danych i ich opisu jest zasadne, gdy nie dysponujemy innymi sposobami, ale należy szukać nowych lepszych rozwiązań.

Propozycja autorki, rozwiązująca niektóre z przedstawionych kwestii, dotyczy tworzenia plików uczących dla klasyfikatorów złożonych z wykorzystaniem kontekstów danych uczących. Przyjętą reprezentacją kontekstowego klasyfikatora złożonego i klasyfikatorów składowych jest drzewo decyzyjne.

3.1. Tworzenie plików uczących z wykorzystaniem kontekstu danych

Proponowane podejście jest modyfikacją manipulacji etykietami Dietterichа i Bakiriego [1995], zwanej korektą błędów kodów wyjść. Polega ona na zamianie losowego podziału pliku uczącego na kontekstowy podział według zidentyfikowanych atrybutów kontekstowych. Każdy atrybut kontekstowy dzieli plik uczący na podzbiory, których liczba jest zależna od struktury wartości, jakie przyjmuje cecha kontekstowa. Na podstawie każdego podzbioru generowany jest klasyfikator.

Podstawą do odkrycia kontekstu są definicje atrybutów podstawowych (decyzyjnych), kontekstowych i kontekstowo zależnych Turney [1993] przedstawia je następująco (przyjmując, że: X_i atrybut opisujący, Y – atrybut klasy, α_i minimalny, a β_i maksymalny rozmiar kontekstu, p – prawdopodobieństwo, $S_{i,j}$ – zbiór wszystkich atrybutów z wyjątkiem X_i i X_j, $s_{i,j}$ – wartości wszystkich atrybutów w zbiorze $S_{i,j}$):

a) atrybut jest podstawowy wtedy i tylko wtedy, gdy $\alpha_i=0$; oznacza to, że istnieją pewne wartości x_i oraz y, dla których $p(X_i=x_i)>0$, takie że: $p(Y=y|X_i=x_i)\neq p(Y=y)$;
b) atrybut jest kontekstowy wtedy i tylko wtedy, gdy $\alpha_i>0$, co oznacza, że dla wszystkich x_i i y zachodzi: $p(Y=y|X_i=x_i)=p(Y=y)$;
c) atrybut X_i jest kontekstowo zależny względem atrybutu X_j wtedy i tylko wtedy, gdy istnieje podzbiór atrybutów $S'_{i,j}$ zbioru $S_{i,j}$, dla których istnieją pewne x_i, x_j, $s'_{i,j}$ oraz y i dla których: $p(X_i=x_i,X_j=x_j,S'_{i,j}=s'_{i,j})>0$ takie, że spełnione są następujące warunki: $p(Y=y|X_i=x_i,X_j=x_j,S'_{i,j}=s'_{i,j})\neq p(Y=y|X_i=x_i,S'_{i,j}=s'_{i,j})$, $p(Y=y|X_i=x_i,X_j=x_j,S'_{i,j}=s'_{i,j})\neq p(Y=y|X_j=x_j,S'_{i,j}=s'_{i,j})$.

Sposób tworzenia plików uczących zależnych od zidentyfikowanych atrybutów podstawowych, kontekstowych i kontekstowo zależnych, przedstawia się następująco:

1. Utworzenie drzewa decyzyjnego na podstawie pełnego materiału uczącego; ten krok umożliwia podział cech na te, które mają bezpośredni wpływ na zadanie klasyfikacji (atrybuty podstawowe), i te, które wpływu nie mają (atrybuty niedecyzyjne).
2. Utworzenie drzewa decyzyjnego dla każdego atrybutu decyzyjnego, z uwzględnieniem tylko atrybutów niedecyzyjnych; w tym kroku szuka się wśród atry-

butów niedecyzyjnych, atrybutów kontekstowych, a wśród atrybutów decyzyjnych (podstawowych) cech kontekstowo zależnych.

3. Identyfikacja, na podstawie zadanego poziomu trafności klasyfikacyjnej, wygenerowanych par: atrybut kontekstowy i kontekstowo zależny, które można zastosować do podziału pliku uczącego.
4. Podział pliku uczącego według rozkładu wartości atrybutu kontekstowego i odpowiadającego mu atrybutu kontekstowo zależnego.

Badania przeprowadzono na danych z repozytorium maszynowego uczenia *Credit Approval*. Zbiór zawiera 690 przykładów opisanych piętnastoma niejawnymi atrybutami, które zostały nazwane kolejnymi liczebnikami: jeden, dwa, ... , piętnaście. Ostatni szesnasty atrybut to „decyzja" o przyznaniu lub odrzuceniu wniosku kredytowego. Dziewięć atrybutów jest symbolicznych, a sześć – numerycznych.

Przeprowadzone badania przebiegały w kilku etapach. W pierwszym etapie wygenerowano drzewo decyzyjne[1] dla pełnego zbioru danych. W wyniku otrzyma-

```
decyzja
dziewięć = t :
    piętnaście = f lub piętnaście = i lub piętnaście = g lub  piętnaście = d lub piętnaście = c lub
    piętnaście = h lub piętnaście = b : tak (135.0)
    piętnaście = a :
        czternaście = d : tak (26.0)
        czternaście = f : nie (8.0)
        czternaście = c :
            dwanaście = f : tak (11.0)
            dwanaście = t : nie (13.0)
        czternaście = 14b :
            dziesięć = t : tak (24.0)
            dziesięć = f : nie (27.0)
        czternaście = a :
            sześć = w lub sześć = q lub sześć = m lub sześć = r lub sześć = cc lub sześć = k lub
            sześć = c lub sześć = d lub sześć = x lub sześć = i lub sześć = e lub sześć = aa : tak (56.0)
            sześć = ff : nie (4.0)
            sześć = j : tak (1.0)
dziewięć = f :
    trzy = l lub trzy = p lub trzy = o lub trzy = y lub trzy = r lub trzy = w: nie (91.0)
    trzy = t : tak (2.0)
    trzy = v lub trzy = u lub trzy = s lub trzy = m lub trzy = n lub trzy = z : nie (84.0)
    trzy = k :
        czternaście = c lub czternaście = b lub czternaście = d lub czternaście = a : nie (62.0)
        czternaście = f : tak (8.0)
```

Rys. 1. Drzewo decyzyjne – atrybuty decyzyjne

Źródło: opracowanie własne.

[1] Do obliczeń wykorzystano moduł DETREEX pakietu SPHINX do generowania drzew decyzyjnych, firmy AITECH w Katowicach.

no listę atrybutów decyzyjnych (podstawowych). Aby zapewnić pełen ogląd sytuacji klasyfikacyjnej przy stosunkowo niewielkiej liczbie atrybutów, wyłączono przycinanie drzewa, a zostawiono ograniczenie do minimalnej liczby przypadków (10). Atrybutów decyzyjnych jest 7, a mianowicie: dziewięć, piętnaście, czternaście, dwanaście, sześć, dziesięć, trzy. Wygenerowane drzewo w postaci tekstowej przedstawiono na rys. 1.

Następnym etapem było wyjaśnienie każdego atrybutu decyzyjnego tylko atrybutami pozadecyzyjnymi (zgodnie z definicją atrybutu kontekstowego, nie wpływa on bezpośrednio na przynależność do klasy, ma natomiast wpływ na zmienną decyzyjną). Zadanie to polegało na wygenerowaniu drzew dla każdego atrybutu decyzyjnego (atrybut decyzyjny, jako klasa) i wyborze par: atrybut decyzyjny – atrybut kontekstowy (atrybuty kontekstowe). Podstawę wyboru stanowiła trafność klasyfikacyjna powyżej 60%. Zestawienie wyników przedstawiono w tab. 1.

Tabela 1. Zestawienie trafności klasyfikacyjnej dla atrybutów decyzyjnych

Atrybut decyzyjny	dziewięć	piętnaście	trzy	czternaście	dwanaście	dziesięć	sześć
Trafność klasyfikacyjna	75%	62,78%	30%	31,67%	61,11%	100,0%	38,89%

Źródło: opracowanie własne.

Atrybutami kontekstowo zależnymi są: dziewięć, piętnaście i dwanaście. Wyboru atrybutów kontekstowych dokonano, analizując wygenerowane drzewa decyzyjne dla atrybutów kontekstowo zależnych. Przykładowe drzewo dla atrybutu dziewięć przedstawia rys. 2.

```
dziewięć
jedenaście > 2 : t (149.0)
jedenaście <= 2 : ...
            ...
jedenaście <= 0 : f (35.0)
jedenaście > 0 : t (16.0) ...
```

Rys. 2. Atrybuty wyjaśniające atrybut decyzyjny dziewięć

Źródło: opracowanie własne.

Tabela 2. Pary atrybutów kontekstowych i kontekstowo zależnych

Atrybut kontekstowozależny	dziewięć	piętnaście	dwanaście
Atrybut kontekstowy	jedenaście	trzy	osiem

Źródło: opracowanie własne.

Na podstawie analizy każdego z drzew wybrany został jeden atrybut kontekstowy; w powyższym przypadku – jedenaście. Wyniki dla pozostałych atrybutów kontekstowo zależnych przedstawia tab. 2.

3.2 Wyniki eksperymentu i ich analiza

Zidentyfikowane cechy kontekstowe dają możliwość rozróżnienia kontekstów w materiale uczącym i opisania ich za pomocą odrębnych zbiorów reguł klasyfikacyjnych. Kontekst wyznaczają cechy: jedenaście, trzy i osiem, i one stanowią wyznaczniki podziału zbioru uczącego i testującego na grupy o różnych kontekstach. Należy zatem wyznaczyć przedziały wartości atrybutu kontekstowego, dobrze opisujące zidentyfikowany kontekst. Podstawą jest zawartość informacyjna atrybutów kontekstowych względem atrybutu kontekstowo zależnego dziewięć (tab. 3).

Tabela 3. Zestawienie zawartości informacyjnej dla kontekstowego atrybutu jedenaście względem atrybutu dziewięć

Wartości atrybutu jedenaście	11	7	5	6	12	0	4	3	2	1	10	8, 9, 13,14,15, 20
Zawartość informacyjna	0,63	0,58	0,38	0,36	0,32	0,21	0,16	0,06	0,04	0,00	–0,01	–0,03

Źródło: opracowanie własne.

Biorąc pod uwagę zawartość informacyjną oraz stosując zasadę unikania tworzenia zbyt dużej liczby przedziałów wartości, przyjęto, że satysfakcjonującą liczbą podziału na dwa podzbiory, w przypadku kontekstowego atrybutu jedenaście, jest wartość 4. Zestawienie punktów podziału i liczności plików uczących dla każdego kontekstu, jako nowego zadania klasyfikacyjnego, przedstawiono w tab. 4.

Tabela 4. Zestawienie punktów podziału i liczności plików uczących

	Jedenaście		Osiem		Trzy	
Punkt podziału	4		0,21		0	
Liczność	539	131	187	503	395	295

Źródło: opracowanie własne.

Wszystkie wydzielone podzbiory pliku uczącego stanowiły podstawę do wygenerowania kontekstowego klasyfikatora złożonego. Trafność klasyfikacyjną jego oraz jego elementarnych klasyfikatorów zawiera tab. 5.

Przedstawione wyniki jednoznacznie wskazują na znaczną przewagę efektywności kontekstowego klasyfikatora złożonego zarówno nad efektywnością klasyfikatora pojedynczego (11,2%), jak i nad efektywnością klasyfikatorów składowych (13,3-2,4%).

Tabela 5. Zestawienie trafności klasyfikacyjnej klasyfikatorów

Trafność klasyfikacyjna klasyfikatora pojedynczego					82,2%	
Trafność kontekstowego klasyfikatora złożonego					93,4%	
Trafność klasyfikacyjna składowych kontekstowego klasyfikatora złożonego	jedenaście		osiem		trzy	
	80,1%	88%	86,3%	85,6%	89,2%	91,0%

Źródło: opracowanie własne.

Kierując się charakterem tworzonych plików uczących, autorka zdecydowała się na schemat głosowania większościowego. Na etapie wstępnym wybór ten wydał się słuszny, natomiast w dalszych badaniach okazało się, że należy rozważyć głosowanie ważone siłą relacji miedzy występującymi kontekstami. Kontekstowy klasyfikator złożony, kwalifikując nowe przypadki, oprócz podania klasy i trafności wyboru podaje kontekst dominującego klasyfikatora. Dostaje się więc pełniejszą informację, która umożliwia właściwą interpretację.

4. Podsumowanie

Zastosowanie kontekstowego klasyfikatora złożonego służy dwóm celom. Po pierwsze zdecydowanie dokładniejszemu i bardziej adekwatnemu opisowi analizowanych zjawisk, co może przyczynić się również do lepszego ich rozumienia. Przekłada się to na większe możliwości celowych modyfikacji danych. Drugim celem jest zwiększenie efektywności reguł klasyfikacyjnych.

Prezentowane podejście ma jeszcze inne zalety. Przede wszystkim generuje się tylko pliki dla zidentyfikowanych atrybutów kontekstowych, nie ma więc problemu z tym, ile plików uczących utworzyć i które z nich wybrać. Nie ma potrzeby sprawdzania, czy utworzone pliki spełniają zasadę różnorodności hipotez z nich generowanych, gdyż gwarantują ją różne konteksty. Nie ma kłopotów z interpretacją wygenerowanych plików uczących z wykorzystaniem kontekstu. Istnieje możliwość wyjaśnienia, który klasyfikator jest lepszy i dlaczego – dla danego przypadku.

Niewątpliwym ograniczeniem zaproponowanej koncepcji jest konieczność występowania wielu kontekstów w danych, co ogranicza obszar jego zastosowań. Kontekstowe klasyfikatory złożone mogą znaleźć zastosowanie nie tylko do kontekstu ukrytego w danych. Poszerzeniem ich zastosowania może być uwzględnienie zewnętrznych kontekstów, których naturalnym przykładem mogą być grupy danych pochodzących z różnych źródeł. Klasyfikator złożony będzie wówczas integratorem danych.

Interesującym obszarem badawczym może być tworzenie złożonych klasyfikatorów kontekstowych, a także tworzenie sztucznych cech kontekstowych jako złożenia (liniowego lub innego) pojedynczych atrybutów kontekstowych. Przeprowadzony eksperyment jest badaniem wstępnym. Prace nad tworzeniem kontekstowych klasyfikatorów złożonych są kontynuowane.

Literatura

Adeva J.J.G., Beresi U.C., Calvo R.A., *Accuracy and Diversity in ECOC Ensembles of Text Categorisers*, dostępny: http://citeseer.ist.psu.edu/732806.html, 2000.

Adeva J.J.G., Calvo R.A., *A Decomposition Scheme Based on Error-Correcting Output Codes for Ensembles of Text Categorisers*, Proceedings of the International Conference on Information Technology and Applications (ICITA), IEEE Computer Society, 2005.

Credit Approval. Repository of Machine Learning Databases [http://www.ics.uci.edu/~mlearn/ /MLRepository.html], Confidential – quinlan@cs.su.oz.au

Dietterich T.G., Bakiri.G., *Solving Multiclass Learning Problems via Error-correcting Output Codes*, „Journal of Artificial Intelligence Research" 1995 nr 2, s. 263-286.

Dietterich T.G., *Ensemble Methods in Machine Learning*, Proc. of 1th Intern. Workshop on Multiple Classifier Systems, 2000, s. 1-15.

Hansen L., Salamon P., *Neural Network Ensembles*, „IEEE Trans. Pattern Analysis and Machine Intelligence" 1990 nr 12, s. 993-1001.

Kolen J.K., Pollack J.B., *Back Propagation is Sensitive to Initial Conditions*, [w:] *Advances in Neural Information Processing Systems*, vol. 3, s. 860-867, CA. Morgan Kaufmann, 1991.

Kuncheva L.I., *Using Diversity Measures for Generating Error-correcting Output Codes in Classifier Ensembles*, „Pattern Recognition Letters" 2005, nr 26, s. 83-90.

Kwok S.W., Carter C., *Multiple Decision Trees*, [w:] *Uncertainty in Artificial Intelligence*, red. R.D. Schachter, T.S. Levitt, L.N. Kannal, J.F. Lemmer, Elsevier Science, Amsterdam 1990, s. 327-335.

Masulli F., Valentini G., *An Experimental Analysis of the Dependence among Codeword Bit Errors in ECOC Learning Machines*, „Neurocomputing" 2004 nr 57C, s. 189–214.

Raviv Y., Intrator N., *Bootstrapping with Noise: An Effective Regularization Technique*, „Connection Science" 1996 nr 8(3-4), s. 355-372.

Ricci F., Aha D.W., *Extending Local Learners with Error-correcting Output Codes*, Technical Report, Naval Center for Applied Research in AI, Washington, D.C., 1997.

Rousu J., *Efficient Range Partitioning in Classification Learning*, Dep. of Computer Science Series of Publications University of Helsinki, *A Report A-2001-1*, dostępny: http://citeseer.ist.psu.edu/ /correct/394680

Schapire R.E, i in., *Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods*, Proc. 14th Intern. Conf. on Machine Learning, 1998.

Schapire R.E., *The Theoretical Views of Boosting and Applications, Algorithmic Learning Theory*, Proceedings of The 10th Intern. Conf., ALT '99, Tokyo, Japan 1999.

ShiXin Y., *Feature Selection and Classifier Ensembles: A Study on Hyperspectral Remote Sensing Data*, PhD The University of Antwerp 2003.

Turney P.D., *Robust Classification with Context-sensitive Features*, „Industrial and Magineering Applications of Artificial Intelligence and Expert Systems", Gordon and Breach, Edinburgh 1993.

Xu L., Krzyżak A., Suen C.Y., *Methods of combining multiple classifiers and their applications to handwriting recoguition*, „IEEE Transations SMC" 1997, vol. 22, s. 418-434.

THE CONTEXT ENSEMBLE CLASSIFIER

Summary

Ensemble methods are the means for difficult learning task in supervised learning. The idea is to construct a collection of individual classifiers on the basis of one learning set. This task can be accomplished by manipulation of data set or control mechanism of learning algorithm. This paper proposal is defining different learning task using context inserted in data set by manipulating the output targets.

Dr Janina Anna Jakubczyc jest adiunktem w Katedrze Systemów Sztucznej Inteligencji Akademii Ekonomicznej we Wrocławiu, e-mail: janina.jakubczyc@ae.wroc.pl.