UNIWERSYTET WARSZAWSKI
Wydział Matematyki, Informatyki i Mechaniki

METODY WNIOSKOWANIA W OPARCIU O NIEKOMPLETNY OPIS OBIEKTÓW

PRACA MAGISTERSKA

Rafał Latkowski
rlatkows@mimuw.edu.pl

Praca pod kierunkiem
prof. dra hab. Andrzeja Skowrona
A.Skowron@mimuw.edu.pl

Warszawa 2001

Streszczenie

Praca stanowi przegląd metod umożliwiających wnioskowanie w oparciu o dane z niekompletnym opisem obiektów. Przedstawione są tutaj zarówno metody mające na celu uzupełnianie brakujących wartości jak i takie, które starają się wnioskować bezpośrednio w oparciu o dane z niekompletnym opisem obiektów. Zamierzeniem autora było możliwie najbardziej kompletne zestawienie metod stosowanych analizie danych i odkrywaniu wiedzy wraz ze wskazaniem, z której dziedziny matematyki się wywodzą. Rozdział pierwszy wprowadza czytelnika w problematykę analizy danych i obiektów o niekompletnym opisie. Rozdział drugi stanowi wstęp do teorii zbiorów przybliżonych i na tej podstawie porusza podstawowe zagadnienia związane z wnioskowaniem na podstawie danych. W trzecim rozdziale zaprezentowane są rozszerzenia teorii zbiorów przybliżonych, umożliwiające wnioskowanie w obliczu brakujących wartości atrybutów. Rozdział czwarty prezentuje metody wnioskowania w oparciu o dane z niekompletnym opisem obiektów, nie wywodzące się z nurtu zbiorów przybliżonych. W rozdziale piątym opisane zostały metody realizujące paradygmat leniwego uczenia się pojęć. Rozdział szósty prezentuje rozwiązania eliminujące brakujące wartości podczas wstępnego przetwarzania danych za pomocą uzupełniania. Na zakończenie prezentowana jest nowa metoda, umożliwiająca zaadaptowanie istniejących algorytmów uczenia się pojęć do danych z brakującymi wartościami obiektów. Zamieszczone wyniki eksperymentalne wskazują na dużą skuteczność tej metody.

Słowa kluczowe

systemy decyzyjne, wnioskowanie indukcyjne, zbiory przybliżone, brakujące wartości atrybutów

Klasyfikacja tematyczna

Klasyfikacja tematyczna według AMS MSC 2000: 68T37, 68U35.

Spis treści

Rozdział 1

Wprowadzenie

Od momentu powstania maszyn umożliwiających przetwarzanie informacji — komputerów, myślą zaprzątającą umysły wielu ludzi, czy to badaczy, czy też reżyserów filmów S-F, jest możliwość skonstruowania maszyny inteligentnej. Bardzo trudno jest jednak zdefiniować, czym dokładnie jest owa inteligentna maszyna. Jak czytamy w encyklopedii [1], inteligencja to zespół zdolności umysłowych, umożliwiających jednostce sprawne korzystanie z nabytej wiedzy, oraz skuteczne zachowanie się wobec nowych zadań i sytuacji.

1.1 Inteligentne przetwarzanie informacji

W dzisiejszych czasach, na początku XXI wieku, rozwijane od dziesięcioleci systemy komputerowe umożliwiają składowanie gigantycznych wręcz ilości informacji. Mogą to być dane dotyczące badań medycznych, zdjęcia satelitarne ziemi, informacje o sterowaniu urządzeń, transakcje dokonywane w sklepach czy też dane dotyczące wypadków. Wszystkie te informacje, wykorzystane w należyty sposób, mogą posłużyć do coraz skuteczniejszego zachowania się wobec nowo powstałych sytuacji i zadań. Przy diagnozowaniu pacjenta nieocenioną pomocą jest wiedza uzyskana na podstawie analizy danych medycznych, tak jak przy poszukiwaniu złóż surowców mineralnych posługiwanie się zdjęciami satelitarnymi ziemi. Zgodnie z powyższą definicją skuteczne rozwiązanie tych problemów wymaga inteligencji, czyli inteligentnego przetwarzania informacji. Jednakże zgromadzone zbiory danych częstokroć przekraczają możliwości percepcji człowieka. Pomocą do sprawnego wykorzystywania tej wiedzy mogą być systemy komputerowe inteligentnie przetwarzające informacje.

Na przestrzeni wielu lat podejmowano liczne próby skonstruowania maszyny umożliwiającej inteligentne przetwarzanie informacji. Sztuczna inteligencja, bo tak można określić całokształt tych zjawisk, jest dzisiaj dość dobrze rozwiniętą dziedziną wiedzy, w której można wyróżnić takie działy jak maszynowe uczenie się, systemy decyzyjne, rozpoznawanie wzorców, systemy wieloagentowe, odkrywanie wiedzy, przetwarzanie języka naturalnego i wiele innych. Pomimo licznych osiągnięć człowiek pozostał jednak niedoścignionym wzorem inteligencji.

1.2 Logika

Podstawowym narzędziem inteligentnego przetwarzania informacji jest logika. Za pomocą logiki staramy się opisać i naśladować sposób rozumowania człowieka. Na przestrzeni dziejów podejmowano różne próby sformalizowania tego typu rozumowań. Pierwszą i najbardziej znaną jest tzw. logika klasyczna, wprowadzona przez greckich filozofów już w starożytności i opierająca się na wnioskowaniu dedukcyjnym. Pomimo jej szerokich zastosowań na potrzeby informatyki i matematyki, posiada liczne ograniczenia, jak monotoniczność i niepełność systemów dedukcyjnych, uniemożliwiające jej użycie do wiernego naśladowania tzw. rozumowań zdroworozsądkowych. W celu uniknięcia trudności z formalizowaniem rozumowań przeprowadzanych przez człowieka wprowadzono liczne odmiany logik, które można podzielić na dwie grupy ze względu na sposób podejścia do problemu. Są to tzw. metody symboliczne i numeryczne. Wśród podejść symbolicznych należy wymienić głównie logiki niemonotoniczne i modalne. Metody numeryczne reprezentowane są przez takie logiki jak logika posybilistyczna, czy logika rozmyta. Jednakże główną niedogodnością zastosowania logiki do analizy i inteligentnego przetwarzania danych jest sam proces wnioskowania dedukcyjnego, czyli rozumowania przeprowadzonego od przesłanek do wniosków za pomocą dowodu formalnego w rozpatrywanym systemie dedukcyjnym.

1.3 Wnioskowanie indukcyjne

Rozumowania przeprowadzane przez człowieka cechuje duża łatwość konstrukcji skomplikowanych wniosków. O tym, że sposób wnioskowania człowieka charakteryzuje się wielką sprawnością i skutecznością, nie trzeba nikogo przekonywać. Jednakże wnioski formułowane przez ludzi nie zawsze okazują się prawdziwe. Poprawność procesu wnioskowania jest ceną, jaką trzeba zapłacić za możliwość szybkiej i skutecznej analizy skomplikowanych sytuacji.

Rozumowania takie możemy przybliżyć za pomocą wnioskowania indukcyjnego. We wnioskowaniu indukcyjnym jako prawdziwe uznajemy zdanie stwierdzające jakąś ogólną prawidłowość, przy czym czynimy to na podstawie uznania zdań stwierdzających poszczególne przypadki tej prawidłowości. Bazując na doświadczeniu i obserwacjach staramy się sformułować wnioski dotyczące nowych sytuacji. Oczywiście wnioskowanie takie nie jest niezawodne, gdyż wnioskując na podstawie prawdziwych przesłanek możemy dojść do fałszywego wniosku. Jeśli bowiem istnieją przypadki spełniające pewną prawidłowość, nie oznacza to wcale, że prawidłowość ta będzie zawsze spełniona. Niemniej jednak wnioskowanie takie jest najbardziej adekwatną metodą przeprowadzania rozumowań w procesie inteligentnego przetwarzania informacji.

W teorii uczenia się maszyn wnioskowanie indukcyjne pojawia się przy okazji problemu uczenia się pojęć w oparciu o przykłady. Problem ten polega na utworzeniu opisu pojęcia, rozumianego jako podzbiór zbioru obiektów należących do rozpatrywanego środowiska, na podstawie przykładów badanego pojęcia. Przez utworzenie opisu pojęcia rozumiemy wykrycie takich własności przykładów obiektów, które umożliwią późniejsze badanie nowych przykładów pod kątem ich przynależności do tego pojęcia. Naturalnym podejściem do rozwiązania problemu uczenia się pojęć na podstawie przykładów jest wnioskowanie indukcyjne, polegające na tym, że otrzymując kolejne przykłady obiektów należących i nie należących do pojęcia, próbuje się znaleźć taki jego opis, który będzie pasował do wszystkich lub

prawie wszystkich przykładów badanego pojęcia. Opis pojęcia formułowany jest w języku logiki i stanowi właśnie wyuczoną ogólną prawidłowość decydującą o należeniu przykładów do badanego pojęcia.

Głównym problemem związanym z uczeniem się pojęć w oparciu o przykłady jest pytanie w jaki sposób konstruować algorytmy, które potrafią wyuczyć się badanego pojęcia w oparciu o dostarczone dane. Przy czym algorytmy te mają osiągnąć jak największą poprawność formułowanych wniosków.

1.4 Niedoskonałość danych

Dane pochodzące ze świata rzeczywistego opisują nieraz bardzo skomplikowane procesy zachodzące w badanym środowisku. Podczas analizy takich danych napotykamy na liczne trudności spowodowane szumem informacyjnym, niedokładnością i błędami pomiaru, czy wreszcie brakiem niektórych informacji. Wiele teoretycznie dopracowanych podejść okazało się nieskutecznymi w konfrontacji z rzeczywistością. Niedoskonałość informacji wprowadza wiele utrudnień do procesu wnioskowania w oparciu o dane. Jednakże te niedoskonałości nie powinny uniemożliwiać skutecznego formułowania wniosków, czego najlepszym przykładem jest człowiek, potrafiący zachować zdolność do przeprowadzania rozumowań nawet w obliczu niedoskonałych i nieprecyzyjnych danych. Niektóre z mechanizmów niedokładności informacji zostały gruntownie zbadane i sformułowano liczne, zadowalające rozwiązania tych problemów.

Analiza głównych składowych i wykrywanie cech znaczących to środki umożliwiające zmierzenie się z problemem szumu informacyjnego. Pozwalają one na wybór interesującej informacji i odrzucenie niepotrzebnej. Metody selekcji istotnej informacji rozwijane były na gruncie statystyki, przetwarzania sygnałów oraz analizy danych i odkrywania wiedzy.

Na potrzeby rozwiązania problemu nieprecyzyjności danych wymyślono wiele podejść, wśród których dominują podejścia logiczno-numeryczne, ale nie tylko. Znakomitym przykładem jest tutaj teoria zbiorów przybliżonych, która umożliwia w sposób formalny ująć nieprecyzyjność danych w postaci pojęć teoriomnogościowych.

Na tym tle osiągnięcia, mające na celu rozwiązanie problemu braku informacji, wydają się być niewielkie. Należy zauważyć, że wśród możliwych rodzajów braku informacji niektóre są z nich są naturalne i nie do uniknięcia, a wręcz korzystne. Badając konkretne zjawisko nie wymagana jest informacja dotycząca nieistotnych parametrów badanego środowiska, co wiąże się z problemem szumu informacyjnego i ograniczonych fizycznie możliwości percepcji. Dotkliwym brakiem informacji jest natomiast niedostępność istotnych cech dla rozpatrywanego problemu. Niniejsza praca poświęcona jest szczególnemu rodzajowi braku informacji, mianowicie niekompletnemu opisowi obiektów.

Najbardziej istotnym brakiem informacji, pozostającym w zakresie zainteresowań inteligentnego przetwarzania informacji jest niekompletny opis obiektów. Sytuacja taka występuje, gdy obiekty pochodzące z badanego środowiska cechuje zróżnicowany poziom dostępnej informacji o tych obiektach.

1.5 Brakujące wartości atrybutów

Wszystkie dane przetwarzane w systemach komputerowych opisane są za pomocą zbioru wartości z pewnych dziedzin, czyli tzw. atrybutów. Ustalając badane środowisko i obiekty z niego pochodzące ustala się zbiór cech — atrybutów, które opisują własności badanych obiektów. Gromadzone dane to zbiór opisanych w ten sposób obiektów. Przez obiekt rozumie się wtedy zbiór wartości wybranych uprzednio atrybutów. Problem brakujących wartości atrybutów występuje wtedy, gdy niektóre obiekty nie są opisane na całym zbiorze cech. W zgromadzonych danych brakuje niektórych wartości atrybutów.

Jest to istotny problem podczas procesu wnioskowania. Stosowane zazwyczaj podejścia nie uwzględniają zróżnicowania w opisie obiektów i zakładają, że wszystkie obiekty muszą być opisane na wszystkich wybranych atrybutach. W rzeczywistości jednak zbiory danych posiadają obiekty o niekompletnym opisie, co jest często spotykanym zjawiskiem.

Brakujące wartości atrybutów to naturalna cecha przetwarzanych informacji. Przyczyn powstawania brakujących wartości może być wiele. Oto krótkie zestawienie niektórych z możliwych przyczyn występowania niekompletnego opisu obiektów:

- zaniedbania,
- zmiana zestawu atrybutów podczas procesu gromadzenia danych,
- dane pochodzą z różnych źródeł, posługujących się różnym zestawem atrybutów,
- brak danej własności spowodowany brakiem fizycznym, np. nie można rozpatrywać koloru samochodu klienta, gdy klient nie ma w ogóle żadnego samochodu,
- rzeczywisty brak danej własności, np. prezes nie ma zwierzchnika,
- wartość niemożliwa do uzyskania, np. pacjent nie może mieć wykonanego pewnego badania z powodu np. alergii,
- wartość wychodzi poza uprzednio zdefiniowaną dziedzinę lub zakres pomiarowy urządzenia, np. „kolor” podczerwony,
- pomiar niemożliwy do przeprowadzenia z powodu np. ograniczonej współbieżności urządzenia,
- błąd aparatury pomiarowej,
- ograniczenia fizyczne spowodowane np. zasadą Heisenberga.

Należy zauważyć, że zaniedbania, zmiana zestawu atrybutów i niejednorodne źródło pochodzenia danych to najczęstsze przyczyny powstawania danych o niekompletnym opisie obiektów.

Kolejną cechą charakteryzującą brakujące wartości atrybutów jest kwestia ich istnienia. Niektóre brakujące wartości atrybutów mogły by zostać poznane lub nawet zostały poznane i później zagubione. Wartości takie istnieją, lecz są przed nami ukryte. Inne brakujące wartości mogą faktycznie nie istnieć i wtedy charakteryzują się zupełnie innymi własnościami. Nie ma sensu np. mówić o uzupełnianiu takich wartości.

Brakujące wartości ponadto mogą być związane pewnymi więzami zależności. Mechanizm ich powstawania może być kompletnie losowy, lub mogą nim rządzić pewne, najczęściej ukryte, prawidłowości. W terminologii statystycznej używa się sformułowań zupełnie losowo brakujących wartości oraz wartości brakujących losowo, ale według pewnego rozkładu prawdopodobieństwa.

Z problemem brakujących wartości doskonale poradzono sobie w przypadku relacyjnych baz danych. Tam, gdzie nie interesuje nas inteligentne przetwarzanie informacji, a jedynie jej gromadzenie i możliwość przeprowadzania prostych operacji na danych, problem ten rozwiązano stosując trójwartościową logikę Łukasiewicza. Jest to mechanizm gwarantujący poprawne wykonywanie standardowych operacji na bazach danych. Niemniej jednak zapotrzebowanie inteligentnej analizy informacji jest daleko większe, niż rozwiązania zastosowane w relacyjnych bazach danych. Jak do tej pory nie wprowadzono tak powszechnie akceptowanych i gruntownie przebadanych rozwiązań dla problemu brakujących wartości, jak ma to miejsce np. wobec problemu informacji niepewnej i niedokładnej.

Zainteresowanie brakującymi wartościami atrybutów nie ogranicza się jednak tylko do praktycznych aspektów budowy skutecznych systemów decyzyjnych. Również na gruncie teorii maszynowego uczenia się podejmowano próby scharakteryzowania problemu brakujących wartości (patrz np. [4, 6, 17]). Jednym z najważniejszych na tym polu wyników jest pokazanie w pracy [6], że w ogóle można stosować uczenie się pojęć w oparciu o przykłady w stosunku do danych z niekompletnym opisem obiektów. Co prawda zaproponowany tam algorytm nie jest efektywny i posiada ponadwielomianową złożoność obliczeniową, jednak dzięki takim podstawom możemy mieć nadzieję, że można opracować skuteczny algorytm uczący się pojęć w oparciu o obiekty z brakującymi wartościami atrybutów.

1.6 Metody postępowania wobec brakujących wartości

Problemem brakujących wartości atrybutów w zakresie inteligentnego przetwarzania informacji zaczęto się poważnie interesować dopiero w drugiej połowie lat osiemdziesiątych. Wcześniej analogiczne problemy były badane na gruncie statystyki, algebry uniwersalnej i logiki, co stanowi inspirację dla większości używanych obecnie rozwiązań. Na tej podstawie wprowadzono wiele metodologii postępowania wobec brakujących wartości atrybutów, które można zaklasyfikować do czterech grup:

1. ignorowanie,
2. eliminacja obiektów lub atrybutów,
3. uzupełnianie brakujących wartości,
4. wnioskowanie bezpośrednio w oparciu o dane z niekompletnym opisem obiektów.

Najprostszymi i jednocześnie najbardziej zaburzającymi jakość wnioskowania metodami są ignorowanie i eliminacja. Pomimo ich oczywistych wad, metody te są niekiedy stosowane ze względu na ograniczenia już istniejących rozwiązań wnioskowania na podstawie danych.

Ignorowanie brakujących wartości to próba analizy danych z niekompletnym opisem obiektów w taki sposób, jakby były to normalne, dopuszczalne wartości. Jest to metoda częściowo stosowana do dzisiaj, gdyż wiele istniejących systemów analizy danych nie uwzględnia możliwości występowania brakujących wartości.

Alternatywną metodą do ignorowania jest eliminacja. Eliminować można obiekty o niekompletnym opisie lub atrybuty, dla których obiekty posiadają brakujące wartości. Usuwanie obiektów i/lub atrybutów niesie ze sobą niebezpieczeństwo utraty możliwości wykrycia ogólnej prawidłowości za pomocą wnioskowania indukcyjnego. Jednakże eliminacja dokonana przez specjalistę i poprzedzona dokładną analizą mechanizmów powstawania brakujących wartości i zależności pomiędzy atrybutami dla niektórych, szczególnych danych może przynieść zadowalający rezultat. Nie jest to jednak metoda uniwersalna, a w szczególności nie można jej ująć w sposób algorytmiczny, gdyż nieodzownym elementem sukcesu jest tutaj człowiek — doświadczony specjalista w zakresie analizy danych.

Uzupełnianie brakujących wartości to pierwsza z metodologii próbujących w sposób inteligentny poradzić sobie z problemem brakujących wartości, dająca się ująć algorytmicznie. Jej korzenie sięgają statystyki. Brakujące wartości usiłuje się uzupełniać za pomocą wartości z dziedziny atrybutów na podstawie mniej lub bardziej wyrafinowanego kryterium. Metoda ta może wprowadzać zaburzenia do danych, dlatego zakres jej zastosowań jest nieco ograniczony. Zaletą tej metody jest to, że dane uzupełniane są przed właściwym procesem wnioskowania i nie trzeba modyfikować istniejących algorytmów, które nie potrafią wnioskować w oparciu o dane z niekompletnym opisem obiektów.

Wnioskowanie bezpośrednio w oparciu o dane z niekompletnym opisem obiektów jest najbardziej uniwersalną metodologią postępowania wobec brakujących wartości. W odróżnieniu od wszystkich poprzednich metod, metoda ta umożliwia osiągnięcie najlepszych wyników. Uwarunkowane jest to jednak od powstania algorytmów, które będą możliwie w jak najbardziej efektywny sposób wykorzystywały zawartą w danych informację. Pewną wadą tej metodologii jest to, że jej adaptacja do już istniejących systemów wnioskowania w oparciu o dane wymaga modyfikacji istniejących algorytmów.

Zaprezentowana w rozdziale 7. metoda podziału usiłuje znaleźć kompromis pomiędzy eliminacją, uzupełnianiem i wnioskowaniem bezpośrednio w oparciu o dane z niekompletnym opisem obiektów w taki sposób, aby wyeliminować wyżej wspomniane wady tych rozwiązań.

Rozdział 2

Wstęp do teorii zbiorów przybliżonych

Przez wiedzę często rozumiemy zdolność do klasyfikacji, czyli umiejętności rozróżniania obiektów z otaczającej nas rzeczywistości. Można stwierdzić, że jednym z najważniejszych elementów wiedzy jest zdolność do klasyfikacji obiektów, przy czym przez obiekt rozumiemy wszystko, co tylko można sobie wyobrazić, np: przedmioty, zwierzęta, osoby, pojęcia abstrakcyjne, momenty czasu itd. Zatem chcąc zdefiniować wiedzę niezbędną do procesu wnioskowania, musimy najpierw zdecydować, jakimi obiektami jesteśmy zainteresowani. Zbiór takich obiektów nazwiemy *uniwersum*. Mając ustalone uniwersum możemy zdefiniować na nim rodziny podziałów, które dzielą nam uniwersum, zbiór wszystkich obiektów, na podzbiory. Podzbiory takie możemy nazywać pojęciami. Na przykład, jeśli za uniwersum przyjmiemy zbiór wszystkich *jabłek*, to możemy określić pojęcie *jabłka zielonego*. Pewne obiekty (jabłka) z uniwersum są reprezentantami pojęcia jabłka zielonego, czyli, co równoważne, należą do zbioru zielonych jabłek. Natomiast jeśli pewne jabłko nie jest zielone, należy do uzupełnienia zbioru zielonych jabłek. Z punktu widzenia danej własności obiektów (koloru jabłka), w oparciu o którą budujemy pojęcie, nie jesteśmy w stanie odróżnić między sobą obiektów należących do pojęcia, jak również obiektów do pojęcia nienależących. Z punktu widzenia koloru dany owoc albo jest zielony, albo taki nie jest i dalsze rozgraniczenie na podstawie takiej informacji pomiędzy reprezentantów zbioru zielonych jabłek nie jest możliwe. Ze względów praktycznych wygodnie jest również określać takie podziały nie tylko binarnie (jabłko zielone vs. pozostałe jabłka), ale na większą liczbę podzbiorów uniwersum. Na przykład ze względu na kolor jabłka można podzielić na zbiory jabłek zielonych, żółtych i czerwonych.

2.1 Reprezentacja wiedzy

Na początku lat 80-tych Profesor Zdzisław Pawlak zaproponował nowe podejście do problemu formalnego opisu wiedzy niepełnej i niedokładnej — teorię zbiorów przybliżonych (patrz [37]). Zaproponowane podejście stanowi dobrą podstawę teoretyczną do rozwiązywania problemów dotyczących inteligentnych systemów informacyjnych. Jak okaże się w następnym rozdziale, zbiory przybliżone okazały się użyteczne w szczególności przy analizie danych o brakujących wartościach atrybutów.

Teoria zbiorów przybliżonych jest doskonałą metodą starającą się naśladować naszkicowany powyżej model przetwarzania wiedzy. Jej główną zaletą jest formalne, logiczno-teoriomnogościowe ujęcie całokształtu zjawisk związanych z przetwarzaniem wiedzy i wnio-

skowaniem o obiektach. Również takie pojęcia jak nieprecyzyjność i niepewność danych, częstokroć reprezentowane numerycznie, przez co wymykają się stricte formalnemu podejściu, tutaj wyrażone są w postaci prostych do przyswojenia i analizy pojęć teoriomnogościowych.

Zdefiniujmy zatem formalnie nasz zbiór obiektów — uniwersum, wraz z pojęciami, które klasyfikują obiekty z uniwersum.

Definicja 2.1 *System informacyjny. (patrz [2, 38])*
System informacyjny to para $\mathbb{A} = (U, A)$, gdzie:

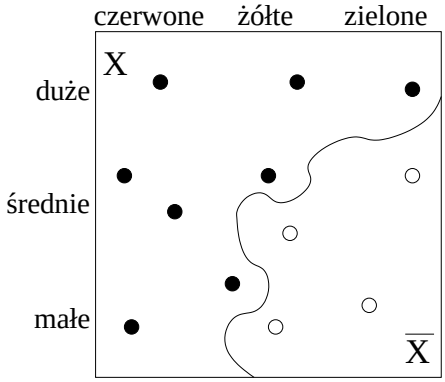
- *U jest skończonym, niepustym zbiorem, zwanym uniwersum. Elementy zbioru U nazywamy obiektami.*
- *A jest skończonym, niepustym zbiorem atrybutów, gdzie każdy atrybut $a \in A$ interpretowany jest jako funkcja $a : U \to V^a_{\mathbb{A}}$ przyporządkowującą obiektom z U wartości atrybutu a, przy czym $V^a_{\mathbb{A}}$ jest zbiorem wartości atrybutu a zwanym dziedziną atrybutu a[1].*

Zwyczajowo systemy informacyjne prezentuje się graficznie w postaci tabel informacyjnych. Postać tabeli jest tutaj szczególnie wygodna, gdyż stanowi podstawową strukturę danych używaną do implementacji systemów informacyjnych.

Przykład 2.1 *Jabłka.*
Niech $\mathbb{A} = (U, A)$, gdzie U to zbiór jabłek, a zbiór atrybutów A jest zdefiniowany jako $A = \{$kolor, wielkość, dojrzałe$\}$. Pojęcie jabłko zielone jest wyznaczone przez zbiór $Z \subset U$, taki, że $Z = \{x_i \in U : kolor(x_i) = zielone\}$. Możemy zobrazować przykładowy system informacyjny $\mathbb{A}$, gdzie $U = \{x_1, ..., x_8\}$, w postaci tabeli informacyjnej. Kolumny tabeli oznaczają atrybuty (cechy) badanego obiektu, a wiersze zawierają opis poszczególnych obiektów. Każda komórka tabeli w wierszu i i kolumnie a zawiera wartość $a(x_i)$, czyli klasyfikację o przynależności x_i do pewnego pojęcia, ze względu na atrybut (cechę) a.

	kolor	*wielkość*	*dojrzałe*
x_1	*czerwone*	*duże*	*tak*
x_2	*żółte*	*średnie*	*tak*
x_3	*zielone*	*małe*	*nie*
x_4	*zielone*	*duże*	*tak*
x_5	*żółte*	*średnie*	*nie*
x_6	*czerwone*	*średnie*	*tak*
x_7	*żółte*	*duże*	*tak*
x_8	*czerwone*	*średnie*	*tak*
x_9	*żółte*	*małe*	*nie*
x_{10}	*żółte*	*małe*	*tak*
x_{11}	*czerwone*	*małe*	*tak*
x_{12}	*zielone*	*średnie*	*nie*

[1] Gdy jasno wynika z kontekstu, jaki system informacyjny jest rozpatrywany, wtedy przyjmuje się również oznaczenie V_a.



Rysunek 2.1: Tak można wyobrażać sobie graficznie przestrzeń uniwersum U dla przykładu 1.1. Zbiór X reprezentuje pojęcie jabłka dojrzałego, a zbiór $\overline{X}$ — pojęcie przeciwne, jabłka niedojrzałego.

Pojęcie zielonego jabłka jest wyznaczane przez zbiór $Z = \{x_3, x_4, x_8, x_{12}\}$. Ponadto opis (klasyfikacja) pewnych obiektów względem atrybutów (własności) ze zbioru A jest identyczny, co zazwyczaj nie oznacza jeszcze, że są to dwa takie same jabłka, gdyż zestaw cech A jest dosyć ubogi.

2.2 Relacja nierozróżnialności

W powyższym przykładzie poruszyliśmy ważną własności cechującą systemy informacyjne. Ze względu na ograniczony charakter reprezentacji wiedzy w postaci praktycznie realizowalnych systemów informacyjnych należy wziąć pod uwagę, że wiedza w ten sposób zgromadzona będzie nieprecyzyjna. W teorii zbiorów przybliżonych modelowane jest to w sposób bezpośredni za pomocą relacji nierozróżnialności. Dwa obiekty (jak w powyższym przykładzie x_6 i x_8) mogą mieć taki sam opis cechami A, jednakże człowiek nie wyciąga z tego od razu wniosku, że są to dwa identyczne jabłka (lub wręcz, że jest to jedno i to samo jabłko), ale zakłada, że na obecnym stanie wiedzy nie jest w stanie ich od siebie rozróżnić.

Definicja 2.2 *Relacja nierozróżnialności*

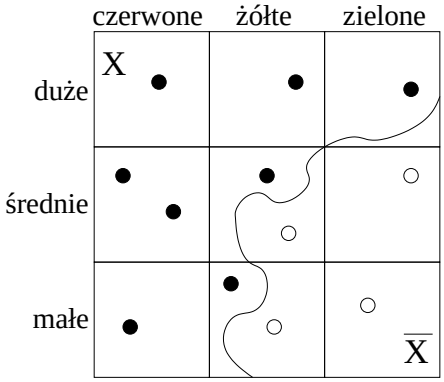
Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym i niech $B \subseteq A$. Relację nierozróżnialności $IND_{\mathbb{A}}(B) \subseteq U \times U$ generowaną przez zbiór B definiujemy w następujący sposób:

$$IND_{\mathbb{A}}(B) = \{(x, y) \in U \times U : \forall a \in B : a(x) = a(y)\}. \tag{2.1}$$

Relacja nierozróżnialności dzielni nam zbiór wszystkich obiektów na najmniejsze podzbiory, którymi możemy operować przy wykorzystaniu wiedzy B. Jeżeli nawet pewne dwa obiekty różnią się od siebie, ale przyjmują te same wartości na atrybutach z B, nie jesteśmy w stanie stwierdzić, czy są to dwa różne, czy jeden i ten sam obiekt, gdy opieramy się tylko na wiedzy o atrybutach (cechach obiektów) ze zbioru B.

Fakt 2.1 *Relacja nierozróżnialności spełnia następujące własności*

1. *$IND_{\mathbb{A}}(B)$ jest relacją równoważności,*



Rysunek 2.2: Klasy abstrakcji relacji nierozróżnialności $IND_{\mathbb{A}}(B)$, gdzie $B = \{\textit{kolor}, \textit{wielkość}\}$. W każdym kwadracie wszystkie obiekty są nierozróżnialne ze względu na opis B.

2. $B_1 \subseteq B_2 \Rightarrow IND_{\mathbb{A}}(B_2) \subseteq IND_{\mathbb{A}}(B_1)$

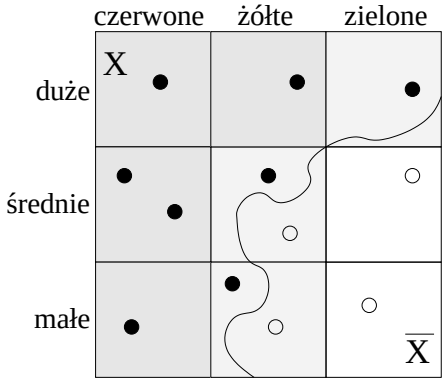
3. $IND_{\mathbb{A}}(B_1 \cup B_2) = IND_{\mathbb{A}}(B_1) \cap IND_{\mathbb{A}}(B_2)$

4. $IND_{\mathbb{A}}(B) = \bigcap_{a \in B} IND_{\mathbb{A}}(\{a\})$.

Powyższe własności wynikają z definicji relacji nierozróżnialności oraz z podstawowych faktów logiki i teorii mnogości. Fakt pierwszy mówi o tym, że relacja nierozróżnialności jest relacją równoważności, a co za tym idzie, dzieli całe uniwersum na klasy abstrakcji, które są rozłączne i niepuste. Fakt drugi ilustruje, że wiedza oparta na większej liczbie atrybutów daje nam większe możliwości rozróżniania obiektów między sobą. Fakt trzeci mówi o tym, że jeśli rozpatrzymy relację nierozróżnialności opartą na sumie dwóch podzbiorów A, to obiekty nie są przez nią rozróżniane tylko wtedy, gdy nie są rozróżniane przez żaden z tych podzbiorów. Wreszcie fakt czwarty, będący uogólnieniem poprzedniego faktu mówi o tym, że wszystkie klasy abstrakcji relacji nierozróżnialności powstają jako przecięcie klas nierozróżnialnych przez poszczególne atrybuty.

Pojedyncza klasa abstrakcji relacji nierozróżnialności jest najmniejszą jednostką, jaką możemy operować. Klasę abstrakcji nazywa się często pojęciem elementarnym lub pojęciem atomowym, gdyż jest najmniejszym podzbiorem uniwersum, jaki możemy sklasyfikować — odróżnić od pozostałych elementów za pomocą cech — atrybutów klasyfikujących obiekty do poszczególnych pojęć podstawowych.

Dane pochodzące z otaczającej nas rzeczywistości czasami nie pozwalają nam na jednoznaczne określenie, czy wartość atrybutu dwóch podanych obiektów jest sobie równa, czy też nie. Zjawisko takie może mieć miejsce przy badaniu identyczności kolorów, kształtów, głosów itd. Dlatego w niektórych zastosowaniach rozpatruje się nie system informacyjny, ale tak zwanym *system tolerancyjny*. W takim systemie relację nierozróżnialności, opartą na relacji równości, zastępuje się *relacją tolerancji*, rozumianą jako podobieństwo obiektów z uniwersum. Systemy tolerancyjne były rozpatrywane na przykład w pracy [39].



Rysunek 2.3: Górna i dolna aproksymacja zbioru.

2.3 Zbiory przybliżone

Celem wnioskowania na podstawie systemów informacyjnych jest próba klasyfikacji obiektów do pewnego pojęcia. Proces wnioskowania opiera się na opisie tego obiektu, wyrażonego w postaci innych pojęć — atrybutów zawartych w systemie informacyjnym. W naszym przypadku oznacza to, że próbujemy na podstawie przynależności obiektów do pewnych klas nierozróżnialności wnioskować o ich zaklasyfikowaniu jako należących do pewnego pojęcia lub nienależących.

Klasyczne podejście do systemów informacyjnych, stosujące standardową definicję teoriomnogościową zbioru (nazywaną też zbiorem „ostrym"), posiada dużo wad uniemożliwiających efektywne wnioskowanie na podstawie danych empirycznych. W ujęciu klasycznym pojęcie jest definiowalne w systemie informacyjnym (patrz [2, 38]), gdy za pomocą dostępnych pojęć możemy całkowicie wyznaczyć zbiór obiektów należących do tego pojęcia. Oznacza to, że pojęcia definiowalne, to tylko takie pojęcia, które możemy przedstawić jako suma pojęć atomowych w danym systemie informacyjnym. Wystarczy spojrzeć na rysunek 2.2, aby się przekonać, że zgodnie z tą definicją większość pojęć występujących w rzeczywistości nie jest definiowalna. Jest to spowodowane niedokładnością danych, co jest zjawiskiem nieuniknionym.

Teoria zbiorów przybliżonych oferuje nam mechanizm teoriomnogościowy pozwalający wyrazić w sposób ścisły i formalny rozumowania operujące na takich nieprecyzyjnych danych. Pomocne okażą się tutaj pojęcia aproksymacji (czyli przybliżenia) górnej i dolnej zbioru.

Definicja 2.3 *Aproksymacja zbioru.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym, $B \subseteq A$ będzie zbiorem atrybutów oraz $X \subseteq U$ będzie pewnym pojęciem, które chcemy aproksymować. Dla każdego obiektu $x \in U$, przez $[x]_{IND_{\mathbb{A}}(B)}$ oznaczmy klasę abstrakcji relacji $IND_{\mathbb{A}}(B)$ do której należy obiekt x.

1. *Dolną B-aproksymacją pojęcia X w systemie informacyjnym $\mathbb{A}$ nazywamy zbiór:*

$$B_{IND_{\mathbb{A}}}X = \{x \in U : [x]_{IND_{\mathbb{A}}(B)} \subseteq X\}. \tag{2.2}$$

2. *Górną B-aproksymacją pojęcia X zbiór:*

$$B^{IND_{\mathbb{A}}}BX = \{x \in U : [x]_{IND_{\mathbb{A}}(B)} \cap X \neq \emptyset\}. \tag{2.3}$$

3. *B-brzegiem pojęcia X w systemie informacyjnym $\mathbb{A}$ nazywamy zbiór:*

$$BN_B(X) = B^{IND_{\mathbb{A}}}X - B_{IND_{\mathbb{A}}}X \tag{2.4}$$

Za pomocą dolnej i górnej aproksymacji jesteśmy w stanie określić nieostre pojęcie X w ścisły sposób. Dolna aproksymacja pojęcia, to wszystkie te obiekty, które należą bez wątpienia do pojęcia. Należą one bowiem do takich klas abstrakcji, które w całości zawierają cię w pojęciu X. Górna aproksymacja pojęcia, to zbiór takich obiektów, co do których nie możemy wykluczyć, że należą do pojęcia. Jest to spowodowane tym, że należą do klas abstrakcji mających niepuste przecięcie z pojęciem X, a co za tym idzie, są nierozróżnialne z pewnym obiektem należącym do pojęcia X.

Fakt 2.2
Dolna i górna aproksymacja pojęcia spełnia następującą nierówność:

$$\emptyset \subseteq A_{IND_{\mathbb{A}}}X \subseteq X \subseteq A^{IND_{\mathbb{A}}} \subseteq U. \tag{2.5}$$

2.4 Definiowalność pojęć

Podstawowym zadaniem wnioskowania indukcyjnego jest wykrycie ogólnych prawidłowości pozwalających na klasyfikowanie obiektów do badanego pojęcia. Teoria zbiorów przybliżonych umożliwia analizę danych niepewnych i niedokładnych za pomocą pojęć aproksymacji dolnej i górnej. Rozszerza to istotnie klasę pojęć definiowalnych, czyli takich, co do których możemy oczekiwać, że wnioskowanie indukcyjne przyniesie oczekiwany rezultat.

Definicja 2.4 *Definiowalność pojęć.*

- *Pojęcie X jest całkowicie B-definiowalne, gdy $B_{IND_{\mathbb{A}}}X = B^{IND_{\mathbb{A}}}X$. Odpowiada to klasycznemu ujęciu definiowalności pojęć w systemach informacyjnych.*
- *Pojęcie X jest w przybliżeniu B-definiowalne, gdy $B_{IND_{\mathbb{A}}}X \neq \emptyset$ i $B^{IND_{\mathbb{A}}}X \neq U$.*
- *Pojęcie X jest wewnętrznie B-niedefiniowalne, gdy $B_{IND_{\mathbb{A}}}X = \emptyset$ i $B^{IND_{\mathbb{A}}}X \neq U$.*
- *Pojęcie X jest zewnętrznie B-niedefiniowalne, gdy $B_{IND_{\mathbb{A}}}X \neq \emptyset$ i $B^{IND_{\mathbb{A}}}X = U$.*
- *Pojęcie X jest całkowicie B-niedefiniowalne, gdy $B_{IND_{\mathbb{A}}}X = \emptyset$ i $B^{IND_{\mathbb{A}}}X = U$.*

Siła zbiorów przybliżonych przejawia się w tym, że, przy umiejętnym doborze rozpatrywanych atrybutów, praktycznie wszystkie interesujące nas pojęcia są w przybliżeniu definiowalne. Pozwala to na skuteczne wnioskowanie i formułowanie hipotez dotyczących aproksymowanych pojęć. Aby ocenić skuteczność aproksymacji wprowadza się współczynnik dokładności (ostrości) pojęcia.

Definicja 2.5 *Współczynnik dokładności pojęcia (patrz [2, 38]).*

Jeśli $\mathbb{A} = (U, A)$ jest systemem informacyjnym, $B \subseteq A$ oraz $X \subseteq U$ taki, że $X \neq \emptyset$, to miarę $\alpha_B(X) = \frac{|B_{IND_{\mathbb{A}}}X|}{|B^{IND_{\mathbb{A}}}X|}$ będziemy nazywać współczynnikiem dokładności (ostrości) pojęcia X w systemie informacyjnym $\mathbb{A}$, względem zbioru atrybutów B.

Współczynnik dokładności pojęcia ma następujące własności:

- $0 \leqslant \alpha_B(X) \leqslant 1$,
- jeśli $\alpha_B(X) = 1$, to pojęcie X jest całkowicie definiowalne, czyli ostre i jego własności mogą być w pełni wyrażone za pomocą zbioru atrybutów B,
- jeśli $\alpha_B(X) = 0$, to pojęcie X jest całkowicie niedefiniowalne (lub wewnętrznie niedefiniowalne) i jego własności nie mogą być wyrażone za pomocą zbioru atrybutów B,
- jeśli $0 < \alpha_B(X) < 1$, to pojęcie jest w przybliżeniu definiowalne (lub zewnętrznie niedefiniowalne) i jego własności mogą być częściowo wyrażone, z „mocą" współczynnika dokładności, przy pomocy atrybutów ze zbiory B.

Rodzaj definiowalności i współczynnik dokładności pojęcia pozwalają na charakteryzację dostępnych danych. Umożliwiają również wykrycie niecelowości stosowania pewnych danych do analizy. Jest to przydatne podczas fazy projektowania systemów gromadzenia danych i pozwala na sprawdzenie, czy w tabelach informacyjnych ujęto wszystkie atrybuty niezbędne do procesu wnioskowania.

2.5 Redukcja wiedzy

W podrozdziale 1.4 zaznaczono istnienie różnych problemów związanych z niedoskonałościami dostępnych danych. Jedną z nich jest tzw. szum informacyjny, czyli zbyt duża liczba nieistotnych informacji zawartych w opisach obiektów. Na gruncie teorii zbiorów przybliżonych również ten problem może zostać w naturalny sposób rozwiązany za pomocą tzw. reduktów.

Zdefiniujmy formalnie zbiór atrybutów, który składa się wyłącznie z istotnych atrybutów, wnoszących nową wiedzę na podstawie zawartej w nich informacji.

Definicja 2.6 *Niezależny zbiór atrybutów.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym. Zbiór atrybutów $B \subseteq A$ nazywamy niezależnym, gdy dla każdego atrybutu $a \in B$ zachodzi następujący warunek:

$$IND_{\mathbb{A}}(B) \neq IND_{\mathbb{A}}(B \setminus \{a\}) \tag{2.6}$$

Niezależny zbiór atrybutów to taki zbiór, z którego nie można usunąć żadnego atrybutu bez utraty cennych informacji, czyli zmniejszenia dokładności aproksymacji pojęcia. Dla każdego zbioru atrybutów możemy określić rodzinę zbiorów atrybutów, za pomocą których możemy uzyskać taką samą dokładność aproksymacji, oraz będących minimalnymi, w sensie relacji inkluzji, zbiorami atrybutów posiadających tę własność.

Definicja 2.7 *Redukt zbioru atrybutów (patrz [2, 38]).*

Jeśli $\mathbb{A} = (U, A)$ jest systemem informacyjnym, oraz $P, Q \subseteq A$, to zbiór atrybutów P jest reduktem zbioru atrybutów Q w systemie $\mathbb{A}$, gdy spełnione są następujące warunki:

- $IND_{\mathbb{A}}(P) = IND_{\mathbb{A}}(Q)$,
- *zbiór atrybutów P jest niezależny.*

Zbiór wszystkich reduktów zbioru atrybutów Q będziemy oznaczali przez $RED_{\mathbb{A}}(Q)$.

Dzięki relacji nierozróżnialności możemy w czytelny i formalny sposób wprowadzić pojęcie reduktu, pozwalające wyznaczyć istotny podzbiór informacji. Własności i metody generowania reduktów były szczegółowo badane np. w pracach [2, 50].

Eliminacja niepotrzebnej informacji spełnia kluczową rolę we wnioskowaniu indukcyjnym. Ponieważ wnioski formułowane są w oparciu o przykłady obiektów istnieje zagrożenie, że wnioski takie mogą być nadmiernie dopasowane do przykładów uczących i nie opisują w poprawny sposób ogólnych prawidłowości występujących w danych. Ograniczenie informacji tylko do podzbioru istotnych atrybutów umożliwia skuteczną walkę z tym tzw. problemem nadmiernego dopasowania. Istnieją również przesłanki statystyczne, jak zasada minimalnego opisu (ang. minimal description length, MDL), które wskazują na celowość posługiwania się reduktami, a nie pełnym zbiorem atrybutów. Stąd redukt to podstawowe narzędzie używane podczas procesu wnioskowania w oparciu o dane.

2.6 Wnioskowanie na podstawie danych

Celem uczenia się pojęć w oparciu o przykłady jest stworzenie opisu pojęcia, pozwalającego na klasyfikację obiektów z uniwersum pod względem przynależności do badanego pojęcia. Opis taki wyrażany jest w postaci formuł logicznych.

Definicja 2.8 *Formuła atomowa.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym. Formułą atomową nazwiemy każdy napis postaci (a, v), gdzie $a \in A$ oraz $v \in V_a$. Powiemy, że obiekt x spełnia formułę (a, v), gdy $a(x) = v$.

Definicja 2.9 *Formuła.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym. Do zbioru formuł $F(\mathbb{A})$ należą

- *wszystkie formuły atomowe,*
- *jeśli φ oraz ψ należą do zbioru formuł, to również $\neg\varphi$, $(\varphi \vee \psi)$, $(\varphi \wedge \psi)$ oraz $(\varphi \rightarrow \psi)$ należą do zbioru formuł.*

Symbole logiczne $\neg$, $\vee$, $\wedge$ oraz $\rightarrow$ należy traktować jako odpowiedniki znanych klasycznych funktorów.

Formuły $F(\mathbb{A})$ umożliwiają nam formalne ujęcie prawidłowości zachodzących w danych i wyrażenie ich w ścisły sposób. Opisy pojęć wyrażone są w postaci formuł szczególnego rodzaju, tzw. reguł decyzyjnych.

Definicja 2.10 *Reguła decyzyjna.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym. Regułą decyzyjną nazwiemy każdą formułę postaci $\varphi \rightarrow \psi$.

Do rozpoczęcia procesu wnioskowania indukcyjnego niezbędne jest wyznaczenie badanego pojęcia. Ponieważ, dla konkretnego systemu informacyjnego, badane pojęcie jest najczęściej trwale wyznaczone, wydziela się je ze zbioru atrybutów i nazywa atrybutem decyzyjnym. System informacyjny z wyznaczonym atrybutem decyzyjnym oznacza się $\mathbb{A} = (U, A \cup d)$, gdzie A nazywamy zbiorem atrybutów warunkowych, a d nazywamy atrybutem decyzyjnym.

Proste reguły decyzyjne, to formuły postaci $\varphi \rightarrow \psi$, które w części warunkowej (φ) zawierają formuły atomowe zbudowane w oparciu o atrybuty warunkowe, natomiast wniosek (ψ) jest formułą atomową postaci (d, v). Tak określone reguły decyzyjne znajdują się w centrum zainteresowania uczenia się pojęć w oparciu o przykłady.

Przykład 2.2

Niech $\mathbb{A}$ będzie systemem informacyjnym z przykładu 2.1. Możemy sformułować następujące reguły decyzyjne:

1. $(kolor, czerwone) \wedge (wielkość, duże) \rightarrow (dojżałe, tak)$
2. $(kolor, czerwone) \wedge (wielkość, duże) \rightarrow (dojżałe, nie)$
3. $(kolor, \text{żółte}) \wedge (wielkość, średnie) \rightarrow (dojżałe, tak)$
4. $(kolor, zielone) \wedge (wielkość, małe) \rightarrow (dojżałe, nie)$

Reguła 1. jest regułą prawdziwą w systemie $\mathbb{A}$, podczas gdy reguła 2. jest regułą fałszywą. Reguła 3. jest regułą aproksymacyjną, gdyż dotyczy klasy abstrakcji relacji nierozróżnialności należącej do górnej aproksymacji pojęcia X, ale nie należącej do dolnej aproksymacji tego pojęcia. Reguła 4. jest regułą dokładną, gdyż dotyczy klasy abstrakcji należącej do dolnej aproksymacji pojęcia $\overline{X}$.

2.7 Systemy decyzyjne

System potrafiący klasyfikować obiekty pod względem ich przynależności do pojęć nazwiemy systemem decyzyjnym. Zadaniem dla systemu decyzyjnego jest indukcja reguł decyzyjnych, czyli wnioskowanie indukcyjne w oparciu o dane, którego celem jest wygenerowanie opisu umożliwiającego klasyfikację obiektów. Stąd system decyzyjny nazywany jest również klasyfikatorem.

Najprostszy system decyzyjny jaki można sobie wyobrazić, to generator reguł decyzyjnych będących w istocie opisem wszystkich obiektów zawartych w tabeli informacyjnej. Zastosowanie teorii zbiorów przybliżonych umożliwia charakterystykę tych reguł jako prawdziwych lub nie, oraz aproksymacyjnych lub dokładnych. Istotnym ulepszeniem takiego algorytmu jest np. zastosowanie zredukowanych opisów obiektów, czyli zastosowania reduktów, jako podstawy do generowania reguł decyzyjnych. Metody konstruowania systemów decyzyjnych w ramach teorii zbiorów przybliżonych opisane są w pracach [2, 26, 32, 34, 35, 38, 48].

Reguły decyzyjne, czyli opis pojęcia, mogą być reprezentowane w różny sposób. Dwa najpopularniejsze sposoby, to reprezentacja reguł w naturalnej, formułowej postaci oraz reprezentacja w postaci drzew decyzyjnych. Drzewa decyzyjne zostały opisane na podstawie algorytmu C4.5 opisywanego w podrozdziale 4.1.

Rozdział 3

Rozszerzenia teorii zbiorów przybliżonych

3.1 Wprowadzenie

Teoria zbiorów przybliżonych oferuje skuteczny i efektywny mechanizm do przetwarzania wiedzy niepewnej i nieprecyzyjnej. Jednak usiłując przetwarzać konkretne dane częstokroć napotykamy na kolejny rodzaj niedoskonałości informacji, jakim są brakujące wartości atrybutów. Brak poszczególnych wartości w systemie informacyjnym stanowi przeszkodę w stosowaniu tradycyjnej teorii zbiorów przybliżonych. W ostatnich latach powstały jednak modyfikacje teorii zbiorów przybliżonych, które umożliwiają w naturalny i intuicyjny sposób przetwarzanie danych z brakującymi wartościami (patrz [21, 22, 27, 29, 49, 51, 53, 54, 56]).

W niniejszym rozdziale prezentowane będą modyfikacje relacji nierozróżnialności, które pozwalają na analizę danych z brakującymi wartościami. Ze względu na to, że prezentowane relacje częstokroć nie będą relacjami równoważności, pewnych drobnych modyfikacji wymagała będzie definicja górnej i dolnej aproksymacji pojęcia. Niemniej jednak zmiany te będą trywialne i będą służyły jedynie w celu ominięcia braku możliwości konstrukcji klas abstrakcji.

3.2 Tolerancja - Podobieństwo symetryczne

Problem nieokreślonych wartości nie jest w matematyce czymś nowym. Na gruncie algebry uniwersalnej (patrz np. [7, 18]) wykształcone zostało pojęcie algebry częściowej, gdzie operacje nie muszą być określone na całej dziedzinie, a tylko na jej części.

3.2.1 Podstawy algebraiczne

Algebra częściowa to pewne uogólnienie pojęcia algebry, nazywanej także algebrą totalną dla rozróżnienia tych dwóch pojęć. Pojęcie częściowości jest bardzo podobne do problemu brakujących wartości atrybutów [46]. Niektóre proste fakty z algebry uniwersalnej mogą być wprost przeniesione na grunt analizy danych z niekompletnym opisem obiektów.

Definicja 3.1 *Sygnatura (patrz [3]).*

Parę (F, η) nazywamy sygnaturą, jeśli F jest dowolnym zbiorem i $\eta : F \to \mathbb{N}$ jest funkcją. Jeśli para (F, η) jest sygnaturą to elementy zbioru F nazywamy symbolami operacji, a η funkcją arności. Jeżeli $\eta(f) = 0, 1, 2, n$ mówimy odpowiednio, że f jest symbolem stałej, operacji unarnej, binarnej lub n-argumentowej.

Definicja 3.2 *Algebra częściowa (patrz [3]).*

Parę $\underline{A} = (A, (f^{\underline{A}})_{f \in F})$ nazywamy algebrą częściową typu (F, η), jeśli A jest niepustym zbiorem zwanym nośnikiem algebry i dla każdego $f \in F$ $f^{\underline{A}}$ jest $\eta(f)$-arną operacją częściową w zbiorze A. Tzn. $f^{\underline{A}} : dom(f^{\underline{A}}) \to A$, gdzie $dom(f^{\underline{A}}) \subseteq A^{\eta(f^{\underline{A}})}$. Gdy $dom(f^{\underline{A}}) = A^{\eta(f^{\underline{A}})}$, wtedy f nazywamy operacją totalną.

Pojęcie algebry częściowej w bardzo naturalny sposób opisuje wiele zjawisk zachodzących w matematyce i w informatyce. Struktury częściowe pojawiają się zarówno przy problemach związanych z odejmowaniem w zbiorze liczb naturalnych, jak i operacjach na abstrakcyjnych typach danych, czy w abstrakcyjnej teorii algorytmów.

Podstawowym pojęciem łączącym algebrę uniwersalną z analizą danych o niekompletnym opisie obiektów jest pojęcie równości słabej.

Definicja 3.3 *Słaba równość (patrz [46]).*

Niech $\nu : X \to A$ będzie dowolnym wartościowaniem, gdzie X to zbiór zmiennych. Niech $\tilde{\nu} : dom(\tilde{\nu}) \to A$ będzie naturalnym rozszerzeniem ν nazywanym wartościowaniem termów. Algebra $\underline{A}$ spełnia słabą równość $p \overset{w}{\approx} q$, gdy zachodzi poniższy warunek.

$$p, q \in dom(\tilde{\nu}) \Rightarrow \tilde{\nu}(p) = \tilde{\nu}(q) \tag{3.1}$$

Słaba równość $p \overset{w}{\approx} q$ jest spełniona, gdy zachodzi równość funkcji indukowanych w $\underline{A}$ przez p i q określonych tylko na wspólnej dziedzinie p i q. Gdy p lub q jest nieokreślone, wtedy nie istotna jest wartość drugiego termu (odp. q lub p) i w szczególności może ona być również nieokreślona.

Dla odmiany aby zachodziła tzw. równość silna $p \overset{s}{\approx} q$ wymagane jest również, aby dziedziny określoności p i q były sobie równe. Koncepcja równości słabej jest istotnie różnym pojęciem od stosowanych do tej pory równości, odpowiadających raczej pojęciu równości silnej. Adaptacja unikalnego pomysłu, aby równość sprawdzać tylko na wspólnej poddziedzinie określoności, na grunt teorii zbiorów przybliżonych umożliwia wnioskowanie w oparciu o dane z niekompletnym opisem obiektów.

3.2.2 Relacja tolerancji

Relacja tolerancji — podobieństwa symetrycznego jest bardzo naturalnym rozszerzeniem relacji nierozróżnialności i była opisywana przez wielu badaczy zarówno na gruncie teorii zbiorów przybliżonych, jak i innych metod (patrz np. [25, 29, 39, 56]). Odpowiada ona pojęciu słabej równości z algebry uniwersalnej, jednak tutaj zyskuje ona dodatkową interpretację. W przypadku analizy danych można bowiem zakładać, że brakująca wartość danego atrybutu potencjalnie może być w rzeczywistości dowolnym elementem dziedziny tego atrybutu. Inaczej mówiąc, tabela którą dysponujemy jest niekompletnym, częściowym obrazem istniejącej tabeli z kompletnym opisem obiektów, która jest przed nami ukryta. Gdybyśmy

poznali w pełni uzupełnioną tabelę, to w miejscu brakujących wartości mogłyby stać dowolne wartości z dziedziny atrybutów. Ponieważ jednak nie znamy kompletnej tabeli w całości, to nie możemy stwierdzić, która z takich tabel w pełni uzupełnionych jest prawdziwym rozszerzeniem naszej wybrakowanej tabeli.

Definicja 3.4 *Relacja tolerancji*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym i niech $B \subseteq A$. Relację tolerancji (podobieństwa symetrycznego) $TOL_{\mathbb{A}}(B) \subseteq U \times U$ generowaną przez zbiór B definiujemy w następujący sposób:

$$TOL_{\mathbb{A}}(B) = \begin{array}{l}\{(x,y) \in U \times U : \\ \forall a \in B \;\; a(x) = a(y) \vee a(x) = * \vee a(y) = *\}.\end{array} \tag{3.2}$$

Należy zauważyć, że metoda uzupełniania wszystkimi możliwymi wartościami, badana nie tylko na gruncie zbiorów przybliżonych (np. [25]), jest równoważna zastosowaniu wyżej zdefiniowanej relacji tolerancji. Możemy wyobrażać sobie, że zastosowanie takiej relacji pozwala nam jednocześnie przetwarzać wszystkie możliwe rozszerzenia tabeli z brakującymi wartościami do tabeli w pełni uzupełnionej. Warto zauważyć, że liczba takich tabel jest wykładnicza ze względu na liczbę brakujących wartości. Oznacza to, że dla typowych tabel liczba takich rozszerzeń jest zazwyczaj większa od 2^{1000}, czyli większa od 300 cyfrowej liczby dziesiętnej. Widać tutaj wyraźnie przewagę teorii zbiorów przybliżonych, gdyż nie potrzebujemy tworzyć żadnych rozszerzeń fizycznie. Wystarczy zastosować tak zdefiniowaną relację tolerancji, aby uzyskać metodę równoważną do uzupełniania wszystkimi możliwymi wartościami.

Fakt 3.1 *Własności relacji tolerancji.*

1. *Relacja tolerancji jest zwrotna.*

$$\forall x \in U \;\; TOL_{\mathbb{A}}(B)(x,x)$$

2. *Relacja tolerancji jest symetryczna.*

$$\forall x,y \in U \;\; TOL_{\mathbb{A}}(B)(x,y) \Leftrightarrow TOL_{\mathbb{A}}(B)(y,x)$$

3. *Relacja tolerancji na ogół nie jest przechodnia.*

$$\forall x,y,z \in U \;\; TOL_{\mathbb{A}}(B)(x,y) \wedge TOL_{\mathbb{A}}(B)(y,z) \not\Rightarrow TOL_{\mathbb{A}}(B)(x,z)$$

Warunek przechodniości zachodzi wtedy i tylko wtedy, gdy obiekt y jest uzupełniony na wszystkich miejscach, gdzie żaden z obiektów x i z nie posiada brakującej wartości atrybutu (patrz [46]).

Relacja tolerancji nie jest relacją równoważności, nie pozwala nam zatem na konstrukcję klas abstrakcji. Definicja górnej i dolnej aproksymacji zbioru w oparciu o relację nierozróżnialności operowała na klasach abstrakcji, niemniej jednak zostały one użyte głównie dla ilustracji koncepcji pojęcia elementarnego i zwięzłości zapisu. Istotą aproksymacji dolnej jest to, że obiekt x należy z całą pewnością do pojęcia, gdy wszystkie obiekty z nim nierozróżnialne, czyli do niego podobne również należą do aproksymowanego pojęcia. Natomiast

obiekt x należy do aproksymacji górnej, gdy nie możemy wykluczyć, że któryś z obiektów z nim nierozróżnialnych (podobnych do niego) należy do badanego pojęcia. Zatem dolną i górną aproksymację zbioru — pojęcia możemy wyrazić bez potrzeby odwoływania się do klas abstrakcji.

Definicja 3.5 *Dolna i górna aproksymacja zbioru.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym, $B \subseteq A$ będzie zbiorem atrybutów oraz $X \subseteq U$ będzie pewnym pojęciem, które chcemy aproksymować.

1. *Dolną B-aproksymacją pojęcia X w systemie informacyjnym $\mathbb{A}$ nazywamy zbiór:*

$$\begin{aligned} B_{TOL_{\mathbb{A}}}X &= \{x \in U : \{y \in U : TOL_{\mathbb{A}}(B)(x,y)\} \subseteq X\} \\ &= \{x \in U : \forall y \in U \ \ TOL_{\mathbb{A}}(B)(x,y) \Rightarrow y \in X\}. \end{aligned} \tag{3.3}$$

2. *Górną B-aproksymacją pojęcia X zbiór:*

$$\begin{aligned} B^{TOL_{\mathbb{A}}}X &= \{x \in U : \{y \in U : TOL_{\mathbb{A}}(B)(x,y)\} \cap X \neq \emptyset\} \\ &= \{x \in U : \exists y \in U \ \ TOL_{\mathbb{A}}(B)(x,y) \wedge y \in X\}. \end{aligned} \tag{3.4}$$

Przykład 3.1

Dana jest następująca tabela decyzyjna $\mathbb{A} = (U, A \cup \{d\})$, gdzie $U = \{x_1, x_2, x_3, x_4\}$ oraz $A = \{a_1, a_2\}$. Dodatkowy atrybut decyzyjny, określający do którego pojęcia należy dany obiekt, oznaczymy przez d. W naszym przypadku U rozbija się na dwa pojęcia X *i* Y*, dlatego też dziedzina atrybutu decyzyjnego d jest określona $V_d = \{\mathrm{X}, \mathrm{Y}\}$.*

	a_1	a_2	d
x_1	1	2	X
x_2	$*$	2	X
x_3	1	$*$	Y
x_4	1	1	Y

Możemy wypisać zbiory elementów podobnych w sensie relacji $TOL_{\mathbb{A}}$: do x_1 podobne są x_2 oraz x_3, do x_2 podobne są x_1 oraz x_3, do x_3 podobne są x_1, x_2 i x_4, wreszcie do x_4 podobny jest x_3.

Aproksymacje pojęć X *i* Y *stanowią zbiory:*

- $A_{TOL_{\mathbb{A}}}\mathrm{X} = \emptyset$
- $A^{TOL_{\mathbb{A}}}\mathrm{X} = \{x_1, x_2, x_3, x_4\}$
- $A_{TOL_{\mathbb{A}}}\mathrm{Y} = \{x_4\}$
- $A^{TOL_{\mathbb{A}}}\mathrm{Y} = \{x_1, x_2, x_3, x_4\}$

Powyższy przykład ilustruje, że relacja tolerancji jest „ostrożna” w określaniu aproksymacji pojęć. Warto tutaj przypomnieć nierówność 2.5 opisującą własności górnej i dolnej aproksymacji dla relacji nierozróżnialności w kompletnych tabelach informacyjnych.

$$\emptyset \subseteq A_{IND_{\mathbb{A}}}X \subseteq X \subseteq A^{IND_{\mathbb{A}}} \subseteq U \tag{3.5}$$

Rozszerzając sens standardowej relacji nierozróżnialności na dane z niekompletnym opisem obiektów, w taki sposób, że brakująca wartość traktowana jest jak dopuszczalna wartość z dziedziny atrybutu, prawdziwy jest następujący fakt (patrz np. [56]).

Fakt 3.2

$$\emptyset \subseteq A_{TOL_{\mathbb{A}}}X \subseteq A_{IND_{\mathbb{A}}}X \subseteq X \subseteq A^{IND_{\mathbb{A}}}X \subseteq A^{TOL_{\mathbb{A}}}X \subseteq U \tag{3.6}$$

Oznacza to, że aproksymacje pojęcia generowane przez relację tolerancji są bardziej ogólne od aproksymacji generowanych przez relację nierozróżnialności.

Warto tutaj zauważyć, że aproksymacje generowane przez relację nierozróżnialności najbardziej przybliżają X w sensie powyższej nierówności. Wynika to wprost z wykorzystania wszystkich możliwych rozróżnień kombinacji wartości zapisanych w tabeli informacyjnej. Niestety takie aproksymacje w obliczu danych o niekompletnym opisie obiektów często prowadzą do nieprawdziwych wniosków.

Z drugiej strony relacja tolerancji jest najbardziej ogólną relacją, jest relacją „najbezpieczniejszą". Generowane przez nią aproksymacje są odpowiednio najmniejsze (największe) dla aproksymacji dolnych (górnych) wykorzystujących wiedzę B. Wszystkie inne relacje wprowadzane w niniejszym rozdziale zawsze ograniczone są przez relację nierozróżnialności i tolerancji, a ich aproksymacje mieszczą się pomiędzy tymi dwoma relacjami w sensie powyższej nierówności.

3.3 Podobieństwo niesymetryczne

W zastosowaniach praktycznych relacja podobieństwa symetrycznego — tolerancji najczęściej nie spełnia pokładanych w niej oczekiwań dobrego odpowiednika relacji nierozróżnialności. Generowane przez nią aproksymacje są zbyt ogólne, a liczba i sposób ułożenia brakujących wartości nie ma dużego wpływu na podobieństwo obiektów. Można powiedzieć, że relacja podobieństwa symetrycznego jest nazbyt „ostrożna", nawet w przypadkach, gdy można z całą pewnością wykluczyć przynależność poszczególnych przykładów do dolnej aproksymacji pojęcia.

Poszukiwania wielu badaczy lepszego zamiennika relacji nierozróżnialności, który pozwalał by na budowę efektywniejszych klasyfikatorów, zaowocowały alternatywnym rozwiązaniem w postaci relacji podobieństwa niesymetrycznego (patrz [20, 22, 52, 54, 55, 56]).

Definicja 3.6 *Relacja podobieństwa niesymetrycznego*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym i niech $B \subseteq A$. Relację podobieństwa niesymetrycznego $SIM_{\mathbb{A}}(B) \subseteq U \times U$ generowaną przez zbiór B definiujemy w następujący sposób:

$$\begin{aligned} SIM_{\mathbb{A}}(B) = \ & \{(x, y) \in U \times U : \\ & \forall a \in B \ \ a(x) = a(y) \vee a(x) = *\}. \end{aligned} \tag{3.7}$$

Relacja ta różni się w istotny sposób od relacji tolerancji. Pomysł wprowadzenia relacji podobieństwa niesymetrycznego może się wydawać nienaturalny, jednakże można go częściowo argumentować przykładem z [54]. Człowiek — ekspert w zakresie malarstwa nie używa sformułowania, że oryginał obrazu jest podobny do jego kopii. Tylko kopia może być podobna do oryginału, a nie na odwrotnie. W innych dziedzinach wiedzy również występują przypadki, gdy podobieństwo jest określane w sposób niesymetryczny.

Aby obiekt x był podobny do obiektu y musi zachodzić standardowy warunek równości wartości określonych atrybutów. Oprócz tego obiekt y musi być „oryginałem" dla obiektu

x, musi być określony na co najmniej tych samych atrybutach co obiekt x. W drugą stronę taki warunek nie jest konieczny i „kopia" x może posiadać więcej brakujących wartości atrybutów niż y. Tak zdefiniowana relacja w oczywisty sposób nie jest symetryczna. Łatwo jednak pokazać, że jest zwrotna i przechodnia.

Fakt 3.3 *Własności relacji podobieństwa niesymetrycznego.*

1. *Relacja podobieństwa niesymetrycznego jest zwrotna.*

$$\forall x \in U \;\; SIM_{\mathbb{A}}(B)(x,x)$$

2. *Relacja podobieństwa niesymetrycznego nie jest symetryczna.*

$$\forall x,y \in U \;\; SIM_{\mathbb{A}}(B)(x,y) \not\Leftrightarrow SIM_{\mathbb{A}}(B)(y,x)$$

3. *Relacja podobieństwa niesymetrycznego jest przechodnia.*

$$\forall x,y,z \in U \;\; SIM_{\mathbb{A}}(B)(x,y) \wedge SIM_{\mathbb{A}}(B)(y,z) \Rightarrow SIM_{\mathbb{A}}(B)(x,z)$$

Relacja podobieństwa niesymetrycznego nie jest oczywiście relacją równoważności, co uniemożliwia konstrukcję klas abstrakcji. Nie możemy zatem posługiwać się klasami abstrakcji w celu zdefiniowania górnej i dolnej aproksymacji pojęcia. Jako zamiennik klas abstrakcji możemy tutaj zastosować dwa zbiory obiektów podobnych, zbiór oryginałów do których obiekt x jest podobny, oraz zbiór kopii podobnych do obiektu x.

Definicja 3.7 *Zbiory obiektów podobnych.*

Każdemu obiektowi x przypiszemy dwa zbiory obiektów podobnych. Przez $R(x)$ oznaczymy zbiór obiektów podobnych do x, a przez $R^{-1}(x)$ oznaczymy zbiór obiektów do których x jest podobny i zdefiniujemy jak następuje:

$$R(x) = \{y \in U : (x,y) \in SIM_{\mathbb{A}}(B)\}, \tag{3.8}$$

$$R^{-1}(x) = \{y \in U : (y,x) \in SIM_{\mathbb{A}}(B)\}. \tag{3.9}$$

Zbiory obiektów podobnych umożliwią nam czytelną interpretację aproksymacji górnej i dolnej. Aproksymacja dolna pojęcia to zbiór obiektów na pewno do pojęcia należących. Aby to zagwarantować trzeba przyjąć, że obiekt x należy do dolnej aproksymacji tylko wtedy, gdy wszystkie obiekty do niego podobne (a zatem i on sam) należą do pojęcia. Do górnej aproksymacji pojęcia należą natomiast te obiekty, które są podobne do pewnego obiektu z badanego pojęcia. Wtedy nie możemy wykluczyć, że gdy poznamy więcej wartości badanego obiektu nie stanie się on identyczny z pewnym obiektem należącym do aproksymowanego zbioru.

Definicja 3.8 *Dolna i górna aproksymacja zbioru.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym, $B \subseteq A$ będzie zbiorem atrybutów oraz $X \subseteq U$ będzie pewnym pojęciem, które chcemy aproksymować.

1. *Dolną B-aproksymacją pojęcia X w systemie informacyjnym $\mathbb{A}$ nazywamy zbiór:*

$$\begin{aligned} B_{SIM_{\mathbb{A}}}X &= \{x \in U : R^{-1}(x) \subseteq X\} \\ &= \{x \in U : \{y \in U : SIM_{\mathbb{A}}(B)(y,x)\} \subseteq X\}. \end{aligned} \tag{3.10}$$

2. *Górną B-aproksymacją pojęcia X zbiór:*

$$\begin{aligned} B^{SIM_{\mathbb{A}}}X &= \bigcup\{R(y) : y \in X\} \\ &= \{x \in U : \exists y \in X \;\; y \in R(x)\} \\ &= \{x \in U : \{y \in U : SIM_{\mathbb{A}}(B)(x,y)\} \cap X \neq \emptyset\}. \end{aligned} \tag{3.11}$$

Tak zdefiniowana górna i dolna aproksymacja pojęcia różni się zdecydowanie od poprzednich aproksymacji względem relacji nierozróżnialności i tolerancji. Aproksymacje generowane przez relację podobieństwa niesymetrycznego najczęściej różnią się zdecydowanie od pozostałych.

Przykład 3.2

Kontynuując przykład 3.1 możemy wyznaczyć odpowiednie aproksymacje względem relacji podobieństwa niesymetrycznego. Na początek potrzebne będą nam zbiory elementów podobnych (zbiór oryginałów i kopii).

- $R(x_1) = \{x_1\}, R^{-1}(x_1) = \{x_1, x_2, x_3\}$
- $R(x_2) = \{x_1, x_2\}, R^{-1}(x_2) = \{x_2\}$
- $R(x_3) = \{x_1, x_3, x_4\}, R^{-1}(x_3) = \{x_3\}$
- $R(x_4) = \{x_4\}, R^{-1}(x_4) = \{x_3, x_4\}$

Możemy teraz łatwo wyznaczyć aproksymacje pojęć X *oraz* Y.

- $A_{TOL_{\mathbb{A}}}\mathrm{X} = \{x_2\}$
- $A^{TOL_{\mathbb{A}}}\mathrm{X} = \{x_1, x_2, x_3\}$
- $A_{TOL_{\mathbb{A}}}\mathrm{Y} = \{x_3, x_4\}$
- $A^{TOL_{\mathbb{A}}}\mathrm{Y} = \{x_3, x_4\}$

Własności aproksymacji względem relacji podobieństwa niesymetrycznego można scharakteryzować w sposób podobny do faktu 3.2. Zgodnie z oczekiwaniami, relacja podobieństwa niesymetrycznego mieści się pomiędzy relacją nierozróżnialności i relacją tolerancji.

Fakt 3.4

$$A_{TOL_{\mathbb{A}}}X \subseteq A_{SIM_{\mathbb{A}}}X \subseteq A_{IND_{\mathbb{A}}}X \subseteq X \subseteq A^{IND_{\mathbb{A}}}X \subseteq A^{SIM_{\mathbb{A}}}X \subseteq A^{TOL_{\mathbb{A}}}X \tag{3.12}$$

Aproksymacje, a co za tym idzie również i klasyfikacja oparta na tej relacji jest odmienna od pozostałych. Definiowalność pojęcia jest nieco bardziej szczegółowa niż dla relacji tolerancji oraz bardziej ogólna niż dla relacji nierozróżnialności zaadaptowanej do danych z niekompletnym opisem obiektów. Można powiedzieć, że tutaj wykorzystuje się więcej informacji ze zbioru danych (systemu informacyjnego), niemniej jednak może się to niekorzystnie odbić na poprawności rezultatów. To, czy wnioskowanie oparte o relację podobieństwa niesymetrycznego charakteryzują lepsze wyniki empiryczne zależy od przyjętej tabeli informacyjnej i musi być rozpatrywane indywidualnie.

3.4 Relacje parametryzowane

Relacje tolerancji i podobieństwa niesymetrycznego w ustalony sposób rozstrzygają o podobieństwie obiektów i definiują jednoznacznie aproksymacje górną i dolną obiektów. Jednakże dla szczególnych danych każda z tych relacji może się okazać niewłaściwa, czy to z powodu nazbyt ogólnej, czy też nieprawidłowej klasyfikacji. Właściwym zatem podejściem było by dopasowanie zamiennika relacji nierozróżnialności do konkretnych danych tak, aby klasyfikacja była poprawna i jednocześnie wystarczająco szczegółowa. Zaproponowane w pracach [19, 21, 53, 54, 55]. rozwiązanie tego zagadnienia opiera się na zastosowaniu rozmytych relacji podobieństwa.

Zbiory i relacje rozmyte

Zbiory rozmyte to pewne uogólnienie standardowego, teoriomnogościowego zbioru, gdzie zakładamy, że elementy mogą albo do zbioru należeć, albo nie należeć. Funkcja charakterystyczna takiego „ostrego" zbioru przyjmuje tylko wartości 0 lub 1.

$$\chi_Z : \mathbb{X} \to \{0, 1\} \tag{3.13}$$

Zbiory rozmyte dopuszczają dużo większą swobodę w określaniu przynależności elementów do zbioru, gdyż elementy mogą należeć do zbioru rozmytego w różnym stopniu. Funkcja charakterystyczna opisująca zbiór rozmyty może przybierać wszystkie wartości z przedziału $[0, 1]$.

$$\mu_Z : \mathbb{X} \to [0, 1] \tag{3.14}$$

Relację w standardowym, teoriomnogościowym podejściu definiuje się jako podzbiór iloczynu kartezjańskiego dziedzin argumentów. W przypadku relacji binarnej na U oznacza to, że relacja r to podzbiór $U \times U$. Utożsamiając relację z jej funkcją charakterystyczną, można powiedzieć, że

$$r : U \times U \to \{0, 1\}. \tag{3.15}$$

Relacja rozmyta, to uogólnienie standardowego pojęcia relacji. Tak jak standardowa relacja jest „ostrym" zbiorem elementów, tak relacja rozmyta jest zbiorem rozmytym. W naszym przypadku relacji binarnej na U oznacza to, że funkcja charakterystyczna relacji rozmytej jest określoną następująco:

$$r : U \times U \to [0, 1]. \tag{3.16}$$

Dzięki rozmytej relacji podobieństwa obiekty z uniwersum U mogą być podobne do siebie w pewnym stopniu, w przedziale $[0, 1]$. Daje to większą siłę wyrazu niż tylko rozgraniczenie na obiekty podobne i niepodobne.

Ponieważ zbiory rozmyte operują na wartościach liczbowych stopnia przynależności elementów, definiowane są za pomocą funkcji charakterystycznych. W istocie pojęcie zbioru rozmytego jest utożsamiane z rozmytą funkcją charakterystyczną i ilekroć operujemy zbiorach rozmytych, używamy do tego rozmytej funkcji charakterystycznej (patrz np. [11, 28]).

Relacje podobieństwa

Dotychczas rozpatrywane relacje podobieństwa, służące do wyznaczania górnej i dolnej aproksymacji pojęć, nie uwzględniały ważnego aspektu jakim jest stopień podobieństwa obiektów pomiędzy sobą.

Przykład 3.3

Dana jest następująca tabela decyzyjna $\mathbb{A} = (U, A \cup \{d\})$*, gdzie* $U = \{x_1, x_2, x_3\}$ *oraz* $A = \{a_1, a_2, a_3\}$.

	a_1	a_2	a_3	d
x_1	1	2	3	X
x_2	$*$	2	3	X
x_3	$*$	$*$	3	Y

Intuicyjnie obiekt x_2 *jest bardziej podobny do* x_1*, niż obiekt* x_3 *do* x_1*. Niemniej jednak zarówno relacja tolerancji, jak i podobieństwa niesymetrycznego, określa podobieństwo tych obiektów w taki sam sposób, nie pozwalający na zróżnicowanie stopnia podobieństwa.*

$$TOL_{\mathbb{A}}(A)(x_2, x_1),\ TOL_{\mathbb{A}}(A)(x_3, x_1),\ SIM_{\mathbb{A}}(A)(x_2, x_1),\ SIM_{\mathbb{A}}(A)(x_3, x_1)$$

Dysponując pojęciem relacji rozmytej w łatwy sposób możemy dobrać taką relację podobieństwa, która zróżnicuje nam stopień podobieństwa obiektów zgodnie z intuicją.

Przykład 3.4 *Rozmyta relacja podobieństwa.*

Najczęściej stosowana relacja podobieństwa rozmytego opiera się na interpretacji probabilistycznej brakujących wartości. Brakujące wartości mogą przybierać jedną z istniejących wartości atrybutu z jednakowym prawdopodobieństwem. Podobieństwo obiektów wzgl. jednego atrybutu można zatem zapisać wzorem:

$$R_{a_i}(x,y) = \begin{cases} 1 & a(x) = a(y) \neq * \\ 0 & a(x) \neq a(y) \wedge a(x) \neq * \wedge a(y) \neq * \\ \frac{1}{|V_{a_i}|} & a(x) = * \wedge a(y) \neq * \vee a(x) \neq * \wedge a(y) = * \\ \frac{1}{|V_{a_i}|^2} & a(x) = * \wedge a(y) = * \end{cases} . \tag{3.17}$$

Teraz możemy łatwo zapisać rozmytą relację podobieństwa, określoną na podzbiorze atrybutów B*,* $R_{\mathbb{A}}(B) : U \times U \to [0,1]$*:*

$$R_{\mathbb{A}}(B)(x,y) = \prod_{a \in B} R_a(x,y). \tag{3.18}$$

Tak zdefiniowana relacja podobieństwa odpowiada probabilistycznej interpretacji brakujących wartości, jako zdarzeń niezależnych ze schematu klasycznego. Ponadto ze względu na zaburzenia, jakie mogło by to wprowadzić do procesu wnioskowania, w literaturze przyjmowane jest niejawnie założenie, że $R(x,x) = 1$.

Weźmy tabelę informacyjną z poprzedniego przykładu (3.3). Przypuśćmy, że dla każdego a $V_a = \{1,2,3\}$*. Możemy zapisać rozmytą relację podobieństwa* $R_{\mathbb{A}}(A)$ *w postaci tablicy stopni przynależności.*

	x_1	x_2	x_3
x_1	1	$\frac{1}{3}$	$\frac{1}{9}$
x_2	$\frac{1}{3}$	1	$\frac{1}{27}$
x_2	$\frac{1}{9}$	$\frac{1}{27}$	1

Rozmyte aproksymacje pojęć

Mając zadaną rozmytą relację podobieństwa możemy przystąpić do definiowania aproksymacji górnej i dolnej, która w tym przypadku również będzie pojęciem rozmytym, określonym na rodzinie podzbiorów U.

Przekładając standardową definicję aproksymacji górnej i dolnej na język logiki rozmytej (patrz np. [53, 56]) uzyskujemy funkcję, która każdemu podzbiorowi U przypisuje stopień przynależności do aproksymacji.

Definicja 3.9 *Rozmyta aproksymacja dolna i górna*[1]

- *Rozmyta aproksymacja dolna pojęcia X to funkcja $\mu_{B_{R_{\mathbb{A}}}X} : \mathcal{P}(U) \to [0,1]$ taka, że*

$$\mu_{B_{R_{\mathbb{A}}}X}(Z) = T_{z\in Z}(T_{x\in U}(I(R_{\mathbb{A}}(z,x), \mu_X(x)))). \tag{3.19}$$

- *Rozmyta aproksymacja górna pojęcia X to funkcja $\mu_{B^{R_{\mathbb{A}}}X} : \mathcal{P}(U) \to [0,1]$ taka, że*

$$\mu_{B^{R_{\mathbb{A}}}X}(Z) = T_{z\in Z}(S_{x\in U}(T(R_{\mathbb{A}}(z,x), \mu_X(x)))). \tag{3.20}$$

Gdzie $\mu_X(x)$ jest stopniem w jakim obiekt x należy do pojęcia X (w przypadku niesprzecznej tabeli decyzyjnej funkcja ta przyjmuje wartości ze zbioru $\{0,1\}$), a T, S oraz I jest odpowiednio koniunkcją (T-normą), alternatywą (T-konormą,S-normą) oraz implikacją rozmytą (patrz np. [11, 28]).

Przykład 3.5

Kontynuując przykład 3.4 możemy użyć „probabilistycznych" operatorów rozmytych:

- $T(a,b) = a \cdot b$
- $S(a,b) = a + b - a \cdot b$
- $I(a,b) = 1 - a + a \cdot b$

Aproksymacja dolna i górna zdefiniowana jest wtedy następująco:

$$\mu_{B_{R_{\mathbb{A}}}X}(Z) = \prod_{z\in Z}\prod_{x\in U}(1 - R_{\mathbb{A}}(z,x) + R_{\mathbb{A}}(z,x)\cdot\mu_X(x)), \tag{3.21}$$

$$\mu_{B^{R_{\mathbb{A}}}X}(Z) = \prod_{z\in Z}(1 - \prod_{x\in U}(1 - R_{\mathbb{A}}(z,x)\cdot\mu_X(x))). \tag{3.22}$$

Stopień, w jakim pojedynczy obiekt $z \in U$ może stanowić dolne lub górne przybliżenie pojęcia X jest zdefiniowane następująco:

$$\mu_{B_{R_{\mathbb{A}}}X}(z) = \prod_{x\in U}(1 - R_{\mathbb{A}}(z,x) + R_{\mathbb{A}}(z,x)\cdot\mu_X(x)), \tag{3.23}$$

$$\mu_{B^{R_{\mathbb{A}}}X}(z) = 1 - \prod_{x\in U}(1 - R_{\mathbb{A}}(z,x)\cdot\mu_X(x)). \tag{3.24}$$

[1] Dzięki wykorzystaniu własności operatorów rozmytych w niniejszej definicji wyeliminowane zostało nie zawsze dobrze określone pojęcie klasy relacji (porównaj [53, 56]).

Rozmyte aproksymacje dolna i górna mogą być bezpośrednio użyte do indukcji reguł decyzyjnych (patrz [53, 56]). Regułom takim przypisuje się wtedy stopień zaufania będący w istocie stopniem, w jakim obiekty pasujące do reguły stanowią aproksymację dolną lub górną badanego pojęcia. Podczas procesu indukcji reguł mogą być generowane tylko reguły posiadające większy stopień zaufania niż pewna zadana wartość. Decydując się na zmniejszenie stopnia zaufania reguł możemy uzyskać więcej reguł, które dokładniej opisują badane pojęcie. Jednakże reguły o zbyt niskim stopniu zaufania mogą prowadzić do fałszywych wniosków.

3.5 Podsumowanie

Teoria zbiorów przybliżonych okazała się być bardzo użyteczna do analizy danych o niekompletnym opisie obiektów. Pojęcia aproksymacji zbiorów dają się łatwo zaadaptować do systemów informacyjnych z brakującymi wartościami atrybutów. System decyzyjny skonstruowany w oparciu o teorię zbiorów przybliżonych z powodzeniem można zastosować do takich danych.

Celem systemów decyzyjnych jest uzyskanie jak najlepszej klasyfikacji badanych obiektów. Przedstawione tutaj rozwiązania co prawda umożliwiają dokonanie analizy danych o niekompletnym opisie obiektów, jednakże posiadają również kilka słabych punktów.

Zaprezentowane relacje tolerancji i podobieństwa niesymetrycznego zakładają ustaloną semantykę brakujących wartości. Relacje te w stały sposób rozstrzygają, czy obiekty są do siebie podobne, czy też nie. Jednakże, wśród danych pochodzących z rzeczywistości, często można natrafić na takie, w których mechanizmy rządzące powstawaniem i znaczeniem brakujących wartości są skomplikowane i nie przystają do ustalonego schematu ich porównywania. Co prawda relacja tolerancji gwarantuje nam maksymalną poprawność wyciąganych wniosków, jednak może się okazać, że dysponując dodatkową wiedzą można w sposób bezpieczny uzyskać dokładniejsze aproksymacje pojęć. Klasyfikatory oparte o relacje tolerancji i podobieństwa niesymetrycznego mogą być nieelastyczne i uzyskiwać nie najlepsze wyniki.

Pewnym rozwiązaniem jest tutaj parametryzowana relacja podobieństwa. Za pomocą funkcji określającej stopień podobieństwa obiektów pomiędzy sobą można podjąć próbę uwzględnienia nawet skomplikowanych mechanizmów rządzących brakującymi wartościami. Jednakże proces doboru takiej funkcji jest bardzo skomplikowany. Usiłując wyznaczyć optymalną funkcję a priori musimy dysponować dużą wiedzą na temat przetwarzanych danych oraz musimy również umieć zawrzeć tę wiedzę w postaci funkcji podobieństwa obiektów. Gdy podejmujemy próbę automatycznego wyznaczenia optymalnej funkcji podobieństwa spośród pewnej klasy funkcji stajemy przed problemem bardzo czasochłonnego problemu optymalizacyjnego. Wszystko to sprawia, że chociaż teoretycznie dysponujemy możliwością wyznaczenia relacji podobieństwa dopasowanej do przetwarzanych danych, to rozwiązanie takie jest niepraktyczne. Należy jednak zauważyć, że dla pewnych obszarów zastosowań może być to rozwiązanie w pełni akceptowalne i bardzo skuteczne.

Idealnym rozwiązaniem było by opracowanie takiej relacji podobieństwa, która mogła by zostać wyznaczona na podstawie danych. Podobnie jak uczymy się pojęć w oparciu o przykłady, mogli byśmy również podjąć próbę wyuczenia się relacji podobieństwa obiektów, która uchwyci wszystkie zawiłości związane z brakującymi wartościami obiektów. Niestety, jak do tej pory nie znaleziono rozwiązania dla tego problemu. Wiele przesłanek wskazuje, że

rozwiązanie takie nie może opierać się na numerycznym wyznaczaniu podobieństwa obiektów, jak ma to miejsce w przypadku parametryzowanych relacji podobieństwa, a powinno operować jedynie pojęciami teoriomnogościowymi, podobnie jak sama teoria zbiorów przybliżonych. Takie „symboliczne” (w przeciwieństwie do numerycznego) rozwiązanie było by wielkim zwycięstwem teorii zbiorów przybliżonych nad danymi o niekompletnym opisie obiektów. Pytanie w jaki sposób konstruować relacje podobieństwa na podstawie danych pozostaje jednak otwarte.

Rozdział 4

Metody wnioskowania bezpośredniego

Zadaniem tego rozdziału jest opisanie metod nie wywodzących się z nurtu teorii zbiorów przybliżonych, które potrafią wnioskować w oparciu o dane z niekompletnym opisem obiektów bez potrzeby modyfikowania danych wejściowych. W odróżnieniu od metod leniwych opisywanych w następnym rozdziale, tutaj celem każdej metody jest konstrukcja pewnej hipotezy opisującej pojęcie.

Istnieje wiele metod wnioskowania indukcyjnego, które mają niewiele wspólnego z teorią zbiorów przybliżonych. Ze względu na zapotrzebowanie na metody potrafiące radzić sobie z brakującymi wartościami również na tym gruncie dopracowano się metod, które nie modyfikują danych z niekompletnym opisem obiektów, a wnioskują na nich w sposób bezpośredni. Porównanie w jaki sposób udaje im się uniknąć problemu niekompletnego opisu obiektów może być bardzo kształcące. W szczególności zaprezentowane w rozdziale 7 wyniki eksperymentalne stanowią porównanie metody podziału z algorytmem C4.5 opisywanego w niniejszym rozdziale.

4.1 C4.5

Metoda C4.5 wymyślona przez Quinlana to chyba jedna z najbardziej popularnych metod wnioskowania indukcyjnego. Jej główna idea opiera się na schemacie zstępującej indukcji drzewa decyzyjnego na podstawie danych treningowych. Za pomocą zbudowanego drzewa decyzyjnego możemy klasyfikować obiekty ze zbioru testowego. Metoda cechuje się wysoką jakością klasyfikacji oraz dużą sprawnością w radzeniu sobie z brakującymi wartościami.

Metody klasyfikacji w oparciu o indukcję drzew decyzyjnych swoimi korzeniami sięgają lat sześćdziesiątych i pierwotnie rozpatrywane były w ujęciu statystycznym. Na grunt maszynowego uczenia się we współczesnej postaci drzewa decyzyjne wprowadził Quinlan, który przyjął odmienną od statystyków perspektywę i terminologię, a także wprowadził teorioinformacyjne kryteria oceny testów oraz techniki przycinania. Rozwijany przez niego system, nazywany w kolejnych wersjach ID3, C4 i C4.5, stanowi punkt odniesienia dla sporej części badań nie tylko nad algorytmami konstruowania drzew decyzyjnych, lecz uczenia się pojęć w ogólności.

W tym podrozdziale ograniczymy się do ogólnego opisu metod bazujących na drzewach decyzyjnych, bez wdawania się w szczegóły implementacyjne metody C4.5. Pierwotny schemat zstępującej konstrukcji drzewa przewija się praktycznie bez modyfikacji w każdej metodzie bazującej na drzewach decyzyjnych. Jedynie rozwiązanie problemu brakujących warto-

ści jest na tyle szczególne dla metody C4.5, że poświęcimy mu więcej uwagi. Metoda C4.5 obfituje w różnorakie ulepszenia prostego schematu budowy drzewa, które zostały szczegółowo opisane w książce [42], a jej kod źródłowy jest ogólno dostępny w internecie.

4.1.1 Drzewa decyzyjne

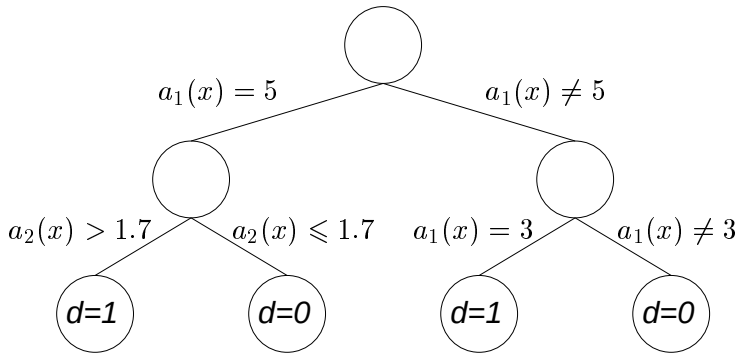
Drzewo decyzyjne, to struktura umożliwiająca klasyfikację obiektów. Składa się ona z wierzchołków połączonych etykietowanymi krawędziami. Każdy obiekt podlegający klasyfikacji rozpoczyna swoją ścieżkę klasyfikacji w korzeniu drzewa, a kończy ją w liściu drzewa. Krawędzie drzewa są etykietowane testami, czyli prostymi formułami logicznymi, które decydują do którego z synów zostanie przesłany obiekt w celu dalszej klasyfikacji. Testy te są rozłączne i pełne w taki sposób, że dla każdego obiektu istnieje jedna, jednoznacznie wyznaczona ścieżka klasyfikacji[1]. Liście natomiast, mają przypisaną klasę decyzyjną, do której należą, lub powinny należeć wszystkie obiekty, których ścieżki klasyfikacji kończą w tym liściu. Gdy obiekt kończy swoją ścieżkę klasyfikacji w danym liściu, mówi się również, że obiekt został zaklasyfikowany do tego liścia.

Drzewo decyzyjne konstruowane jest w oparciu o dwie podstawowe zasady. Pierwszą z nich, jest założenie, aby klasyfikacja uzyskana za pomocą drzewa decyzyjnego posiadała jak najmniejszy błąd (liczbę złych odpowiedzi) na danych treningowych. Ponieważ jednak takie działanie może prowadzić do zjawiska przeuczenia należy uzyskać pewien kompromis pomiędzy współczynnikiem błędu a stopniem skomplikowania hipotezy, czyli wielkością drzewa. Ma to swoje uzasadnienie w zasadzie minimalnego opisu (ang. minimal description length, MDL) (patrz np. [44, 42]). Zasada ta jest również przesłanką do stosowania metod minimalizacji złożoności informacyjnej podzbiorów obiektów, rozdzielanych za pomocą testów na krawędziach drzewa.

Jeśli obiekty ze zbioru treningowego zaklasyfikowane do pewnego liścia należą do różnych klas decyzyjnych, wtedy zbiór zaklasyfikowanych do niego obiektów jest niejednorodny, a liść taki nazywamy niejednorodnym. Gdy wszystkie obiekty treningowe zaklasyfikowane do danego liścia należą do tej samej klasy decyzyjnej, liść taki jest jednorodny lub inaczej „czysty”. Ponieważ w dane pochodzące z rzeczywistości mogą być, i często są sprzeczne (a także ze względu na stosowanie metod przycinania), liściom nie koniecznie muszą odpowiadać obiekty z jednej klasy decyzyjnej. Klasyfikowanym obiektom testowym, które trafiają do niejednorodnego („brudnego”) liścia przypisuje się najczęściej pojedynczą decyzję wybraną przez głosowanie większościowe spośród obiektów treningowych zaklasyfikowanych do tego liścia. Inną koncepcją jest przypisywanie wszystkich decyzji, razem z ich prawdopodobieństwem empirycznym, wyznaczonym na podstawie zaklasyfikowanych do tego liścia obiektów treningowych.

Proces konstrukcji drzewa decyzyjnego przebiega iteracyjnie. Początkowo wszystkie obiekty przypisane są do jednego wierzchołka będącego zarazem korzeniem i liściem. Określa się również warunek stopu, ustanawiający kompromis pomiędzy współczynnikiem błędu a wielkością drzewa. W pętli powtarzany jest proces wyboru liścia. Najczęściej jest to kolejny niejednorodny liść lub liść o najbardziej niejednorodnym zbiorze zaklasyfikowanych obiektów. Zbiór ten usiłuje się rozdzielić za pomocą testu na podzbiory obiektów o jak najmniejszej złożoności informacyjnej. Idealną sytuacją było by rozdzielić zbiór obiektów

[1] Przynajmniej dla danych o kompletnym opisie obiektów.



Rysunek 4.1: Proste drzewo decyzyjne

zaklasyfikowanych do takiego wierzchołka na podzbiory jednorodne. Wybór testów minimalizujących złożoność informacyjną, lub inaczej, maksymalizujących zysk informacyjny, to heurystyczna strategia postępowania, mająca zagwarantować jak najmniejszą złożoność drzewa (liczbę testów i wierzchołków). Postępowanie takie jest motywowane chęcią wygenerowania minimalnego opis hipotezy, zgodnie z zasadą MDL. Jako miarę złożoności informacyjnej zbiorów stosuje się takie funkcje jak entropia, rozróżnialność, Gini index czy test χ^2. Po wyborze optymalnego testu tworzy się nowe wierzchołki (najczęściej dwa), będące synami rozbijanego liścia. Krawędzie prowadzące do nowo utworzonych wierzchołków etykietuje się wybranym testem i jego negacją (lub wybranymi testami, gdy dopuszczamy rozbicia na więcej niż dwa podzbiory). Proces zostaje zakończony, gdy wszystkie liście są wystarczająco jednorodne aby umożliwić skuteczną klasyfikację.

Testy obiektów którymi etykietowane są krawędzie drzewa decyzyjnego rozdzielają obiekty do synów wierzchołka na podstawie wartości atrybutów obiektu. Najprostsze testy, stosowane w metodzie C4.5, opierają się na badaniu wartości jednego atrybutu. Dla atrybutów symbolicznych sprawdza się, czy atrybut na danym obiekcie przyjmuje pewną wartość. Testy tej postaci możemy zapisać jako $a_i(x) = v$, gdzie $v \in V_{a_i}$, oraz x odpowiada testowanemu atrybutowi. Dla atrybutów numerycznych możemy korzystać z liniowego uporządkowania dziedziny atrybutu. Testy dla takich atrybutów mogą mieć postać $a_i(x) < v$. W przypadku gdy obiekt spełnia dany test, przechodzi do odpowiadającego mu syna tego wierzchołka.

Przejście przez obiekt ścieżki klasyfikacji od korzenia do liścia jednoznacznie wyznacza spełnione przez niego testy. Możemy to zapisać w postaci formalnej za pomocą koniunkcji testów, uzyskujemy wtedy w naturalny sposób reguły decyzyjne, opisywane również w podrozdziale 2.6[2].

Przykład 4.1

Reguły decyzyjne dla drzewa z rysunku 4.1 wyglądają następująco:

- $(a_1(x) = 5) \wedge (a_2(x) > 1.7) \Rightarrow (d(x) = 1)$
- $(a_1(x) = 5) \wedge (a_2(x) \leqslant 1.7) \Rightarrow (d(x) = 0)$
- $(a_1(x) \neq 5) \wedge (a_1(x) = 3) \Rightarrow (d(x) = 1)$
- $(a_1(x) \neq 5) \wedge (a_1(x) \neq 3) \Rightarrow (d(x) = 0)$

[2]Tutaj stosujemy nieco bogatszy język do zapisu formuł atmomowych.

Krawędzie mogą być również etykietowane bardziej skomplikowanymi testami. W wierzchołku można sprawdzać jednocześnie wartości wielu atrybutów. W przypadku atrybutów numerycznych oznacza to cięcie przestrzeni obiektów za pomocą hiperpłaszczyzn (patrz np. [32]). Ponadto można konstruować nie tylko dwa wykluczające się testy, ale ich większą liczbę. Na przykład, można skonstruować po jednym teście dla każdej wartości atrybutu (symbolicznego). Podejście takie stosowane począwszy od algorytmu ID3 opisanego w [40].

Raz utworzone drzewo decyzyjne może być wielokrotnie wykorzystywane do klasyfikacji obiektów testowych, inaczej niż ma to miejsce w metodzie LazyDT opisywanej w podrozdziale 5.2. Proces klasyfikacji obiektu jest szybki i polega na znalezieniu takiej ścieżki w drzewie, że obiekt spełnia testy wszystkich krawędzi tej ścieżki.

4.1.2 Brakujące wartości

Gdy usiłujemy przetwarzać dane o niekompletnym opisie obiektów za pomocą metod opartych na drzewach decyzyjnych napotykamy na kilka trudności.

- Wybór testu, za pomocą którego dzielimy obiekty, jest dokonywany na podstawie heurystycznego kryterium jakim jest zysk informacyjny. Jeśli dwa testy używają różnej liczby obiektów o brakującej wartości atrybutu, jak powinno być to uwzględniane podczas porównywania ich przydatności?

- Gdy test zostanie już wybrany, obiekty z brakującą wartością testowanego atrybutu nie mogą być zaklasyfikowane do żadnego z potomków. Jak powinny być traktowane takie obiekty podczas rozdzielania?

- Kiedy drzewo decyzyjne używane jest do klasyfikacji nowych, testowych obiektów, jak powinno się postąpić, gdy obiekt posiada brakującą wartość testowanego atrybutu?

Na podstawie badań opisanych w pracy [41] wybrana została strategia postępowania, która co prawda nie uzyskuje najlepszych wyników dla wszystkich danych eksperymentalnych, ale średnio przewyższa swoją skutecznością inne podejścia. Metoda ta została szczegółowo opisana w książce [42]. Ponadto w pracach [30, 36, 58] rozważano słuszność przyjętego przez Quinlana podejścia i proponowano pewne ulepszenia zarówno procesu indukcji drzewa, jak i np. przycinania drzew decyzyjnych.

Podejście zastosowane w algorytmie C4.5 opiera się na empirycznym rozkładzie prawdopodobieństwa z jakim obiekty o znanych wartościach atrybutów spełniają rozważane testy. Modyfikacja kryterium wyboru testu została wyprowadzona z interpretacji znaczenia informacji. Zysk informacyjny, jako funkcja podlegająca maksymalizacji przez wybór optymalnego testu, powinien zostać tak przeliczony, aby uwzględniał obiekty z brakującymi wartościami atrybutów. Ponieważ informacja pozwalająca zaklasyfikować te obiekty do któregoś z podzbiorów nie jest znana, dlatego na tych obiektach zysk informacyjny powinien wynosić zero. Oznacza to, że zysk informacyjny powinien zostać zmodyfikowany o współczynnik częstości występowania obiektów bez brakujących wartości obiektów. Odbywa się to według wzoru:

$$gain(X)' := F \cdot gain(X), \tag{4.1}$$

gdzie $F = \frac{liczba\ obiektów\ bez\ brakujących\ wartości}{liczba\ obiektów}$.

Po wyborze testu musimy rozdzielić obiekty do podzbiorów, tak aby spełniały ustalone testy. Jednakże obiekty o nieznanej wartości testowanego atrybutu nie mogą być zaklasyfikowane do żadnego z podzbiorów. Metoda zaproponowana przez Quinlana polega na zastosowaniu obiektów ważonych i dystrybucji obiektów z brakującymi wartościami atrybutów do wszystkich podzbiorów jednocześnie. Przypuśćmy, że zbiór obiektów O za pomocą n testów dzielimy na podzbiory $O_1, \ldots, O_n$. Obiekty, które mają brakującą wartość testowanego atrybutu przypisywane są do zbioru O_i z wagą równą $\frac{|O_i|}{|O|}$. Oznacza to, że obiekty takie są rozdzielane do wszystkich podzbiorów zgodnie z empirycznym prawdopodobieństwem takiego zdarzenia. Komplikacji musi ulec algorytm, gdyż teraz musimy operować nie „całymi" obiektami, ale również „częściami" obiektów. Uzyskuje się to przez zastosowanie wag z zakresu $[0, 1]$.

Podobne podejście zastosowane zostało podczas klasyfikacji obiektów testowych. Tutaj również obiekty o nieznanej wartości atrybutu rozdzielane są po wszystkich krawędziach drzewa decyzyjnego z wagami z zakresu $[0, 1]$. Nie możemy zatem mówić o pojedynczej ścieżce klasyfikacji, gdyż obiekt może teraz posiadać wiele ścieżek klasyfikacji. Wszystkie odpowiedzi (tzn. decyzje pochodzące z liści) sumowane są z wagami, z jakimi obiekt został zaklasyfikowany do danego liścia. W ten sposób uzyskuje się nie pojedynczą klasyfikację do klasy decyzyjnej, ale klasyfikację do wielu klas decyzyjnych wraz z prawdopodobieństwami przynależności do danej klasy decyzyjnej. Na tej podstawie dokonuje się ostatecznej klasyfikacji za pomocą głosowania.

4.2 LRI

Zaproponowana w przez Weissa i Indurkhya metoda LRI (Lightweight Rule Induction) prezentuje nieco odmienne podejście do indukcji reguł decyzyjnych. W odróżnieniu od metod takich jak C4.5, gdzie reguły budowane są na podstawie wyindukowanego drzewa decyzyjnego, tutaj reguły decyzyjne indukowane są z danych bezpośrednio. Różnic pomiędzy takimi podejściami jest wiele. Chyba najważniejszą z nich jest to, że reguły powstałe z drzewa decyzyjnego są wzajemnie wykluczające się, podczas gdy reguły wyindukowane w sposób bezpośredni nie muszą spełniać takiego wymagania. Metody bezpośredniej indukcji reguł stanowią drugą, najbardziej popularną po drzewach decyzyjnych grupę algorytmów uczenia się pojęć.

4.2.1 Indukcja reguł decyzyjnych

Reguła to najczęściej koniunkcja prostych testów, podobnie jak miało to miejsce w przykładzie 4.1. Mówimy, że reguła pokrywa obiekt, gdy obiekt spełnia warunkową część reguły.

Standardowa metoda indukowania reguł decyzyjnych opiera się na konstrukcji zbioru reguł pokrywającego dane treningowe. Zazwyczaj proces indukcji przebiega iteracyjnie. Indukowana jest reguła, pokrywająca możliwie wiele obiektów i poprawiająca jakość klasyfikacji, a następnie obiekty pokryte przez regułę są usuwane ze zbioru treningowego i proces jest powtarzany, dopóki zbiór obiektów treningowych nie został wyczerpany. Proces generowania pojedynczej reguły polega na iteracyjnym dodawaniu testów (formuł atomowych) maksymalizujących jakość klasyfikacji. Warunkiem stopu jest tutaj osiągnięcie określonej długości reguły. Gdy reguła składa się z zadanej liczby formuł atomowych algorytm prze-

chodzi do konstruowania następnej reguły, aż do momentu, w którym wszystkie obiekty ze zbioru treningowego są prawidłowo klasyfikowane przez wygenerowany zbiór reguł.

W metodzie LRI rozszerza się nieznacznie standardowy model reguły decyzyjnej, umożliwiając połączenie kilku reguł w postaci koniunkcyjnej w jedną regułę w postaci DNF, o ile tylko reguły dotyczyły tej samej klasy decyzyjnej. Rozwiązanie zadania klasyfikacji składa się ze zbioru równej liczby nieważonych reguł dla każdej klasy decyzyjnej. Nowy przykład jest klasyfikowany do pewnej klasy decyzyjnej przez głosowanie proste, czyli do klasy wskazanej przez największą liczbę aktywnych[3] reguł.

Kolejną modyfikacją zastosowaną w metodzie LRI jest adaptacyjny system ważenia obiektów. Ma to na celu wygenerowanie zbioru reguł jak najlepiej określających badane pojęcie. System ten jest szczegółowo opisany w pracach [59, 60]. Podobny system próbowano zastosować do procesu generowania wzorców w metodzie podziału, jednakże wyniki eksperymentalne nie potwierdziły jego skuteczności przy rozwiązywaniu tego problemu.

4.2.2 Brakujące wartości

W celu przetwarzania danych z niekompletnym opisem obiektów w metodzie LRI stosuje się podobny mechanizm do wykorzystywanego w metodzie C4.5.

Podczas wyboru optymalnego testu napotyka się na trudności w porównywaniu jakości testów bazujących na atrybutach o różnej liczbie brakujących wartości. Jakość testów jest mierzona za pomocą liczby popełnianych przez regułę błędów, inaczej niż ma to miejsce w metodzie C4.5, gdzie jakość testów mierzona jest zyskiem informacyjnym uzyskanych podziałów obiektów. Liczba błędów, w przypadku danych o niekompletnym opisie obiektów, jest normalizowana przez iloraz sumy wag wszystkich obiektów przez sumę wag obiektów posiadających wypełnione wartości rozpatrywanych atrybutów, co stanowi odwrotność współczynnika F stosowanego w metodzie C4.5. Główna różnica w stosunku do metody C4.5 polega tutaj na tym, że test nie są oceniane niezależnie. Oceniana jest reguła powstająca przez dodanie kolejnego testu do już wybranych. Oznacza to, że uwzględniana jest liczba brakujących wartości dla któregokolwiek z atrybutów wchodzących w skład reguły.

Klasyfikacja obiektów testowych przez wyindukowany zbiór reguł nie przewiduje możliwości używania brakujących wartości. Przyjmuje się, że wygenerowane reguły są na tyle krótkie i jest ich na tyle dużo, że dla każdego obiektu, nawet o niekompletnym opisie, znajdzie się pokrywająca go reguła. Nie jest to jednak rozwiązanie satysfakcjonujące. Znacznie bardziej adekwatną metodą postępowania była by tutaj na przykład próba częściowego dopasowania obiektów do reguł. Jeśli obiekt spełnia część warunkową reguły na obecnych wartościach atrybutów można przyjąć, że spełnia część warunkową reguły, analogicznie do równości słabych w algebrach częściowych (patrz podrozdział 3.2.1). Liczbę brakujących wartości atrybutów, które wchodzą w skład warunkowej części reguły można potraktować wtedy jako podstawę do obliczenia tzw. współczynnika kary, służącego do zmniejszenia ważności udziału danej reguły w ostatecznym głosowaniu. Jest to rozwiązanie analogiczne do obiektów „ułamkowych” wprowadzonych w metodzie C4.5. Tutaj jednak zmniejsza się nie wagę obiektu, ale wagę reguły (w zakresie $[0, 1]$), aby modelować niedokładne dopasowanie obiektu do jej części warunkowej. Mechanizm głosowania prostego, podczas wyboru ostatecznej klasyfikacji należy wtedy zastąpić głosowaniem z ważoną ważnością głosów.

[3] Reguła jest aktywna dla danego obiektu, gdy obiekt spełnia jej część warunkową.

4.3 Podsumowanie

Zaprezentowane tutaj metody oczywiście nie są jedynymi, które umożliwiają przetwarzanie danych z niekompletnym opisem obiektów w sposób bezpośredni. Jednakże opisane tutaj rozwiązania problemu brakujących wartości uznawane są za skuteczne. Co więcej, praktycznie każda metoda wnioskowania bezpośrednio w oparciu o dane z niekompletnym opisem obiektów i nie wywodząca się z teorii zbiorów przybliżonych działa w oparciu o zbliżone, jeśli nie identyczne, mechanizmy. Należy również zauważyć, że choć istnieją inne metody, umożliwiające przetwarzanie danych z niekompletnym opisem obiektów, nie jest ich znowu aż tak wiele i większość istniejących rozwiązań nie potrafi poradzić sobie z tym problemem.

Rozdział 5

Leniwe metody uczenia maszynowego

W dotychczas zaprezentowanych metodach uczenia maszynowego podejmowaliśmy próbę skonstruowania pewnego pojęcia (klasyfikatora) na podstawie innych pojęć — atrybutów warunkowych obiektów z dostępnego nam podzbioru uniwersum. Zbiór, na którym próbujemy tego dokonać, nazywa się zbiorem obiektów treningowych. Klasyfikacja przynależności innych obiektów (zwanych testowymi) dokonywana jest na podstawie indukcyjnie wyuczonego pojęcia i jest relatywnie szybsza (o znacznie mniejszym nakładzie obliczeniowym) niż sam proces uczenia, który ze swej natury jest zazwyczaj aproksymacją NP-trudnego problemu optymalizacyjnego. Algorytmy z grupy tych metod mają za zadanie jawne sformułowanie pewnej hipotezy, która klasyfikuje wszystkie obiekty, przypisując je do określonego pojęcia (klasy decyzyjnej).

Paradygmat leniwego uczenia maszynowego opiera się na każdorazowej klasyfikacji nowego obiektu — obiektu testowego — na podstawie uprzednio zgromadzonych danych treningowych, a nie wyuczonego opisu pojęcia. Dane treningowe w takim przypadku nie podlegają uprzedniemu specjalnemu przygotowaniu, bądź to przygotowanie jest relatywnie nieskomplikowane i szybkie. Cały ciężar wnioskowania indukcyjnego przerzucony jest tutaj na proces klasyfikacji obiektu testowego i wiąże się z analizą wszystkich zgromadzonych przykładów treningowych.

5.1 Metoda najbliższych sąsiadów

Najprostszą i najbardziej intuicyjną metodą leniwego uczenia maszynowego jest metoda najbliższych sąsiadów (Nearest Neighbours). Jej główną ideą jest selekcja pewnej liczby obiektów treningowych „najbardziej podobnych” do aktualnie klasyfikowanego przykładu. Następnie, na podstawie przynależności tak wyselekcjonowanych obiektów do poszczególnych klas decyzyjnych, dokonuje się głosowania i klasyfikuje się obiekt testowy do tej klasy decyzyjnej, do której przynależało najwięcej spośród wyznaczonych najbliższych sąsiadów. Oczekujemy, że obiekty o podobnym opisie będzie cechowała również podobna klasyfikacja.

Metoda ta daje dobre wyniki wszędzie tam, gdzie zmiany klasyfikacji mają charakter „ciągły” ze względu na opis obiektów i niewielka zmiana opisu najczęściej nie powoduje zmiany przynależności do danego pojęcia. Do zastosowania tej metody potrzebne nam są pojęcie podobieństwa obiektów pomiędzy sobą oraz sposób wyboru zbioru najbliższych sąsiadów i decyzji na podstawie takiego zbioru.

5.1.1 Podobieństwo obiektów

Niech $\mathbb{A} = (U, A \cup d)$ będzie systemem informacyjnym. Dotychczas zbiór atrybutów warunkowych $A = \{a_1, \ldots, a_n\}$ definiowaliśmy jako funkcję $a_i : U \to V_{a_i}$. Każdy z atrybutów postrzegaliśmy jako pojęcie (proste lub złożone) opisujące cechy danego obiektu. Można jednak obiekty z uniwersum U interpretować jako uporządkowane n-tki $U \ni x = (v_{a_1}, \ldots, v_{a_n})$. Wtedy na zbiór U możemy patrzeć jak na podzbiór przestrzeni n-wymiarowej $U \subseteq \mathbb{U} = V_{a_1} \times \ldots \times V_{a_n}$.

W przypadku, gdy przetwarzamy dane o kompletnym opisie obiektów, na przestrzeni $\mathbb{U}$ definiujemy metrykę μ, która określa odległości pomiędzy obiektami. Tak zdefiniowana metryka decyduje o podobieństwie obiektów między sobą. Jeśli obiekty są bliskie sobie, w sensie metryki μ mówimy, że obiekty są do siebie podobne.

Przykład 5.1 *Metryka na przestrzeni $\mathbb{U}$.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym. Zbiór atrybutów A rozkłada się na dwa rozłączne podzbiory, zbiór atrybutów symbolicznych A_s oraz zbiór atrybutów numerycznych A_n. Metrykę μ_1 na przestrzeni $\mathbb{U}$ zdefiniujemy jako funkcję $\mu_1 : \mathbb{U} \times \mathbb{U} \to \mathbb{R}$:

$$\mu_1(x,y) = \sum_{a \in A} \begin{cases} 0 & : \quad a \in A_s \wedge a(x) = a(y) \\ 1 & : \quad a \in A_s \wedge a(x) \neq a(y) \\ |a(x) - a(y)| & a \in A_n \end{cases} \tag{5.1}$$

Metrykę unormowaną μ_2 na przestrzeni $\mathbb{U}$ zdefiniujemy jako funkcję $\mu_1 : \mathbb{U} \times \mathbb{U} \to [0,1]$:

$$\mu_2(x,y) = \frac{1}{n} \sum_{a \in A} \begin{cases} 0 & : \quad a \in A_s \wedge a(x) = a(y) \\ 1 & : \quad a \in A_s \wedge a(x) \neq a(y) \\ \frac{|a(x)-a(y)|}{\sup V_a - \inf V_a} & a \in A_n \end{cases} \tag{5.2}$$

Dla obiektów o kompletnym opisie definiowanie podobieństwa za pomocą metryki jest intuicyjne i wygodne. Warto tutaj przypomnieć, że w teorii zbiorów przybliżonych dla kompletnych danych definiowaliśmy relację nierozróżnialności, która bardzo dobrze odpowiadała intuicyjnemu podobieństwu obiektów między sobą i posiadała tę ważną własność, że była relacją równoważności. Jednakże dla danych z brakującymi wartościami definiowane były inne relacje, które nie koniecznie spełniały warunek przechodniości lub symetrii. Podobnie rzecz ma się i tutaj. W przypadku, gdy tabela informacyjna składa się również z obiektów o niekompletnym opisie może okazać się przydatne zdefiniowanie funkcji μ, która nie spełnia warunku nierówności trójkąta lub przemienności. Jednakże cały czas w mocy pozostaje założenie, że funkcja μ odpowiada podobieństwu obiektów pomiędzy sobą i w dalszej części będzie nazywana funkcją podobieństwa.

Przykład 5.2 *Funkcja podobieństwa dla danych o niekompletnym opisie obiektów.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym oraz wszystkie atrybuty ze zbioru A będą symboliczne. Funkcję podobieństwa μ_3 na przestrzeni $\mathbb{U}$ zdefiniujemy jako funkcję $\mu_3 : \mathbb{U} \times \mathbb{U} \to [0, n]$:

$$\mu_3(x,y) = \sum_{a \in A} \begin{cases} 0 & : \quad a(x) = a(y) \\ 1 & : \quad a(x) \neq a(y) \wedge a(x) \neq * \\ 0 & : \quad a(x) \neq a(y) \wedge a(x) = * \end{cases} \tag{5.3}$$

Funkcja podobieństwa μ_3 nie spełnia ani nierówności trójkąta, ani nie jest przemienna. Niemniej jednak spełniona jest zależność $\mu(x,x) = 0$. Jest to pożądana cecha do procesu klasyfikacji obiektów. Ponieważ nie wiemy, czy dwa identyczne obiekty to jeden i ten sam obiekt, czy też nie, bezpiecznie jest przyjąć zerową „odległość" pomiędzy nimi. [1]

Przy ocenie podobieństwa obiektów można zastosować tzw. ważoną funkcję podobieństwa. Każdemu z atrybutów przypisujemy wagę $w_{a_i} \geqslant 0$, która decyduje o stopniu istotności różnicy obiektów na danym atrybucie. Znajduje to zastosowanie w przypadku, gdy zmienności opisów obiektów na atrybutach w różnym stopniu wpływają na decyzję do której obiekt jest zaklasyfikowany.

Przykład 5.3 *Ważona funkcja podobieństwa.*

Niech $\mathbb{A} = (U, A)$ będzie systemem informacyjnym z poprzedniego przykładu. Przykładem ważonej funkcji podobieństwa na przestrzeni $\mathbb{U}$ jest funkcja μ_4 :

$$\mu_4(x,y) = \sum_{a \in A} \begin{cases} 0 & : \ a(x) = a(y) \\ w_a & : \ a(x) \neq a(y) \wedge a(x) \neq * \\ 0 & : \ a(x) \neq a(y) \wedge a(x) = * \end{cases} \tag{5.4}$$

Wagi atrybutów mogą być arbitralnie dobrana na podstawie wstępnej analizy danych. Jest to również wdzięczne zadanie optymalizacyjne dla algorytmów ewolucyjnych, gdzie w naturalny sposób możemy przyjąć $< w_{a_1}, \ldots, w_{a_n} >$ zarówno za genotyp jak i fenotyp osobnika.

5.1.2 Wybór zbioru najbliższych sąsiadów

Mając zdefiniowaną funkcję podobieństwa możemy przystępować do wyboru zbioru najbliższych sąsiadów. Zbiór najbliższych sąsiadów dla obiektu x będziemy oznaczać przez S_x. Zbiór S_x powinien spełniać następującą własność :

$$\forall y \in U \ \ \mu(x,y) < \max_{z \in S_x} \mu(x,z) \ \Rightarrow \ y \in S_x. \tag{5.5}$$

Proces wyboru zbioru najbliższych sąsiadów ma zazwyczaj ustalony parametr k, który decyduje o liczności zbioru S_x. Przez $T \subseteq U$ oznaczymy zbiór obiektów treningowych. Obiekt x zazwyczaj nie należy do zbioru T, a w szczególności nie należy do zbioru S_x. Jest to nowy obiekt, którego klasyfikacji nie znamy i chcemy ją właśnie wyznaczyć.

Algorytm 5.1 *Wyznaczanie zbioru S_x*

1. S_x := $\emptyset$
2. *wyznacz y takie, że* $\mu(x,y) = \min_{z \in T \setminus S_x} \mu(x,z)$
3. $S_x := S_x \cup \{y\}$
4. *jeśli* $|S_x| = k$ *zakończ, w p.p. przejdź do 2.*

[1] Inaczej, niż będzie miało to miejsce w uzupełnianiu brakujących wartości za pomocą metody najbliższych sąsiadów.

Stosując metodę najbliższych sąsiadów najczęściej wyznacza się zbiór S_x zawierający dokładnie k obiektów, tak jak zostało to zilustrowane powyższym algorytmem. Niemniej jednak można sobie również wyobrazić inną metodę postępowania. W przypadku, gdy funkcja odległości przyjmuje niewiele wartości, wtedy wiele obiektów zostaje „sklejonych" w klasy obiektów równo odległych od x. Możemy wtedy zastosować inny sposób doboru zbioru S_x. Mianowicie wybieramy co najmniej k obiektów, dodając klasy równo odległych obiektów w całości. Gdy okaże się, że liczebność zbioru S_x równa się lub przekracza k kończymy dodawanie, jednakże może się okazać, że zbiór S_x jest istotnie większy niż k obiektów.

5.1.3 Klasyfikacja obiektu

Będąc w posiadaniu zbioru najbliższych sąsiadów. Możemy przystępować do klasyfikacji obiektu x.

Najprostszą metodą klasyfikacji jest głosowanie. Polega to na ustaleniu najczęściej powtarzającej się decyzji w zbiorze S_x. Innymi słowy obiektowi x przypisujemy wartość atrybutu decyzyjnego $d(x) = d_{max}$ taką, że

$$|\{y \in S_x : d(y) = d_{max}\}| = \max_{v_d \in V_d} |\{y \in S_x : d(y) = v_d\}|. \tag{5.6}$$

W przypadku, gdy wartość d_{max} nie może być wyznaczona jednoznacznie możemy poniechać klasyfikacji (odpowiadając „nie wiem") lub przyjąć którąkolwiek z wartości arbitralnie (np. taką, która częściej występuje w całym zbiorze T). Z tego też powodu dobrze jest dobierać nieparzystą wartość k. W przypadku, gdy atrybut decyzyjny przyjmuje tylko dwie wartości (częsty przypadek), wtedy zawsze uzyskamy jednoznaczny wynik głosowania.

Oprócz prostego głosowania można stosować również bardziej skomplikowane metody wyboru decyzji. Na przykład można ważyć głosy obiektów za pomocą wartości funkcji podobieństwa lub stosować kryterium absolutnej większości głosów.

Wartość k należy dobierać eksperymentalnie. Zbyt mały rozmiar zbioru najbliższych sąsiadów prowadzi do częstych błędów przy klasyfikacji obiektów na granicy pojęć. Zbyt duża wartość k prowadzi natomiast do utraty lokalności algorytmu. Wtedy do głosowania brane są również mało lub wcale podobne obiekty i przypomina to bardziej wyznaczanie decyzji dominującej w całym zbiorze treningowym. Zjawisko to jest szczególnie wyraźne, gdy dysponujemy danymi w których pewne wartości atrybutu decyzyjnego są wyraźnie liczniej reprezentowane niż inne.

5.1.4 Brakujące wartości

Metoda najbliższych sąsiadów potrafi wnioskować również na podstawie danych o niekompletnym opisie obiektów. Dzieje się to dzięki abstrakcji jaką nakłada się na zbiór obiektów. Podejmując decyzję nie rozpatruje się tutaj poszczególnych wartości atrybutów, tylko operujemy na podobieństwie obiektów pomiędzy sobą. Jest to podejście naturalne dla człowieka, który często przedstawia dane za pomocą różnego rodzaju diagramów. Szczególnie w przypadku, gdy funkcja podobieństwa jest metryką można wyobrazić sobie, że usiłujemy wyznaczyć kulę zawierającą k najbliższych obiektów w stosunku do badanego i na tej podstawie podjąć decyzję. Jakość klasyfikacji zależy oczywiście od dobranej funkcji podobieństwa, która jest tutaj parametrem.

Niemniej jednak niezależnie od przyjętej funkcji podobieństwa nie dla wszystkich danych możemy uzyskać tutaj zadowalające rezultaty. Ponadto wybór dobrej funkcji podobieństwa jest sam w sobie trudny i często czasochłonny. Również nie bez znaczenia pozostaje fakt, że dla klasyfikacji pojedynczego obiektu musimy wykonać $|T|$ obliczeń funkcji podobieństwa. Oznacza to, że metoda ta jest dużo wolniejsza od innych, nie leniwych metod wnioskowania.

5.2 Leniwe drzewa decyzyjne

Standardowy schemat budowania drzew decyzyjnych opiera się na próbie konstrukcji pojęcia na podstawie danych treningowych. W szczególności, jeśli jest to klasyczna metoda nie adaptacyjna (tzw. off-line), drzewo decyzyjne, raz zbudowane dla danych treningowych, nie ulega żadnym modyfikacjom podczas wyznaczania przynależności do pojęcia poszczególnych obiektów ze zbioru danych treningowych. Jednak, podobnie jak ma to miejsce w metodzie najbliższych sąsiadów, można sobie wyobrazić, że dokonujemy budowy drzewa decyzyjnego nie raz, dla wszystkich obiektów treningowych, ale dla każdego z obiektów testowych z osobna. Ponieważ takie postępowanie niesie ze sobą ryzyko dużej złożoności obliczeniowej, związanej z wielokrotną konstrukcją drzewa decyzyjnego, nieodzownym staje się odpowiedni mechanizm buforowania wspólnych wyników (testów, poddrzew itp.).

Friedman, Kohavi i Yun w pracy [13] zaproponowali metodę LazyDT realizującą paradygmat leniwego uczenia się przy konstrukcji drzew decyzyjnych. Zaprezentowany tam algorytm potrafi w naturalny sposób analizować również dane o niekompletnym opisie obiektów. Charakteryzuje go również kilka innych interesujących własności, które nie są możliwe do uzyskania w modelu tradycyjnych drzew decyzyjnych. Dzięki zastosowaniu mechanizmów buforowania wspólnych wyników częściowych algorytm cechuje się akceptowalnym czasem wykonania.

Metody budowania drzew decyzyjnych borykają się z problemami takimi jak replikacja i fragmentacja. Przypuśćmy, że naszym zadaniem jest klasyfikacja pacjentów jako zdrowy lub chory. Niezwykle ważna wydaje się być informacja, czy ta osoba jest HIV pozytywna, czy też nie, wtedy od razu można stwierdzić, że pacjent jest chory. Jednak jest to mało prawdopodobne, żeby standardowe drzewo decyzyjne posiadało test tego atrybutu w korzeniu, a to za sprawą małej liczby przykładów. Zamiast tego test takiego atrybutu zostanie odsunięty w dół drzewa i tam, na każdej ścieżce, na której występują przykłady pacjentów HIV pozytywnych, test tego atrybutu będzie zreplikowany.

Na podstawie takiej obserwacji można oczekiwać, że drzewa, a raczej ścieżki klasyfikacyjne zbudowane dla poszczególnych przypadków mogą być znacznie krótsze i dawać łatwiejsze wytłumaczenie takiej klasyfikacji (decyzji). Test kilku badań krwi lub podobnych atrybutów może być jasnym i zrozumiałym wytłumaczeniem dla klasyfikacji pacjenta jako zdrowego. Natomiast pacjent łatwo może być sklasyfikowany jako chory na podstawie wyjaśnienia, że jest HIV pozytywny.

5.2.1 Realizacja algorytmiczna

Algorytm klasyfikacji obiektów testowych za pomocą leniwych drzew decyzyjnych jest stosunkowo prosty. Podobnie jak klasyczne algorytmy oparte na drzewach decyzyjnych w swej podstawowej postaci operuje na atrybutach symbolicznych, zatem w celu zaaplikowania go

do danych zawierających atrybuty numeryczne należy proces klasyfikacji poprzedzić dyskretyzacją danych.

Algorytm 5.2 *LazyDT.*
Wejście: Zbiór obiektów treningowych $T = \{t_1, t_2, \ldots\}$ oraz obiekt x będący przedmiotem klasyfikacji.

1. *Jeśli T jest jednorodny, tzn. składa się z obiektów jednej klasy decyzyjnej d, zwróć d jako decyzję dla obiektu x.*
2. *Jeśli obiekty ze zbioru T posiadają wartości wszystkich atrybutów równe x zwróć dominującą klasę d jako decyzję dla obiektu x.*
3. *Wybierz atrybut a_k.*
4. *Jako nowy zbiór T wybierz zbiór tych obiektów treningowych, dla których $a_k(t_j) = a_k(x)$ (dokonaj cięcia na atrybucie a_k przypisując na zbiór T obiekty zgodne z x na atrybucie a_k). Przejdź do 1.*

Podstawowym pytaniem jest w jaki sposób wybierać atrybut a_k w trzecim kroku algorytmu. Zazwyczaj stosuje się w takich przypadkach jedną ze standardowych miar cięć, mierzącą zysk informacyjny (entropia), różnice rozkładu (Gini index,test χ^2) i tym podobne. Jednak nie jest to rozwiązanie satysfakcjonujące. Należy zauważyć, że najwięcej problemów powstaje gdy klasa d_a jest dominująca w zbiorze T, ale klasa d_b była by odpowiedzią prawidłową. Ze względu na to, że standardowe miary cięć biorą pod uwagę jedynie względne częstości występowania obiektów z poszczególnych klas decyzyjnych, nie były by w stanie odgadnąć poprawnej decyzji, a zysk informacyjny przyjął by ujemną wartość.

Przed przystąpieniem do wyboru najbardziej obiecującego atrybutu należy znormalizować liczbę wystąpień każdej klasy decyzyjnej tak, aby były równoliczne. Wtedy łatwo jest wskazać atrybut (czyli zarazem test), który daje największy zysk informacyjny.

Algorytm ten wymaga dla każdego obiektu testowego budowy drzewa decyzyjnego, które zaklasyfikuje ten obiekt do właściwej klasy decyzyjnej. Dla każdego obiektu dokonywany jest wielokrotnie wybór właściwego testu i podział zbioru treningowego. Tak sformułowany algorytm byłby stosunkowo wolny. Kosztem dodatkowej pamięci na przechowywanie wyników częściowych można zastosować pewne mechanizmy buforowania, które bardzo przyspieszą działanie całego procesu klasyfikacji.

5.2.2 Brakujące wartości

Leniwe drzewa decyzyjne ze względu na swoją budowę są łatwe w zaadaptowaniu do działania na danych z niekompletnym opisem obiektów.

Brakujące wartości atrybutów dla obiektów testowych są obsługiwane w naturalny sposób. Atrybut obiektu testowego który posiada brakującą wartość nie jest brany pod uwagę podczas wyboru kolejnego cięcia w trzecim kroku algorytmu. Jest to największa różnica w stosunku do klasycznych drzew decyzyjnych, tam nie można zawczasu wybrać które spośród atrybutów mogą być wzięte do klasyfikacji danego obiektu.

Obiekty treningowe mogą posiadać brakujące wartości na atrybutach nie wchodzących w skład bieżącej ścieżki decyzyjnej dla klasyfikowanego obiektu. Jeśli natomiast dokonywane

jest cięcie na atrybucie, gdzie pewna liczba obiektów treningowych posiada brakujące wartości takie obiekty są eliminowane (tzn. nie wchodzą w skład żadnego z dwóch podzbiorów powstających po cięciu na danym atrybucie). Oczywiście można sobie wyobrażać bardziej wyrafinowane metody filtrowania obiektów treningowych posiadających brakujących wartości podobnie jak ma to miejsce np. w algorytmie C4.5 (patrz podrozdział 4.1).

Rozdział 6

Uzupełnianie

W przypadku napotkania na dane z niekompletnym opisem obiektów naturalnym postępowaniem wydaje się być próba rekonstrukcji pełnych danych. Przy takiej rekonstrukcji wykorzystujemy dostępną wiedzę o obiektach i na tej podstawie staramy się w miejsce brakujących wartości wstawić takie, które wydają się być najbardziej odpowiednie. Jako odpowiedniość można stosować tutaj wiele kryteriów: niesprzeczność, podobieństwo, zachowanie zgodne empirycznym rozkładem prawdopodobieństwa itp. Należy jednak przypomnieć rozgraniczenie na wartości brakujące z powodu braku pomiaru lub zaniedbania oraz na takie, które nie są stosowalne w danym przypadku. Dobrym przykładem na wartość brakującą pierwszego rodzaju jest brak danych co do wzrostu pacjenta. Każdy pacjent cechuje się pewnym wzrostem i w pewnych okolicznościach można podjąć próbę uzupełnienia tej wartości na podstawie innych, znanych informacji. Czasami jednak brak wartości sam w sobie posiada duże znaczenie. Przykładem braku z powodu niestosowalności mogą być tutaj informacje o posiadanym samochodzie takie jak kolor, model, wielkość itp. Wszystkie one nie znajdują zastosowania w przypadku, gdy osoba nie jest posiadaczem żadnego samochodu. Widać od razu, że uzupełnianie takich brakujących wartości nie niesie ze sobą żadnej wartości merytorycznej i pogarsza zdecydowanie jakość danych wejściowych.

6.1 Motywacje i podstawowe problemy

Ze względów zarówno implementacyjnych jak i teoretycznych bardzo pożądane było by, gdyby istniała uniwersalna metoda pozwalająca na rekonstrukcję danych z niekompletnym opisem obiektów do postaci w pełni wypełnionej tabeli informacyjnej. Wszystkie metody pracujące doskonale w przypadku danych z kompletnym opisem obiektów znajdowały by wtedy zastosowanie również w przypadku danych z brakującymi wartościami atrybutów. Również rozważania teoretyczne, dopasowane do przypadku pełnych tabel informacyjnych, mogły by być bez kłopotliwego rozpatrywania brakujących wartości atrybutów przeniesione na grunt tabel niekompletnych. Naturalne wydaje się zatem, że problem ten był i jest wnikliwie badany. Powstało wiele prac na temat uzupełniania brakujących wartości, jednakże metody te uzyskują dobrą skuteczność jedynie w dość wąskim obszarze zastosowań (patrz np. [14, 23, 24, 25, 41, 45]).

Pierwszą, najprostszą metodą radzenia sobie z niekompletnym opisem obiektów, było ignorowanie specjalnego znaczenia brakującej wartości i traktowanie jej jak normalnej, dopuszczalnej wartości z dziedziny atrybutu. Wynikało to wprost z metod implementacji prze-

chowywania zbiorów danych z brakującymi wartościami. Abstrahując od problemów implementacyjnych takie postępowanie jest równoważne uzupełnianiu brakujących wartości za pomocą pewnej specjalnej wartości, która dodawana była do dziedziny każdego z atrybutów na równi ze zwykłymi, dopuszczalnymi wartościami.

Ponieważ brak wartości nie może być reprezentowany w pamięci komputera w sposób bezpośredni, każda implementacja obejmująca brakujące (lub niezdefiniowane) wartości musi je kodować za pomocą pewnego specjalnego słowa, które należy do dziedziny typu danych używanego do reprezentacji, ale nie odzwierciedla żadnej wartości należącej do dziedziny atrybutu. Dlatego interpretowanie tego specjalnego wpisu jako normalnej, dopuszczalnej wartości na równi z pozostałymi, może być interpretowane jako forma uzupełniania brakujących wartości pewną ustaloną wartością z dziedziny atrybutu.

Przykład 6.1 *Przypuśćmy, że mamy tabelę decyzyjną opisującą stan zdrowia pacjentów, w której występuje kolumna „Wzrost" i dla każdego pacjenta przyjmuje ona wartości:*

1 – *niski*
2 – *średniego wzrostu*
3 – *wysoki*

Dziedziną wartości tego atrybutu jest $\{1, 2, 3\}$*. Brakujące wartości muszą być tutaj przedstawione, ze względu na ograniczenia implementacyjne, jako jedna spośród możliwych do reprezentowania liczb całkowitych. Możemy przyjąć, że będziemy traktować* 0 *jako wartość specjalną, oznaczającą wartość brakującą — brak wpisu w daną komórkę pamięci. Rozszerzając odpowiednio dziedzinę atrybutu o* 0 *uzyskujemy możliwość traktowania tak zakodowanych brakujących wartości na równych prawach z pozostałymi, dopuszczalnymi wartościami z dziedziny atrybutu.*

Takie postępowanie wydaje się naturalne i jest często z dużym powodzeniem stosowane w innych dziedzinach informatyki. Jednakże przy dokładnej analizie danych, jaka jest wymagana w inteligentnym przetwarzaniu informacji, niezbędne okazuje się zachowanie wiedzy o tym, że brakujące wartości różnią się zdecydowanie od pozostałych wartości z dziedziny atrybutu.

6.2 Uzupełnianie globalne

Najprostszą metodą uzupełniania danych stosującą „inteligentne" przetwarzanie danych w celu dopasowania odpowiedniej wartości z dziedziny atrybutu do brakującej wartości w opisie obiektu jest uzupełnianie globalne. Przetwarzanie danych zawartych w tabeli informacyjnej polega tutaj na zastosowaniu pewnych statystyk na posiadanym zbiorze danych. Standardowym postępowaniem jest dobór jakiejś naturalnej statystyki, obliczenia jej wartości dla wszystkich znanych wartości danego atrybutu (czyli wszystkich wypełnionych miejsc danej kolumny), a następnie uzupełnienie brakujących miejsc za pomocą tak wyliczonej wartości. Najczęściej używane tutaj statystyki, to średnia lub mediana dla atrybutów o dziedzinie liniowo uporządkowanej (zazwyczaj liczbowej) oraz najczęściej występująca wartość dla pozostałych atrybutów.

Algorytm 6.1

1. *Wyznacz wartość $s \in V_a$ za pomocą statystyki S,*
   s:=$S(\{v_a : x \in U \wedge a(x) = v_a \neq *\})$.
2. *Dla każdego obiektu x takiego, że $a(x) = *$ powtarzaj*
   (a) $a(x)$:=s.

Odczytanie wszystkich wartości jest wymagane, ponieważ musimy dysponować wyliczoną statystyką, żeby przystąpić do uzupełniania brakujących wartości. Zatem złożoność problemu jest $\Omega(N)$ (gdzie N to liczba obiektów). Algorytm można zapisać tak, żeby odczytywał co najwyżej dwukrotnie zawartość tabeli, zatem jego złożoność obliczeniowa jest rzędu $\Theta(N)$. Złożoność pamięciowa zależy od przyjętej statystyki S i wynosi $O(1)$ dla średniej oraz $O(|V_a|)$ (gdzie V_a to dziedzina atrybutu a) dla najczęściej występującej wartości. Jeśli wybraną statystyką jest mediana, to teoretycznie można algorytm zaimplementować w miejscu (tj. o złożoności pamięciowej $O(1)$), ale albo zwiększa to czas wykonania do $O(N \log_2(N))$, albo wymaga użycia takich algorytmów liniowych (np. algorytm Bluma-Floyda-Pratta-Rivesta-Tarjana), gdzie w notacji $O(N)$ jest ukryta duża stała, zazwyczaj większa zarówno od $|V_a|$ jak i od $\log_2(N)$. Taka implementacja była by więc nieefektywna ze względów praktycznych, gdzie podstawowym problemem jest czas działania, a nie zajętość pamięci.

Pomimo swej prostoty, metoda ta daje najczęściej dosyć dobre wyniki, chociaż odbiegające wyraźnie od pozostałych, bardziej wyrafinowanych metod. Stosując ten algorytm do konkretnych danych można próbować go dostroić, dobierając bardziej odpowiednią statystykę, jednakże ze względu na globalne wyliczanie wartości używanej do uzupełniania brakujących miejsc, takie strojenie można przeprowadzić tylko w ograniczonym zakresie.

Prezentowana powyżej metoda, to uogólnienie opisywanych w literaturze metod „Most Common Value” (patrz [23, 24, 25]) oraz „Mean-and-Mode” (patrz [14]).

6.3 Uzupełnianie lokalne względem decyzji

Poprzednią metodę można na gruncie uczenia maszynowego zakwalifikować do metod „bez nauczyciela” („bez nadzoru”). W przypadku, gdy wśród atrybutów wyróżniamy atrybut decyzyjny d, dysponujemy klasyfikacją obiektów do poszczególnych klas decyzyjnych. Można wtedy ulepszyć takie uzupełnianie dzieląc wstępnie obiekty na zbiory odpowiadające poszczególnym klasom decyzyjnym. Takie postępowanie odpowiada metodom „z nauczycielem” („z nadzorem”), które cechują się najczęściej większą sprawnością niż metody „bez nauczyciela”. Podczas gdy w poprzednim algorytmie bazujemy na dystrybucji wartości danego atrybutu na wszystkich obiektach w tabeli, teraz możemy lokalnie obliczyć dystrybucję wartości oddzielnie dla obiektów z różnych klas decyzyjnych. Patrząc na to w taki sposób, że obiekty w tablicy są przykładami należącymi do różnych pojęć, a pojęcia te zakodowane są w postaci różnych wartości atrybutu decyzyjnego, odpowiada to podzieleniu tabeli na zbiory przykładów poszczególnych pojęć. Dopiero na tak podzielonej tabeli stosujemy poprzedni algorytm oddzielnie dla każdego zbioru obiektów.

Algorytm 6.2

1. *Podziel zbiór atrybutów na grupy względem przynależności do klas decyzyjnych,*
   U_{d_i}:=$\{x \in U : d(x) = d_i\}$.

2. *Dla każdej grupy wyznacz wartość* $s_{d_i} \in V_a$ *za pomocą statystyki* S, s_{d_i}:=$S(\{v_a : x \in U_{d_i} \wedge a(x) = v_a \neq *\})$.

3. *Dla każdego obiektu* x *takiego, że* $a(x) = *$ *powtarzaj*

   (a) d_i:=$d(x)$, *pod warunkiem, że* $a(x)$=s_{d_i}.

Algorytm musi poznać zawartość całej tabeli. Można go zaimplementować tak, aby odczytywał zawartość tabeli dwukrotnie. Zatem jego złożoność obliczeniowa jest rzędu $\Theta(N)$. Złożoność pamięciowa zależy od przyjętej statystyki S i wynosi $O(|V_d|)$ (gdzie $|V_d|$ to liczba klas decyzyjnych) dla średniej, oraz $O(|V_d| \cdot |V_a|)$ (gdzie V_a to dziedzina atrybutu a) dla najczęściej występującej wartości. Ponieważ jednak V_d zazwyczaj jest małe oraz z góry ustalone dla danego zastosowania możemy o niej myśleć jak o stałej.

Metoda ta daje dosyć dobre wyniki w porównaniu z innymi metodami radzenia sobie z brakującymi wartościami (nie tylko uzupełnianiem). Należy jednak zwrócić uwagę, że brakujące wartości uzupełniane są zgodnie z naszymi oczekiwaniami dotyczącymi klasyfikacji obiektów do poszczególnych klas decyzyjnych. Takie postępowanie może prowadzić do nadmiernego wzmacniania i wyostrzania danych do już posiadanej informacji — czyli ich samych. Jest to swoiste sprzężenie zwrotne, które eliminuje na siłę sprzeczności w danych, będące zazwyczaj ich integralną częścią, występującą często na granicy pojęć. Zjawisko takie jest szeroko znane w uczeniu maszynowym i określa się je jako nadmierne dopasowanie (ang. „over-fitting").

Prezentowana powyżej metoda to ulepszenie uzupełniania globalnego, prezentowanego powyżej, zainspirowane metodami „Global Closest Fit" i „Concept Closest Fit" opisanymi w pracach [23, 24, 25]. Ponadto ostatnio, w pracy [14], opisana została bardzo podobna metoda „Natural Cluster Based Mean-and-Mode", która jest analogicznym rozwinięciem prezentowanej tam metody „Mean-and-Mode".

6.4 Uzupełnianie lokalne względem atrybutu

Warto zauważyć, że metoda uzupełniania lokalnego względem decyzji (czyli atrybutu decyzyjnego) w dość naiwny sposób zakłada, że pojedyncze atrybuty warunkowe są skorelowane z atrybutem decyzyjnym. Jednakże taka sytuacja wcale nie musi mieć miejsca. Oddzielnie traktowane atrybuty warunkowe mogą być niezależne od decyzji, chociaż w większej liczbie mogą dokładnie wyznaczać decyzje.

Przykład 6.2 *Problem XOR.*

Załóżmy, że mamy dwie zmienne losowe A *i* B*, które przyjmują wartości* 0 *lub* 1 *z jednakowym prawdopodobieństwem* $\frac{1}{2}$*. Zdefiniujmy zmienną losową* $C = A \text{ xor } B$*. Zmienna losowa* C *jest całkowicie wyznaczona przez zmienne losowe* A *i* B*. Jednakże* C *jest niezależną zmienną losową z* A *i* B *traktowanymi oddzielnie.*

Naturalnym ulepszeniem powyższej metody jest zastosowanie zamiast atrybutu decyzyjnego innego atrybutu, bardziej skorelowanego z atrybutem, którego wartość chcemy uzupełnić. Dobór atrybutu, który jest związany większymi zależnościami, powinien zaowocować mniejszym nadmiernym dopasowaniem wpisywanych wartości do znanych obiektów treningowych i daje większe szanse prawidłowego zaklasyfikowania obiektów testowych. Podsta-

wowym zatem problemem jest zbadanie, które atrybuty są ze sobą związane największymi zależnościami.

Jeśli badana jest para atrybutów numerycznych, możemy zastosować dobrze znany ze statystyki i wykorzystywany często przy wstępnym wykrywaniu cech znaczących współczynnik korelacji. Jeśli chcemy sprawdzić jak bardzo są od siebie zależne dwa atrybuty symboliczne (tj. o dyskretnej i nieuporządkowanej dziedzinie) możemy wykorzystać miary informacyjne zbiorów stosowane najczęściej do konstrukcji drzewach decyzyjnych. Możemy tutaj zastosować takie miary jak: entropia, rozróżnialność, konflikt, Gini indeks, χ^2 i inne podobne, szeroko znane i badane przy okazji problemu optymalnych testów w wierzchołkach drzew decyzyjnych oraz dyskretyzacji atrybutów. Zazwyczaj kosztem niewielkiej dodatkowej pamięci można zaimplementować obliczenie takiej miary w czasie liniowym ze względu na liczbę obiektów.

O ile porównywanie par atrybutów numerycznych oraz par atrybutów symbolicznych ze sobą nie nastręcza większych trudności, to nie istnieje dobra i niezawodna metoda porównywania atrybutów symbolicznych z numerycznymi. Takie porównania mogą być konieczne, jeśli np. w tabeli informacyjnej jeden atrybut jest numeryczny, a wszystkie pozostałe są symboliczne. Ponadto, może się okazać, że większe zależności wiążą parę atrybutów różnego typu, czego nie jesteśmy w stanie stwierdzić analizując tylko pary atrybutów tego samego typu. Jeśli dziedzina atrybutu numerycznego jest dyskretna i niewielkiej mocy możemy wtedy pominąć informację o tym, że wartości takiego atrybutu są liniowo uporządkowane i potraktować tak, jak by były wartościami symbolicznymi. W przeciwnym przypadku celowe jest zastosowanie dyskretyzacji.

Metody analizy danych oparte na teorii zbiorów przybliżonych wymagają danych wstępnie zdyskretyzowanych. W takich danych wszystkie atrybuty numeryczne zostały zamienione atrybutami symbolicznymi wyznaczonymi w sposób, który ma na celu zachowanie jak najwięcej cennych informacji dla procesu analizy. Ma to też tę zaletę, że odsiewa zbędny szum informacyjny związany z gęstą dziedziną liczb rzeczywistych, a związany z takimi zjawiskami jak błędy pomiaru, czy naturalny rozrzut danego parametru dookoła pewnej wartości. Dane tak przygotowane składają się wyłącznie z atrybutów symbolicznych. Można wtedy zastosować metodę uzupełniania lokalnego względem atrybutu stosując jedną miarę informacyjną zbiorów dla wszystkich atrybutów. Wyniki porównania zależności atrybutów pomiędzy sobą są wtedy obiektywne i lepiej nadają się do wyznaczenia atrybutu związanego największymi zależnościami.

Prezentowana powyżej metoda jest połączeniem metod „Attribute Rank Cluster based Mean-and-Mode algorithm” oraz „K-Means Clustering based Mean-and-Mode algorithm” prezentowanych w pracy [14].

6.5 Uzupełnianie metodą najbliższych sąsiadów

Bardziej wyrafinowanym sposobem uzupełniania brakujących wartości jest zastosowanie metody najbliższych sąsiadów. Metoda ta jest zazwyczaj wykorzystywana do klasyfikacji obiektów i opisana jest bardziej szczegółowo w rozdziale dotyczącym leniwych metod uczenia się. Jednakże można jej główną ideę wykorzystać również do uzupełniania brakujących wartości.

Prezentowane do tej pory metody uzupełniania niekompletnego opisu obiektów koncen-

trowały się głównie na zależnościach pomiędzy atrybutami w badanej tabeli informacyjnej. Wszystkie obiekty były traktowane grupowo i tylko w niewielkim stopniu wykorzystywana była informacja o wzajemnym podobieństwie obiektów do siebie. Co najwyżej jeden atrybut brany był pod uwagę przy ocenie podobieństwa obiektów. Można sobie jednak wyobrazić metodę działającą w odmienny sposób, gdzie pierwszym i najważniejszym krokiem jest dobór obiektów w pewnym sensie najbardziej podobnych do badanego, wykorzystującą całą dostępną informacje o obiektach.

Motywacją do zastosowania metody najbliższych sąsiadów jest to, że obiekty o zbliżonym opisie na istniejących wartościach atrybutów prawdopodobnie cechuje również podobieństwo na pozostałych atrybutach (w tym niewypełnionych). Ponieważ klasyfikacja oparta na metodzie najbliższych sąsiadów uzyskuje dosyć dobre rezultaty (przynajmniej dla niektórych danych) i jest intuicyjnie prosta w interpretacji, można przyjąć, że powinna również dawać dobre rezultaty gdy wykorzysta się ją do uzupełniania brakujących wartości.

Podstawowym pojęciem jakie należy zdefiniować do zastosowania tej metody jest podobieństwo obiektów między sobą. Dla danych o w pełni kompletnym opisie obiektów przyjmuje się najczęściej, że przestrzeń obiektów jest przestrzenią metryczną. W przypadku, gdy dane posiadają obiekty o niekompletnym opisie przyjmuje się słabsze założenia, określając funkcję podobieństwa na przestrzeni obiektów. Za pomocą dobranej funkcji podobieństwa wybiera się k sąsiadów o najmniejszej odległości od obiektu badanego, dla pewnego ustalonego k. Polega to na wyliczeniu odległości wszystkich obiektów od obiektu badanego i wybraniu spośród nich k obiektów najbliższych.

Zdefiniowanie funkcji podobieństwa, która nie spełnia nawet warunku $\mu(x, x) = 0$, może mieć w przypadku uzupełniania brakujących wartości swoje uzasadnienie. Zaprezentowana poniżej funkcja podobieństwa preferuje obiekty bardziej wypełnione. Zatem w zbiorze najbliższych sąsiadów znajdzie się więcej wartości, na podstawie których możemy wyznaczyć wartość odpowiednią do wstawienia na miejsce brakującej.

Przykład 6.3 *Funkcja podobieństwa, która nie spełnia warunku* $\mu(x, x) = 0$.

$$\mathrm{f}_1(x, y) = \sum_{a \in A} \begin{cases} 0 & : a(x) = a(y) \neq * \\ 1 & : a(x) = a(y) = * \\ 1 & : a(x) \neq a(y) \wedge a(x) \neq * \\ \frac{1}{2} & : a(x) \neq a(y) \wedge a(x) = * \end{cases} \tag{6.1}$$

Dysponując dużym zbiorem danych można również zastosować nieco inną metodę selekcji k obiektów najbardziej podobnych, tutaj jednak k nie jest z góry ustalone. Jako funkcję podobieństwa można przyjąć tym razem funkcję, która zwraca 0, jeśli obiekty są identyczne na uzupełnionych wartościach, oraz 1 w przeciwnym przypadku. Odpowiada to relacji podobieństwa symetrycznego na gruncie teorii zbiorów przybliżonych.

$$\mathrm{f}_2(x, y) = \begin{cases} 0 & gdy \ \ \forall a \in A \ \ a(x) = a(y) \vee a(x) = * \vee a(y) = * \\ 1 & wp.p. \end{cases} \tag{6.2}$$

Teraz, jako k najbliższych sąsiadów wybieramy wszystkie obiekty, które są w zerowej „odległości" od obiektu badanego. k w tym przypadku jest zmienne, niemniej jednak w dalszym ciągu dysponujemy zbiorem najbliższych sąsiadów. Należy również zauważyć, że dla niektórych danych zbiór taki może okazać się pusty.

Dysponując zbiorem (niepustym) k najbliższych sąsiadów możemy zastosować metodę uzupełniania brakujących wartości za pomocą prostych statystyk (najczęstszej wartości, mediany czy średniej). Oczekujemy, że w tym przypadku obiekty będące najbliższymi sąsiadami zostały starannie dobrane spośród zbioru obiektów treningowych i będą w lepszym stopniu opisywały możliwą do uzupełnienia wartość.

W naturalny sposób możemy zmodyfikować poprzednie metody tak, aby zamiast wybranych obiektów na podstawie decyzji, czy też atrybutu najbardziej skorelowanego, do procesu wyliczania statystyk brały zbiór obiektów wyznaczonych na podstawie metody najbliższych sąsiadów, czy też powyższej modyfikacji. W metodzie tej zarówno możemy sterować parametrami funkcji podobieństwa obiektów, jak i również statystyką, na podstawie której wylicza się wartość do wstawienia. Należy zauważyć, że statystyka najczęstszej wartości odpowiada standardowemu głosowaniu w oryginalnej metodzie najbliższych sąsiadów.

Ze względu na to, że proces wyboru obiektów najbardziej podobnych może być czasochłonny, warto jest od razu wypełnić wszystkie brakujące miejsca w uzupełnianym obiekcie, żeby oszczędzić czasu na ponowne wyznaczanie zbioru najbliższych sąsiadów. Niemniej jednak ta metoda jest dużo bardziej czasochłonna niż wcześniej opisane metody uzupełniania brakujących wartości i szczególnie w przypadku gdy dysponujemy dużymi zbiorami danych treningowych należy zastanowić się nad celowością jej stosowania. Można również ze zbioru danych treningowych wydzielić mniejszy zbiór i tylko w nim poszukiwać najbliższych sąsiadów. Należy tego dokonać starannie, aby zbiór ten był reprezentatywny w odniesieniu do całego zbioru treningowego.

6.6 Uzupełnianie za pomocą systemu decyzyjnego

Powyżej opisany sposób uzupełniania brakujących wartości za pomocą metody najbliższych sąsiadów może nasunąć spostrzeżenie, że każdy klasyfikator — system decyzyjny byłby dobrym, a nawet lepszym substytutem metody najbliższych sąsiadów.

Proces wypełniania brakujących wartości jest analogiczny do procesu klasyfikacji obiektów do poszczególnych pojęć. Podczas klasyfikacji wypełniamy brakującą wartość obiektu na atrybucie decyzyjnym. Zatem teoretycznie można by zastosować analogiczny proces do uzupełniania innych brakujących wartości, nie tylko decyzji ale również atrybutów warunkowych, traktując je jako „tymczasowy atrybut decyzyjny”. Należy jednak zastanowić się nad zasadnością takiego postępowania.

Dysponując systemem decyzyjnym, który nie potrafi wnioskować w oparciu o dane z brakującymi wartościami atrybutów, musimy ograniczać się albo do pewnego podzbioru obiektów treningowych, które posiadają kompletny opis (często taki zbiór może być pusty) lub do pewnego podzbioru atrybutów, na których wszystkie obiekty są opisane (również może okazać się pusty). Nawet jeżeli proces taki dla konkretnych danych jest wykonalny, to ze względu na to, że nie uwzględnia on całej informacji zawartej w danych, a tylko jej wycinek, może wprowadzać duże zaburzenia i mylne wartości, które skutecznie zaszumiają wiedzę zawartą w tabeli informacyjnej. Ponadto wszystkie systemy decyzyjne cechuje ograniczona sprawność klasyfikacji, która dla typowych danych oscyluje zazwyczaj w przedziale 60%–95%, zatem nie możemy mieć gwarancji, że system wykorzystujący informację zawartą w danych dobrze uzupełni brakujące wartości.

Gdy dysponujemy systemem decyzyjnym, który potrafi wnioskować w oparciu o dane

z niekompletnym opisem obiektów, wtedy powstaje pytanie, czy w ogóle warto jest dane uzupełniać. Uzupełnianie brakujących wartości po pierwsze nie zawsze znajduje oparcie w rzeczywistości (gdy np. dany obiekt istotnie nie posiada żadnego opisu względem danego pojęcia), a po drugie wprowadza zniekształcenia i powoduje zjawisko nadmiernego dopasowania się do danych (ang. *over-fitting*). Nie bez znaczenia pozostaje też fakt, że proces klasyfikacji obiektów jest dosyć czasochłonny i jego wielokrotne wykonywanie przy braku gwarancji powodzenia przestaje być zasadne. Tym bardziej, że istnieje możliwość jednokrotnego zanalizowania danych za pomocą tej metody, która i tak potrafi się uporać z brakującymi wartościami bez potrzeby ich uzupełniania.

6.7 Podsumowanie

Uzupełnianie brakujących wartości jest uniwersalną metodą radzenia sobie z problemem danych o niekompletnym opisie obiektów. Należy jednak zdawać sobie sprawę z ograniczonego zakresu zastosowań tego podejścia. Wypełnianie brakujących miejsc niesie ze sobą zagrożenie wprowadzenia istotnych zaburzeń do danych, uniemożliwiając tym samym wykrycie subtelnych zależności pomiędzy atrybutami warunkowymi a decyzją.

Należy wspomnieć, że na gruncie statystyki dopracowano się ważnych wyników dotyczących uzupełniania. Przede wszystkim należy tutaj wspomnieć o metodzie EM (patrz np. [15, 61]). Oryginalnie jest to metoda służąca klastrowaniu danych. Polega ona na dopasowaniu pewnej liczby rozkładów prawdopodobieństwa do grup obiektów w taki sposób, aby maksymalizować szansę, że istniejące obiekty zostały wylosowane właśnie z tych rozkładów. Rozkłady te są wyznaczane iteracyjnie, kolejno przybliżając coraz dokładniej zaobserwowane empiryczne prawdopodobieństwa wartości obiektów. Uzupełnianie brakujących wartości metodą EM polega na dolosowaniu brakujących wartości z tak wyznaczonych rozkładów prawdopodobieństwa.

We współczesnej statystyce metodę EM stosuje się w połączeniu z tzw. uzupełnianiem wielokrotnym. Polega to na wygenerowaniu kilku alternatywnych tabel uzupełnionych za pomocą metody EM oraz połączeniu wyników klasyfikacji na każdej z tych tabel przez głosowanie. W pracy [45] zostało udowodnione, że nawet niewielka liczba takich alternatywnych tabel potrafi znacząco poprawić jakość klasyfikacji.

Warto tutaj przypomnieć, że na gruncie teorii zbiorów przybliżonych istnieje analogiczne rozwiązanie w postaci relacji tolerancji. Odpowiada to rekombinacji wyników z wszystkich możliwych uzupełnień danej tabeli. Powstaje zatem pytanie o zasadność stosowania tak wyrafinowanych i czasochłonnych metod uzupełniania, gdy dostępne są równoważne i szybsze rozwiązania oparte o teorię zbiorów przybliżonych.

Rozdział 7

Metoda podziału

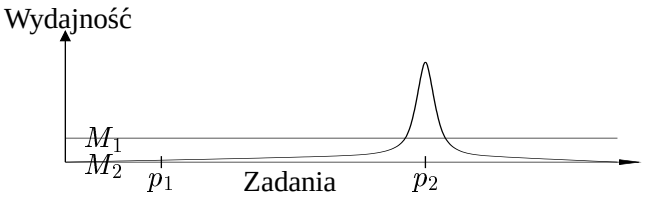
7.1 Wprowadzenie

W większości dotychczas opisywanych metod wnioskowania na podstawie danych z niekompletnym opisem obiektów usiłowano dopasować brakujące miejsca do istniejących wartości danego atrybutu. Działo się to przez założenie, że brakująca wartość może być dowolną z dopuszczalnych wartości atrybutu czy też przez dystrybucję obiektów „ułamkowych" do grup obiektów o poszczególnych wartościach danego atrybutu. Również dosyć uniwersalna metoda, jaką jest uzupełnianie brakujących wartości, miała na celu zaniedbanie informacji o tym, że dana wartość jest brakująca i wypełnienie wszystkich brakujących wartości kosztem zaburzenia danych. Nie jest to jednak postępowanie naturalne i zgodne z ludzką intuicją. Poszukiwać należało by raczej metody, która umożliwiała będzie bezpośrednie operowanie na danych o niekompletnym opisie obiektów.

Pierwszym powodem, dla którego istniejące metody mogą okazać się nieskuteczne, jest nienaturalne traktowanie brakujących wartości. Umysł człowieka, który jest najlepszym znanym systemem decyzyjnym, zawsze potrafi poradzić sobie z tym problemem. Jeżeli lekarz ma stwierdzić stan zdrowia pacjenta nie dysponując kompletem badań, wtedy nie usiłuje uzupełniać brakujących wyników na podstawie istniejących, tylko próbuje sformułować diagnozę tylko i wyłącznie na podstawie tych danych, którymi dysponuje. Jeżeli nie jest to całkowicie możliwe, wtedy formułuje odpowiedź przybliżoną i ewentualnie zleca wykonanie dodatkowych badań. Może on dokonywać porównań z wynikami innych pacjentów, jednakże nie dzieje się to w oparciu o brakujące wartości. Pomimo tego, że lekarz dysponuje wiedzą o podobnych przypadkach, wnioskuje jednak na podstawie istniejących informacji i ani nie uzupełnia danych, ani nie ocenia możliwego wyniku danych, gdyż nie było by to wiarygodne.

Istniejące algorytmy, które potrafią poradzić sobie z brakującymi wartościami atrybutów, takie jak LRI czy LazyDT różnią się zdecydowanie od najpopularniejszych obecnie algorytmów. Wykorzystane tam metody generowania drzew decyzyjnych i indukcji reguł co prawda potrafią poradzić sobie z brakującymi wartościami, niemniej jednak odbija się to niekorzystnie na efektywności. Ponadto metody te uniemożliwiają wykorzystanie ugruntowanej wiedzy w zakresie tak dobrze zbadanych zagadnień jak zbiory przybliżone, czy metod optymalizacji drzew decyzyjnych (np. przycinanie [8, 58]).

Kolejną motywacją do poszukiwań innej metody radzenia sobie z brakującymi wartościami jest duża liczba istniejących skutecznych metod, które nie potrafią sobie poradzić z



Rysunek 7.1: Metoda M_1 jest bardziej ogólna niż metoda M_2 i może być z powodzeniem stosowana do szerszej klasy zadań. Jednakże na swoim odcinku specjalizacji metoda M_2 osiąga zdecydowanie większą wydajność (patrz [31]).

brakującymi wartościami. Metody te były badane na przestrzeni wielu lat, mają ugruntowane podłoże teoretyczne oraz są licznie reprezentowane przez częstokroć duże programy komputerowe, które zostały zaimplementowane wielkim nakładem pracy. Adaptacja istniejących gotowych programów komputerowych niewielkim nakładem pracy tak, aby były w stanie poradzić sobie z danymi o niekompletnym opisie obiektów z zadowalającą jakością, byłaby znakomitym rozwiązaniem.

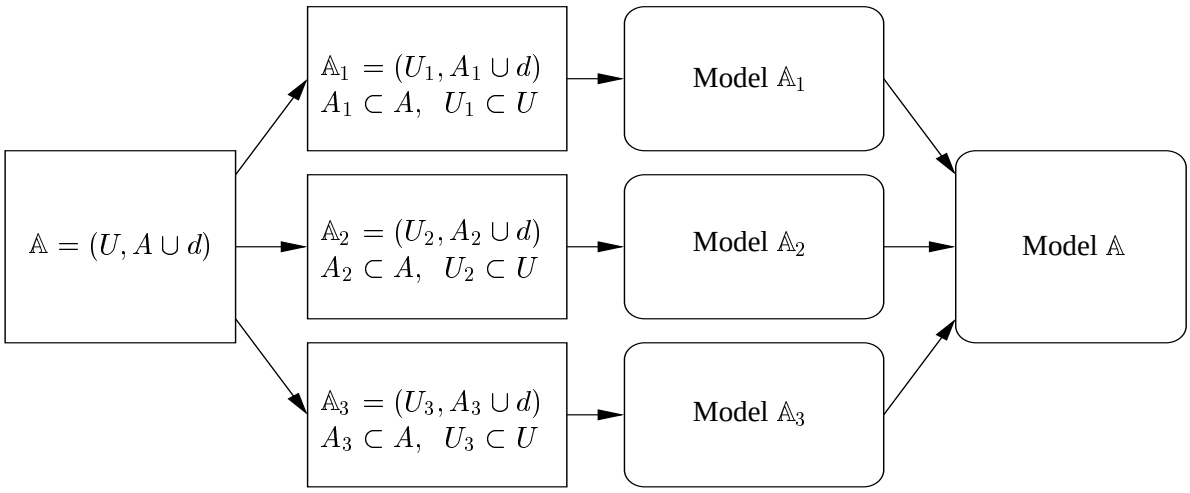
7.2 Motywacje

Powszechnie znanym faktem jest, że metody wąsko wyspecjalizowane lepiej sprawdzają się w swojej dziedzinie, niż metody ogólne (zobacz rys. 7.1). Co prawda metody ogólne można stosować na szerszej klasie problemów, jednakże metody wyspecjalizowane w rozwiązywaniu konkretnych problemów uzyskują zdecydowanie większą wydajność. Sytuację tę można przyrównać do człowieka ogólnie wykształconego i np. specjalisty w zakresie samochodów. W zasadzie każdy wie, gdzie w samochodzie znajduje się silnik, niemniej jednak jego naprawę lepiej zlecić specjaliście w tej dziedzinie, niż wykonywać samemu.

W ostatnich latach na znaczeniu uzyskały metody merologiczne jak i obliczeń na granulach (patrz np. [26]), których myślą przewodnią jest dekompozycja skomplikowanych zadań na prostsze, które można by wykonywać za pomocą wyspecjalizowanych metod. Dekompozycja skomplikowanych zadań nie jest zresztą pomysłem nowym i znajdowała się zawsze w polu zainteresowań sztucznej inteligencji takich jak planowanie, czy systemy wieloagentowe. Niemniej jednak dopiero niedawno za sprawą obliczeń na granulach stało się realne inteligentne wykorzystanie dekompozycji do celów analizy danych i odkrywania wiedzy.

Dekompozycja to bardzo silne narzędzie do walki ze złożonością problemów ze świata rzeczywistego. Polega to najczęściej na podziale modelu całego zadania na lokalne podmodele opisujące prosty i niezależny fragment, który może zostać w całości poddany dalszej analizie. W następnych krokach dokonuje się syntezy wiedzy z lokalnych podmodeli, która może być wykonywana wieloetapowo, tworząc strukturę drzewa (lub grafu) zależności pomiędzy modelami. Mówi się czasem również o przetwarzaniu warstwowym, czy uczeniu warstwowym w kontekście maszynowego uczenia.

Kolejnym, godnym zainteresowania zagadnieniem, są zaawansowane metody uzupełniania brakujących wartości za pomocą systemów decyzyjnych. Ich działanie opisane zostało w poprzednim rozdziale. Tutaj warto tylko przypomnieć, że metody te używają klasyfikatora na wejściowej tablicy (podtablicy) w celu wypełnienia brakujących wartości w jednej z kolumn tej tablicy. W celu uzupełnienia większej liczby kolumn, musimy posłużyć się większą liczbą klasyfikatorów. Można zatem mówić o swoistym sprzężeniu zwrotnym, gdyż mody-



Rysunek 7.2: Oto jest myśl przewodnia metody podziału. Dane z tabeli $\mathbb{A}$ dekomponowane są na podtabele $\mathbb{A}_i$. Dla każdej z podtabel tworzony jest model opisujący pojęcię obcięte do $\mathbb{A}_i$. Końcowy model uzyskuje się na podstawie modeli pojęcia dla podtabel.

fikujemy tablicę na podstawie której wnioskujemy. Nie jest to zatem metoda bezpieczna, gdyż uzupełnianie wprowadza szumy i zaburzenia do danych, a procesy, w których występuje sprzężenie zwrotne są zazwyczaj mało stabilne i zwracają trudne do przewidzenia wyniki. Niemniej jednak, jest to teoretycznie najlepsza metoda uzupełniania brakujących wartości. Gdyby udało się z tej metody wyeliminować sprzężenie zwrotne i uzupełnianie samo w sobie, gwarantowało by to nam dużą staranność i skuteczność w obchodzeniu się z brakującymi wartościami.

7.3 Metoda podziału

Ideą przewodnią metody podziału, jest dekompozycja i zastosowanie wielu klasyfikatorów. Proces wnioskowania na danych wejściowych jest dekomponowany w taki sposób, żeby wnioskować tylko i wyłącznie na podstawie tabel informacyjnych z kompletnym opisem obiektów. Bardzo ważnym aspektem jest to, żeby taka dekompozycja zachowała możliwie najwięcej informacji z początkowych danych. W przeciwnym przypadku możemy utracić zarówno atrybuty warunkowe, które są związane zależnościami z atrybutem decyzyjnym, jak i niezbędną liczbę obiektów, umożliwiającą poprawne wyuczenie się pojęcia. Dekompozycja realizowana jest poprzez wydzielenie lokalnych podzbiorów danych treningowych, które nie zawierają żadnych brakujących wartości. Podzbiory te mogą mieć zarówno mniejszą liczbę obiektów jak i mniejszą liczbę atrybutów niż dane wejściowe. Jednakże wszystkie obiekty ze zbioru treningowego powinny znaleźć się w przynajmniej jednym z podzbiorów, a każdy z takich podzbiorów powinien mieć jak największą liczbę atrybutów.

Następnie, na podstawie podzbiorów danych tworzone są lokalne modele. Modele te mają za zadanie jedynie opisać pojęcie na swoim podzbiorze danych treningowych. W celu uzyskania opisu pojęcia na całym zbiorze należy ponownie zastosować system decyzyjny, który tym razem przyjmuje jako dane wejściowe odpowiedzi od modeli lokalnych. Na tej podstawie podejmuje się decyzję dla wszystkich obiektów z uniwersum (patrz rys. 7.2).

Do formalnego zdefiniowania procesu dekompozycji potrzebne nam będzie pojęcie wzorca wypełnienia.

7.4 Wzorce wypełnienia

Często stosowanym w analizie tabel informacyjnych pojęciem jest wzorzec. Mówimy, że obiekt pasuje do wzorca, gdy jego opis spełnia formułę logiczną definiującą dany wzorzec. Pojęcie wzorca jest bardzo ogólne i szeroko stosowane (patrz np. [33, 34, 35]). Tutaj jednak będziemy posługiwali się uproszczoną postacią wzorców. Ich jedynym zadaniem będzie selekcja obiektów o podobnym wypełnieniu opisu wartościami atrybutów.

Definicja 7.1 *Wzorzec wypełnienia.*

Deskryptorem wypełnienia nazwiemy każdy napis postaci $a \neq *$, gdzie $a \in A$ jest atrybutem występującym w badanej tabeli informacyjnej. Powiemy, że obiekt x spełnia deskryptor $a \neq *$, wtedy i tylko wtedy, gdy $a(x) \neq *$. Wzorcem wypełnienia nazwiemy koniunkcję zbioru (może być pusty) deskryptorów wypełnienia. Obiekt spełnia wzorzec wypełnienia, gdy spełnia każdy z deskryptorów wypełnienia wzorca. Obiekt x spełnia wzorzec w oznaczymy $x \models w$.

Przykład 7.1

Obiekt x określony na atrybutach $a_1, \ldots, a_4$ w sposób następujący $a_1(x) = 1$, $a_2(x) = *$, $a_3(x) = 0$, $a_4(x) = *$ spełnia wzorce wypełnienia:

- $t_1 = \epsilon$ *— wzorzec pusty, każdy obiekt spełnia wzorzec pusty,*
- $t_2 = a_1 \neq *$*,*
- $t_3 = a_3 \neq *$*,*
- $t_4 = a_1 \neq * \wedge a_3 \neq *$.

Obiekt x nie spełnia wzorców wykorzystujących atrybuty a_2 i a_4, np. $a_2 \neq *$ oraz $a_1 \neq * \wedge a_2 \neq *$ nie są spełniane przez obiekt x.

Od tej pory wzorce wypełnienia będą nazywane po prostu wzorcami. Dla każdego obiektu istnieje jeden szczególny wzorzec zwany schematem wypełnienia, który opisuje wszystkie wypełnione wartości danego obiektu.

Definicja 7.2 *Schemat wypełnienia obiektu.*

Schematem wypełnienia obiektu x nazwiemy taki wzorzec s_x, który posiada maksymalną liczbę deskryptorów wypełnienia.

$$s_x = \bigwedge_{a \in A \wedge a(x) \neq *} a \neq * \tag{7.1}$$

Posługując się wzorcami możemy łatwo definiować podzbiory uniwersum obiektów, które cechują się podobnym wypełnieniem wartości atrybutów. Oznaczmy przez P_t zbiór obiektów, które spełniają wzorzec t.

$$P_t = \{x \in U : x \models t\} \tag{7.2}$$

Możemy zatem w pewien sposób utożsamiać wzorzec z obiektami, które go spełniają.

Wzorce charakteryzuje się za pomocą tzw. gabarytów wzorca. Termin ten ma swoje intuicyjne uzasadnienie, gdy zwizualizujemy obiekty spełniające wzorzec w postaci tabeli.

Definicja 7.3 *Szerokość wzorca.*

Szerokością wzorca t nazwiemy liczbę deskryptorów wchodzących w skład wzorca i oznaczymy w_t. Wzorzec t_1 z poprzedniego przykładu posiada szerokość równą zero, $w_{t_2} = w_{t_3} = 1$ oraz $w_{t_4} = 2$.

Definicja 7.4 *Wysokość wzorca.*

Wysokością wzorca t nazwiemy liczbę obiektów spełniających wzorzec i oznaczymy h_t. Zatem $h_t = |P_t|$.

Teraz możemy ściśle wyrazić naszą intuicję dotyczącą podziału danych wejściowych na podtabele. Ponieważ podtabela taka nie może zawierać żadnych brakujących wartości, więc składa się z obiektów pasujących do wzorca zawierającego deskryptory wypełnienia dla wszystkich kolumn tej tabeli. Ponadto zależy nam na tym, żeby szerokość takiego wzorca była jak największa. Umożliwia to wykrycie zależności pomiędzy atrybutami warunkowymi, a atrybutem decyzyjnym. Jednocześnie liczba obiektów tej tabeli, czyli wysokość wzorca, nie może być zbyt mała, aby na jej podstawie można było się wyuczyć żądanej klasyfikacji. W oczywisty sposób oba te warunki są przeciwstawne i niezbędnym jest wypracowanie pewnego kompromisu. Szczegółowy opis metod poszukiwania wzorców opisany będzie w podrozdziale 7.6.

7.5 Opis algorytmu

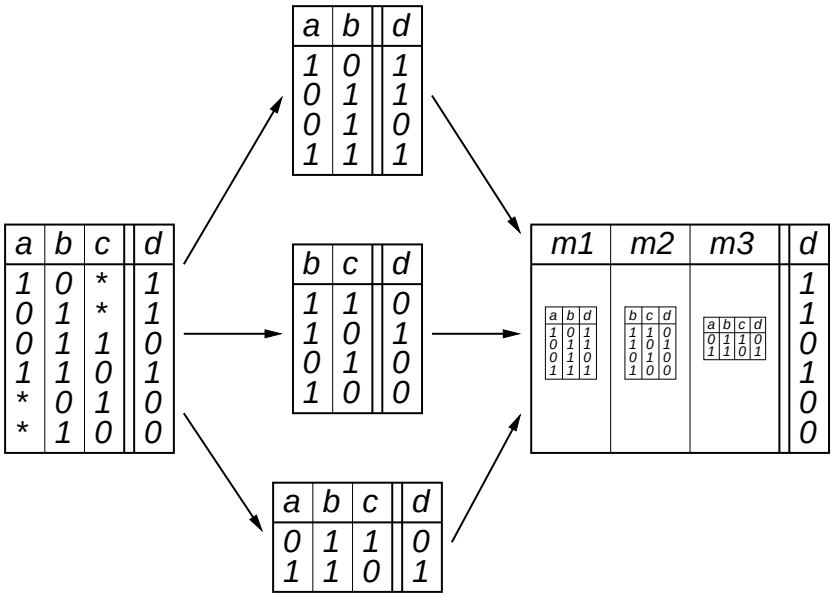
Metoda podziału składa się z dwóch podstawowych etapów. Na początku należy dokonać podziału danych wejściowych a następnie syntezy wyników.

Algorytm 7.1 *Metoda podziału.*

1. *Podział*
2. *Synteza wyników*

7.5.1 Podział

Celem podziału jest uzyskanie pewnej liczby podtabel posiadających określone cechy. Tabele powstające w wyniku podziału danych wejściowych nie mogą zawierać żadnych brakujących wartości. Jest to warunek, który musi zostać bezwzględnie spełniony, aby taki podział był poprawny. Ponadto tabele takie powinny umożliwiać skuteczne wnioskowanie indukcyjne, co może być osiągnięte np. przez zagwarantowanie odpowiednio dużych gabarytów takich tabel (tzn. wysokości i szerokości, czyli liczby obiektów i liczby atrybutów) oraz poprzez wykorzystanie możliwie największej liczby wartości z tabeli wejściowej. Są to dwa podstawowe kryteria oceny skuteczności podziału. Oprócz tego istnieją jeszcze pewne własności podziałów, które nie wpływają bezpośrednio na jakość wnioskowania. Na przykład liczba podtabel powstających z takiego podziału ma zdecydowany wpływ na szybkość



Rysunek 7.3: Metoda podziału polega na dekompozycji wejściowych danych na podtablice o kompletnym opisie obiektów, zastosowaniu klasyfikatora na podtablicach, a następnie syntezie wyników końcowych na podstawie podmodeli.

klasyfikacji. Dlatego korzystniej jest dzielić dane wejściowe na mniejszą liczbą podtabel. Przeciwnym argumentem, jest hipoteza statystyczna, że większa liczba podtabel może zagwarantować lepszą jakość podczas syntezy wyników. Hipoteza ta została zweryfikowana empirycznie i, jak pokażą wyniki eksperymentów, nie znajduje zastosowania w tym przypadku.

Wynikiem podziału jest pewna liczba tabel o kompletnym opisie obiektów, które podlegają następnie procesowi wnioskowania i syntezy wyników. Przyjmuje się również założenie, że wszystkie obiekty zawarte w tabeli wejściowej muszą zostać zaklasyfikowane do przynajmniej jednej z powstałych podtabel. Ponieważ zagadnienie podziału jest kluczowym elementem mającym wpływ na jakość wnioskowania zostanie omówione od strony algorytmicznej w podrozdziale 7.6.

7.5.2 Synteza wyników

Końcowym rezultatem każdego systemu decyzyjnego jest klasyfikacja obiektów do poszczególnych pojęć — klas decyzyjnych. W tym przypadku dysponujemy nie jedną, ale wieloma tabelami informacyjnymi. Co więcej, obiekty posiadające różne schematy wypełnienia rozproszone są pomiędzy różne tabele. W skrajnym przypadku poszczególne obiekty mogą być elementami tylko jednej podtabeli powstałej z podziału, dlatego w procesie wnioskowania musimy uwzględnić wszystkie podtabele.

Syntezę wyników przeprowadzimy w dwóch krokach. Inspiracji do zastosowania takiej metody można poszukiwać w metodzie uzupełniania za pomocą systemów decyzyjnych, obliczeń na granulach, czy nawet tak odległemu zagadnieniu jakim są wielowarstwowe sieci neuronowe. Na każdej z podtabel (patrzy rys. 7.2 i 7.3) dokonujemy niezależnej konstrukcji lokalnego modelu pojęcia za pomocą sytemu decyzyjnego. Lokalność modeli polega tutaj

na ograniczeniu informacji do pewnego podzbioru atrybutów i obiektów, które są całkowicie wypełnione na danym podzbiorze atrybutów. Drugim krokiem jest zastosowanie systemu decyzyjnego łączącego wyniki częściowe z każdego modelu lokalnego. Odpowiada to konstrukcji nowej tabeli informacyjnej, gdzie atrybutami są podmodele lokalne, a wartościami atrybutów jest klasyfikacja obiektów z tabeli wejściowej do pewnego pojęcia lub odmowa takiej klasyfikacji spowodowana tym, że obiekt nie należy do dziedziny danego podmodelu. Na takiej tablicy dokonywana jest ostateczna klasyfikacja wszystkich obiektów do określonych klas decyzyjnych.

Każdy lokalny system decyzyjny może być postrzegany jako specjalista w swojej dziedzinie. Zawsze do sklasyfikowania przyjmuje w pełni uzupełnione obiekty na określonym podzbiorze atrybutów i może wypracować hipotezę opisującą pojęcie na swoim wycinku wiedzy. Synteza, oparta na systemie decyzyjnym łączącym odpowiedzi częściowe rozstrzyga ewentualne konflikty pomiędzy specjalistami. Jej zadaniem jest wyuczenie się, który ze specjalistów lepiej sprawdza się na określonym podzbiorze obiektów, wyznaczonym czasem przez dość skomplikowane formuły logiczne operujące na spełnianiu lub nie spełnianiu przez obiekt określonych wzorców[1].

Algorytm 7.2 *Synteza wyników.*

1. *Zastosowanie niezależnych systemów decyzyjnych do wyznaczonych wcześniej podtabel.*
2. *Konstrukcja tabeli informacyjnej łączącej wyniki częściowe.*
3. *Zastosowanie systemu decyzyjnego udzielającego odpowiedzi dla wszystkich obiektów wejściowych.*

Pozostałym do rozstrzygnięcia zagadnieniem, jest wybór metod klasyfikacji na każdym z kroków syntezy wyników. Teoretycznie, można by w dość dowolny sposób dobierać niezależnie od siebie metody klasyfikacji dla podtabel powstałych z podziału oraz tabeli łączącej wyniki częściowe. Jednakże nie widać powodu, dla którego warto narażać się na takie komplikacje. Metoda podziału projektowana była jako środek zaradczy, umożliwiający zastosowanie istniejących zaawansowanych i zaimplementowanych metod klasyfikacji. Ich siła wyrazu, czyli zdolność do konstrukcji złożonych hipotez, jest na tyle duża, że z powodzeniem można je stosować w każdym kroku syntezy wyników. Warto tutaj tylko zauważyć, że o ile konkretna klasyfikacja na etapie lokalnych podtabel nie jest tak istotna (można by wręcz zastosować klasyfikację do jakiś pojęć pomocniczych), to łączenie wyników częściowych musi opierać się na metodzie, która potrafi konstruować wystarczająco zaawansowane hipotezy do rozstrzygania ewentualnych konfliktów.

7.6 Podział danych wejściowych

Podział danych wejściowych na podtabele jest sam w sobie zagadnieniem skomplikowanym. Ponadto w decydującym stopniu przyczynia się do uzyskanych wyników. Jak pokazane to zostało w podrozdziale 7.4, każdą podtabelę możemy utożsamiać z pewnym wzorcem t, a

[1] W zależności do zastosowanego systemu decyzyjnego. Np. dla klasyfikatora regułowego będą to koniunkcje spełniania lub nie spełniania przez obiekt wzorców wyznaczonych w fazie podziału.

raczej zbiorem obiektów spełniających ten wzorzec P_t. Zatem wyznaczanie podziałów danych wejściowych to nic innego jak wyszukiwanie wzorców o pożądanych własnościach. Opis metod wyszukiwania podziałów rozpoczniemy charakteryzacji złożoności obliczeniowej problemu.

7.6.1 Złożoność obliczeniowa

Większość problemów związanych z wyszukiwaniem pojedynczego wzorca (ogólnego) zawiera się w klasie problemów NP-trudnych. W szczególności klasyczny problem wyszukiwania wzorca o maksymalnych gabarytach zdefiniowanych jako szerokość×wysokość jest NP-trudnym problemem optymalizacyjnym (zobacz np. [33, 35, 34]). W przypadku wyszukiwania wzorców wypełnienia możemy posłużyć się analogią takiego zadania do wyszukiwania wzorców ogólnych np. w tabelach gdzie wszystkie atrybuty mają dwuelementową dziedzinę (atrybuty binarne).

Twierdzenie 7.1

Problem wyszukiwania wzorca wypełnienia jest NP-trudny, o ile odpowiadający mu problem wyszukiwania wzorca ogólnego również jest NP-trudny[2].

Dowód

Jest oczywiste, że problem wyszukiwania wzorca wypełnienia zawarty jest w klasie problemów NP. Wystarczy zatem pokazać, że za pomocą wielomianowego sprowadzenia potrafimy algorytmem wyszukiwania wzorców wypełnienia rozwiązać problem wyszukiwania wzorców ogólnych.

Weźmy tablicę informacyjną $\mathbb{A} = (U, A)$ taką, że $A = \{a_1, \ldots, a_n\}$, oraz $\forall_i V_{a_i} = \{0, 1\}$. W czasie wielomianowym możemy skonstruować tablicę $\mathbb{A}' = (U, A')$ taką, że $A' = \{a_1^0, a_1^1, \ldots, a_n^0, a_n^1\}$. Wartości atrybutów zdefiniowane są następująco:

$$a_i^0(x_j) = \begin{cases} 0 & : a_i(x_j) = 0 \\ * & : a_i(x_j) = 1 \end{cases}, \tag{7.3}$$

$$a_i^1(x_j) = \begin{cases} * & : a_i(x_j) = 0 \\ 1 & : a_i(x_j) = 1 \end{cases}. \tag{7.4}$$

Złożoność tej konwersji jest wielomianowa i wynosi $O(|A|)$. Stosujemy algorytm wyszukiwania wzorców wypełnienia na tabeli $\mathbb{A}'$ i dostajemy rozwiązanie t'. Teraz wystarczy pokazać, jak dokonać konwersji rozwiązania t' dla tabeli $\mathbb{A}'$ na rozwiązanie t dla tabeli $\mathbb{A}$.

Przy wyszukiwaniu wzorca o maksymalnych gabarytach (odpowiednio zdefiniowanych) nigdy nie zostaną jednocześnie wybrane atrybuty a_i^0 oraz a_i^1 dla żadnego i. Jest tak dlatego, że żaden obiekt nie spełnia wzorca wypełnienia zawierającego jednocześnie a_i^0 oraz a_i^1. Zatem dla każdego i w znalezionym wzorcu wypełnienia istnieje co najwyżej jeden deskryptor zawierający a_i^0 lub a_i^1. Wzorzec (ogólny) dla tabeli $\mathbb{A}$ powstaje w następujący sposób. Każdy deskryptor wypełnienia wzorca t' postaci $a_i^v \neq *$ zamieniamy na deskryptor wzorca t postaci $a_i = v$. W ten oto sposób otrzymamy rozwiązanie t dla tabeli $\mathbb{A}$, które posiada dokładnie te same gabaryty co rozwiązanie t' dla tabeli $\mathbb{A}'$.

[2]Istnieją problemy wyszukiwania wzorca rozwiązywalne w czasie wielomianowym, np. gdy poszukujemy wzorca o największych gabarytach zdefiniowanych jako szerokość + wysokość (patrz [34] str. 19).

Q.E.D

Oprócz zagadnienia wyszukiwania jednego wzorca może nas również interesować zagadnienie wyszukiwania wielu wzorców jednocześnie. Ma to swoje uzasadnienie przy próbie wygenerowania wszystkich wzorców stanowiących podział danych wejściowych na podtabele. Problem ten jest co najmniej tak trudny, jak wyszukiwanie jednego wzorca. Zatem aby pokazać, że należy do klasy problemów NP-trudnych wystarczy pokazać, że jest problemem klasy NP. Ponieważ nie zdefiniowane zostały jeszcze dokładne kryteria wyboru takich wzorców, posłużymy się poniższym faktem do pokazania, że bardzo szeroka klasa problemów decyzyjnych związanych z wyszukiwaniem wielu wzorców mieści się w klasie NP.

Fakt 7.1
Mając zadaną tablicę informacyjną i zbiór wzorców można w wielomianowym czasie sprawdzić czy:

1. *wzorce pokrywają wszystkie obiekty,*
2. *wzorce pokrywają wszystkie atrybuty,*
3. *wzorce posiadają określone gabaryty będące dowolną funkcją*[3] *wysokości i szerokości,*
4. *każdy obiekt jest pokryty przez zadaną liczbę wzorców,*
5. *liczba „ominiętych" przez wzorce istniejących wartości atrybutów jest mniejsza od zadanej.*

7.6.2 Wyszukiwanie wielu wzorców

Podstawowym pomysłem na wygenerowanie pożądanego podziału jest znalezienie rodziny wzorców pokrywającej łącznie wszystkie obiekty i posiadającej dodatkowe, pożądane cechy. Standardowymi wymaganiami może być tutaj, aby wzorce posiadały jak największe gabaryty w sensie szerokość$\times$wysokość lub szerokość$^2\times$wysokość. Oprócz tego, możemy żądać, żeby liczba istniejących wartości atrybutów nie pokrytych przez żaden obiekt była minimalna. Mówimy wtedy o tzw. „ominiętych" wartościach. Może tak się zdarzyć, gdy suma deskryptorów wzorców spełnianych przez dany obiekt jest mniejsza niż deskryptory schematu wzorca.

Zadanie wygenerowania kompletnej rodziny wzorców o zadanych własnościach jest skomplikowanym problemem. Nie jest to jednak zagadnienie zupełnie nowe. W podobnych problemach, jak np. wyszukiwanie zbioru pokrywających reguł decyzyjnych czy reguł asocjacyjnych również występuje problem pokrycia całej tabeli informacyjnej pewną liczbą wzorców (patrz np. [33, 34]). Nie istnieje jednak dobre rozwiązanie algorytmiczne, które umożliwiało by aproksymację tego problemu NP-trudnego w sposób bezpośredni. Praktycznie wszystkie problemy tego typu rozwiązywane są poprzez iteracyjne, zachłanne pokrywanie coraz większej liczby obiektów tabeli wejściowej.

Istnieje co prawda uniwersalna metoda optymalizacyjna, która umożliwiła by rozwiązanie takiego zadania w sposób bezpośredni. Algorytmy genetyczne — bo o nich mowa, umożliwiają optymalizację prawie dowolnej funkcji. Należy się jednak zastanowić nad realnością i efektywnością takiego rozwiązania.

[3] Ale taką funkcją, którą można obliczyć w czasie wielomianowym dysponując wartościami argumentów

Po pierwsze, chcąc zastosować algorytm genetyczny musimy zdefiniować kodowanie osobników i operatory genetyczne. Niemniej jednak w niniejszym zadaniu nie mamy zadanej z góry liczny wzorców wchodzących w skład rodziny. Utrudnia to w sposób znaczący implementację i, co ważniejsze, niekorzystnie wpływa na takie parametry algorytmu genetycznego jak zbieżność, czy generowanie osobników należących do dziedziny poprawnych rozwiązań.

Drugim aspektem jest stopień swobody rozwiązania, czyli liczba zmiennych. Jak pokazuje doświadczenie w badaniu algorytmów genetycznych, gdy stopień swobody przekracza pewną dużą liczbę[4], rzędu 100–1000, algorytmy genetyczne zaczynają generować rozwiązania dużo bardziej odległe od rozwiązania optymalnego oraz zaczynają mieć problemy ze zbieżnością od rozwiązań gorszych do lepszych.

Ta krótka charakterystyka sugeruje, że do rozwiązywania tego zadania należy zastosować standardowe i o dobrze poznanych własnościach algorytmy zachłannego, iteracyjnego generowania kolejnych wzorców. Jak pokażą wyniki eksperymentalne zaimplementowanie ew. metody generującej całościowe rozwiązanie w jednym przebiegu nie może znacząco wpłynąć na liczbę wygenerowanych wzorców jak i również na ostateczną klasyfikację, gdyż liczba wzorców wygenerowanych za pomocą metody zachłannej jest już wystarczająco niewielka.

7.6.3 Zachłanna konstrukcja pokrycia

Algorytm zachłannego generowania pokrycia wzorcami jest dobrze znaną i skuteczną metodą aproksymacyjną rozwiązywania tego problemu.

Algorytm 7.3

Mamy daną tabelę informacyjną $\mathbb{A}$, oraz algorytm $\mathcal{P}$ wyszukiwania optymalnego wzorca[5] *t dla zadanej tabeli informacyjnej.*

1. $\mathbb{A}_0 := \mathbb{A}$, $i = 0$
2. $t_i := \mathcal{P}(\mathbb{A}_i)$
3. $\mathbb{A}_{i+1} := \mathbb{A}_i \setminus P_{t_i}$[6]
4. $i := i + 1$
5. *Jeśli $\mathbb{A}_i = \emptyset$ zakończ. W przeciwnym przypadku przejdź do 3.*

Algorytm generuje kolejno najlepsze wzorce dla danej tabeli, po czym usuwa wszystkie pokryte już obiekty i wyszukuje kolejnego najlepszego wzorca dla pozostałych elementów. Oczywiście kolejno wygenerowane wzorce mogą również pokrywać elementy uprzednio wyrzucone, niemniej jednak nie ma to wypływu na ocenę wzorca podczas zastosowania algorytmu $\mathcal{P}$ wyszukiwania jednego wzorca.

[4] Liczbę 100 możemy w tym przypadku traktować jako dużą, ponieważ najczęściej towarzyszy jej rozmiar przestrzeni rozwiązań co najmniej 2^{100}.

[5] Najczęściej jest to aproksymacja wzorca optymalnego.

[6] P_{t_i} oznacza zbiór obiektów spełniających wzorzec t_i (patrz roz. 7.4).

Istniejące również modyfikacje tego algorytmu, nie usuwające permanentnie już pokrytych obiektów, a tylko zmniejszające ich znaczenie podczas wyboru wzorca poprzez zastosowanie ważenia obiektów. Modyfikacje takie zostały przebadane eksperymentalnie dla różnych metod ważenia obiektów, jednakże uzyskane wyniki okazały się być zdecydowanie gorsze. Dalsze eksperymenty przeprowadzone zostały tylko i wyłącznie dla powyżej opisanego algorytmu, odpowiadającego zmniejszeniu wagi raz pokrytego obiektu do zera.

7.7 Algorytmy wyszukiwania wzorca

Wyszukiwanie wzorca jest zagadnieniem wystarczająco trudnym i tak często spotykanym, że samo w sobie stanowi osobną dziedzinę inteligentnego przetwarzania informacji. Na przestrzeni lat dopracowano się różnych skutecznych metod aproksymacji rozwiązania optymalnego. Czasem stosuje się również metody dokładne, przeprowadzające analizę wszystkich możliwych wzorców, co prowadzi do wykładniczej złożoności obliczeniowej.

Przede wszystkim należy zdefiniować pojęcie wzorca optymalnego lub najlepszego. Celem metody podziału jest uzyskanie jak najlepszej klasyfikacji obiektów, zatem podtabele powstałe w wyniku podziału powinny umożliwiać skuteczne wnioskowanie indukcyjne. Skuteczne wnioskowanie może zostać uniemożliwione, gdy nie dysponujemy zbyt małym zbiorem atrybutów, aby zachodziła chociażby częściowa zależność atrybutu decyzyjnego od tego podzbioru atrybutów. Również niewystarczająca liczba obiektów może uniemożliwić wybranie prawidłowej hipotezy opisującej pojęcie. Naturalną oceną wzorca wdaje się zatem standardowa funkcja jakości wzorca postaci szerokość$\times$wysokość. Czasami stosuje się również inne modyfikacje, jak szerokość$^2\times$wysokość itp. Badania eksperymentalne pokazały jednak, że różnice w liczbie znalezionych wzorców były nieduże, a co najważniejsze, w ostatecznej klasyfikacji wyniki nie różniły się zbytnio w zależności od przyjętej funkcji jakości, w zakresie szerokość$^1\times$wysokość, ..., szerokość$^4\times$wysokość.

Eksperymenty uwidoczniły jednak niedoskonałość takiego podejścia. Podczas wyszukiwania wzorców znajdowano dużo wzorców o podobnych gabarytach, jednakże dających drastycznie różne wyniki klasyfikacji. Należy sobie zatem zadać pytanie, dlaczego taką cechę danych, jaką jest możliwość przeprowadzenia dokładnego wnioskowania mierzymy gabarytami wzorców, a nie w sposób bezpośredni.

Definicja 7.5 *Jakość predykcyjna wzorca.*

Jakością predykcyjną wzorca t dla danej metody M nazwiemy współczynnik poprawnych odpowiedzi klasyfikacji metodą M danych testowych obciętych do wzorca. Dane treningowe również podlegają procesowi obcięcia do wzorca.

Obcięcie danych do wzorca oznacza, że zarówno do zbioru danych treningowych jak i testowych wybieramy tylko obiekty spełniające wzorzec t, a zbiór atrybutów warunkowych zawężamy do atrybutów występujących w deskryptorach wypełnienia wzorca t.

Podczas oceny jakości wzorca można w bezpośredni sposób użyć jakości predykcyjnej wzorca. Należy jednak mieć na uwadze, że proces ewaluacji tej wartości jest długi i posiada złożoność obliczeniową rzędu co najmniej $O(KN\log N)$[7], gdzie K to liczba atrybutów, a N to liczba obiektów. Do ostatecznej oceny jakości wzorca można również zastosować funkcję uwzględniającą zarówno jakość predykcyjną wzorca jak i jego gabaryty.

[7] Złożoność obliczeniowa konstrukcji klasyfikatora zależy również od rozmiaru dziedzin wartości atrybutów

W publikacjach [34, 35] zaprezentowane zostały efektywne algorytmy, deterministyczny Max I i randomizowany Max II, które cechuje szybki czas działania i duża skuteczność aproksymacji, większa od prostego algorytmu genetycznego. Niestety algorytmy te nie mogły być wykorzystane eksperymentów, gdyż ich konstrukcja bazuje na wyszukiwaniu wzorców w oparciu o gabaryty zdefiniowane jako szerokość$\times$wysokość. Eksperymenty przeprowadzane były również dla funkcji opartych o jakość predykcyjną wzorca. Funkcje takie nie mogą być optymalizowane za pomocą wyżej wspomnianych algorytmów.

7.7.1 Algorytmy genetyczne

Algorytmy genetyczne to dobrze rozwinięta dziedzina sztucznej inteligencji. Ze szczegółowym opisem zasad działania i metod projektowania algorytmów genetycznych można zapoznać się np. w pracach [9, 16, 31]. Algorytmy te należą do skutecznych metod optymalizacji, które potrafią z powodzeniem optymalizować nawet najbardziej skomplikowane funkcje. Należy jednak zwracać uwagę na sposób zmienności optymalizowanej funkcji oraz na reprezentację rozwiązania w postaci genotypu. Od właściwego dobrania reprezentacji i parametrów algorytmu genetycznego zależy szybkość zbieżności do rozwiązania suboptymalnego i jakość tego rozwiązania.

Charakter zmienności funkcji jakości wzorca jest dosyć szczególny i zastosowanie prostego algorytmu genetycznego do wyszukiwania wzorców może nie przynieść zadowalających rezultatów. Aby uzyskać algorytm genetyczny odpowiadający naszym oczekiwaniom należy go nieco przeprojektować.

Algorytm 7.4 *Genetyczny algorytm wyszukiwania wzorców.*

1. *Utwórz populację początkową ze wszystkich schematów wypełnienia występujących w tabeli.*
2. *Za pomocą operatorów genetycznych utwórz populację potomną o liczbie osobników* $3P$.
3. *Zastosuj selekcję do całej grupy* $4P$ *osobników (najlepiej ruletkową lub turniejową) w celu uzyskania następnej populacji o liczbie osobników* P.
4. *Powtarzaj od 2. zadaną liczbę iteracji.*

Algorytm ten różni się istotnie od klasycznych algorytmów genetycznych. Ze względu na kolejność zastosowania operatorów genetycznych i selekcji przypomina on nieco metody ewolucyjne. Również istotną modyfikacją jest częściowo zmienna wielkość populacji podczas różnych faz algorytmu. Także operatory genetyczne zostały indywidualnie zaprojektowane do rozwiązywania problemu wyszukiwania wzorców wypełnienia. Jako operatory genetyczne zastosowano również proste operacje teoriomnogościowe umożliwiające duże skoki w przestrzeni rozwiązań jednocześnie zachowujące własności osobników wejściowych. Zastosowane operatory:

- mutacja jednorodna,
- przecięcie,
- suma,

- krzyżowanie jednorodne.

Algorytm tej postaci doskonale nadaje się do wyszukiwania najlepszego wzorca niezależnie od stopnia skomplikowania funkcji jakości wzorca. Sterując parametrami algorytmu, czyli wielkością populacji P, liczbą iteracji oraz prawdopodobieństwami użycia operatorów genetycznych, możemy wyznaczyć empirycznie ustawienia gwarantujące dobre rozwiązania.

7.7.2 Optymalizacja wyszukiwania wzorca

Zastosowanie algorytmów wyszukiwania wzorca może być czasochłonne. Można jednak zredukować czas wyszukiwania dokonując kilku prostych optymalizacji.

Podstawową metodą, jaką należy zastosować w celu zredukowania czasu wykonania jest tzw. kompresja tabeli. Kompresja tabeli polega na utworzeniu tabeli pomocniczej w czasie $O(N \log N)$, w której zawarte będą schematy wszystkich obiektów wraz z liczebnością ich wystąpienia. Jak pokazały doświadczenia kompresja taka redukuje liczbę wierszy tabeli do wielkości porównywalnych z K (liczbą atrybutów). Nawet dla dużych tabel liczba występujących schematów nie przekracza zazwyczaj 100–200 różnych schematów. Wyznaczenie wysokości wzorca (liczby obiektów spełniających wzorzec) odbywa się wtedy nieporównanie szybciej, niż na tabeli wejściowej. Zastosowanie kompresji tabeli, pomimo wstępnego, przetwarzania pozwala na bardzo duże oszczędności czasowe.

Kolejną metodą godną polecenia jest zapamiętywanie wyników częściowych. Szczególnie jest to istotne podczas optymalizacji funkcji jakości zależnej od jakości predykcyjnej wzorca. Obliczenie takiej wartości jest bardzo czasochłonne i zastosowanie np. prostej tablicy haszującej zawierającej wartości jakości predykcyjnej już sprawdzanych wzorców przynosi duże oszczędności czasowe. Jest to istotne przy użyciu algorytmów genetycznych. Jeśli używany przez nas algorytm genetyczny został dobrze zaprojektowany, wtedy charakteryzuje się szybką zbieżnością do rozwiązania suboptymalnego. Liczba istotnie różnych wzorców jest wtedy dwa lub więcej razy mniejsza niż liczba wszystkich osobników poddanych sprawdzeniu.

7.7.3 Podsumowanie

W tej chwili dysponujemy już pełnym opisem metody podziału. Dwa podstawowe etapy tej metody to dekompozycja i synteza wyników. Dekompozycja to wygenerowanie podziałów, czyli wzorców określających podtablice. Podziały generowane są iteracyjnie za pomocą algorytmu zachłannego, który wykorzystuje algorytm wyszukiwania jednego wzorca, genetyczny lub inny. Dysponując podziałami, stosujemy algorytm wnioskowania indukcyjnego na podtablicach, a wyniki zapisujemy do tabeli łączącej wyniki częściowe. Ponownie stosujemy algorytm wnioskowania indukcyjnego, tym razem do tabeli łączącej wyniki i uzyskujemy ostateczny klasyfikator wszystkich obiektów tabeli wejściowej.

7.8 Opis eksperymentów

Teoretyczna analiza algorytmów nie zawsze okazuje się stosowna w zetknięciu z rzeczywistością. Tym bardziej, że nie potrafimy tutaj przewidzieć dokładności wyników, gdyż w

bardzo szczególny sposób zależą one od danych wejściowych. Każdą metodę w dziedzinie analizy danych należy również sprawdzić empirycznie. Jest to bardzo popularna metoda postępowania. W zasadzie dla wszystkich metod istnieją publikacje dokumentujące osiągane wyniki, a dane na których testy te były wykonane znajdują się w ogólno dostępnych repozytoriach stworzonych właśnie w tym celu. Istotnym zatem elementem pracy jest gruntowne przetestowanie metody podziału w celu porównania jej wyników z innymi dostępnymi metodami.

Wyniki eksperymentów uzyskano stosując metodę testowania klasyfikatorów **CV5**. Metoda ta polega na podzieleniu zbioru danych na 5 równolicznych i rozłącznych podzbiorów. Podczas klasyfikacji $\frac{4}{5}$ danych traktuje się jako dane treningowe, a $\frac{1}{5}$ jako dane testowe. Cały proces powtarzany jest pięciokrotnie tak, aby wykorzystać wszystkie możliwości przydziału 4 podzbiorów do zbioru treningowego, a jeden pozostały użyć jako zbiór testowy. Jako wynik końcowy podaje się średnią z pięciu prób klasyfikacji danych. Metoda CV5 (ang. cross validation) umożliwia dość dobre wyznaczenie sprawności klasyfikatora.

Niestety wynik CV5 może się nieco oscylować (w zakresie kilku procent) w zależności od dystrybucji elementów oryginalnej tabeli do pięciu podzbiorów. Aby wynik eksperymentu był miarodajny i powtarzalny każdy eksperyment został wykonany 100 razy dla różnych rozbić oryginalnej tabeli. Ostateczne wyniki pochodzą z uśrednienia wyników każdego z eksperymentów. Ma to na celu zapobiec ewentualnemu zaburzeniu wyników przez mniej lub bardziej sprawiedliwy podział danych na zbiór testowy i treningowy.

7.8.1 Algorytmy

Eksperymenty przeprowadzane były za pomocą 11 różnych algorytmów. Wszystkie algorytmy oprócz **C4.5** są konkretną realizacją metody podziału. Jako klasyfikator w etapie syntezy wyników wykorzystany został algorytm C4.5. Ponieważ podtabele powstające w etapie dekompozycji (podziału) nie zawierają żadnych brakujących wartości, umożliwia to dokładne porównanie zachowania się metod radzenia sobie z brakującymi wartościami w algorytmie C4.5 i w różnych implementacjach metody podziału. Do porównania celowo została wybrana metoda C4.5, gdyż uchodzi ona za jedną z najlepszych metod zarówno klasyfikacji, jak i radzenia sobie z brakującymi wartościami.

Opis algorytmów będących implementacją metody podziału ogranicza się tylko i wyłącznie do charakterystyki użytej metody generowania podziałów. Synteza wyników jest taka sama dla wszystkich algorytmów i opiera się na metodzie C4.5.

- **J48** — odpowiednik algorytmu **C4.5** opracowanego przez J. R. Quinlana. Algorytm ten był opisywany w podrozdziale 4.1.
- **all** — wszystkie schematy wypełnienia.
- **exact** — dokładny algorytm sprawdzający wszystkie 2^n wzorców. Wzorce najlepsze wybierane są na podstawie jakości określonej $q(t) = w_t \cdot h_t$.
- **ga50** — algorytm genetyczny wykonujący 50 iteracji dla populacji o zmiennej liczbie 50–200. Funkcja jakości $q(t) = w_t \cdot h_t$.
- **ga20** — algorytm genetyczny wykonujący 20 iteracji dla populacji o zmiennej liczbie 20–80. Funkcja jakości $q(t) = w_t \cdot h_t$.

p_1	0.25	0.30	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90
p_2	0.35	0.40	0.45	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95	1.00
α	2.06	2.41	2.76	3.11	3.45	3.80	4.15	4.50	4.84	5.19	5.54	5.88	6.23	6.58

Tablica 7.1: Jeśli sprawność wzorca wynosi p_2, to liczba p_2^α jest dwa razy większa niż p_1^α. Oznacza to, że względem miary $q = w \cdot h \cdot p^\alpha$ wzorzec posiadający sprawność p_2 może mieć prawie dwa razy mniejsze gabaryty $w \cdot h$, niż wzorzec posiadający sprawość p_1, a mimo tego będzie oceniony jako lepszy.

- **ga10** — algorytm genetyczny wykonujący 10 iteracji dla populacji o zmiennej liczbie 10–40. Funkcja jakości $q(t) = w_t \cdot h_t$.
- **ev1** — algorytm genetyczny wykonujący 10 iteracji dla populacji o zmiennej liczbie 10–40. Funkcja jakości $q(t) = w_t \cdot h_t \cdot p_t^1$, gdzie p_t oznacza jakość predykcyjną wzorca (sprawność wyrażoną w zakresie $[0, 1]$, wyliczoną na podstawie wstępnej klasyfikacji obiektów pasujących do tego wzorca).
- **ev2** — algorytm genetyczny wykonujący 10 iteracji dla populacji o zmiennej liczbie 10–40. Funkcja jakości $q(t) = w_t \cdot h_t \cdot p_t^2$.
- **ev4** — algorytm genetyczny wykonujący 10 iteracji dla populacji o zmiennej liczbie 10–40. Funkcja jakości $q(t) = w_t \cdot h_t \cdot p_t^4$.
- **ev8** — algorytm genetyczny wykonujący 10 iteracji dla populacji o zmiennej liczbie 10–40. Funkcja jakości $q(t) = w_t \cdot h_t \cdot p_t^8$.
- **ev** — algorytm genetyczny wykonujący 10 iteracji dla populacji o zmiennej liczbie 10–40. Funkcja jakości $q(t) = p_t$.

Użycie algorytmu all było spowodowane chęcią zweryfikowania hipotezy, że większa liczba podziałów może mieć wpływ na poprawę wyniku. Algorytmy exact, ga50, ga20 i ga10 umożliwiają ocenę zastosowanego algorytmu genetycznego w zależności od liczby iteracji i wielkości populacji w porównaniu do algorytmu dokładnego, wykonującego wykładniczą liczbę sprawdzeń. Porównanie wyników algorytmów ga10, ev1, . . ., ev8 i ev pozwala ocenić wpływ użycia jakości predykcyjnej wzorca na zachowanie się całego procesu wnioskowania (patrz tabela 7.1). Należy przypomnieć, że $w_t \cdot h_t$ jest tylko heurystyką aproksymującą przydatność wzorca do procesu wnioskowania. Wartość p_t z pewnością jest bliższa nieznanej funkcji przydatności, niemniej jednak jest też dużo bardziej czasochłonna do wyznaczenia.

7.8.2 Tabele

Do eksperymentów wykorzystano 12 zbiorów danych pochodzących z ogólno dostępnych repozytoriów danych do celów badań nad sztuczną inteligencją. Głównym kryterium wyboru tabel informacyjnych była znaczna liczba brakujących wartości atrybutów, w miarę równomiernie rozproszonych po całej tabeli.

Planowana implementacja algorytmów przewiduje używanie metod wnioskowania operujących tylko i wyłącznie na atrybutach symbolicznych. Dlatego jeśli w danych występowały również atrybuty numeryczne, do eksperymentów brane były dwie tabele informacyjne. Jedna tabela składała się z oryginalnych danych, a w drugiej wszystkie atrybuty numeryczne

traktowane były jako atrybuty symboliczne. Taką konwersję możemy interpretować, jako ignorowanie linowego porządku atrybutów numerycznych.

O ile nie zaznaczono inaczej, wszystkie poniżej wymienione tabele pochodzą z UCI Machine Learning Repository (patrz [5]).

- **att**$_n$ — tabela zawiera 1000 obiektów z 2 klas decyzyjnych, 1 atrybut numeryczny oraz 8 atrybutów symbolicznych. 24,4% obiektów posiada brakujące wartości.
- **att**$_s$ — tabela zawiera 1000 obiektów z 2 klas decyzyjnych, 1 atrybut numeryczny traktowany jako symboliczny oraz 8 atrybutów symbolicznych. 24,4% obiektów posiada brakujące wartości.
- **ban**$_n$ — tabela zawiera 540 obiektów z 2 klas decyzyjnych, 19 atrybutów numerycznych oraz 11 atrybutów symbolicznych. 48,3% obiektów posiada brakujące wartości.
- **ban**$_s$ — tabela zawiera 540 obiektów z 2 klas decyzyjnych, 19 atrybutów numerycznych traktowanych jako symboliczne oraz 11 atrybutów symbolicznych. 48,3% obiektów posiada brakujące wartości.
- **cmc2**$_n$ — tabela zawiera 1473 obiekty z 3 klas decyzyjnych, 2 atrybuty numeryczne oraz 7 atrybutów symbolicznych. 14,9% obiektów posiada brakujące wartości.
- **cmc2**$_s$ — tabela zawiera 1473 obiekty z 3 klas decyzyjnych, 2 atrybuty numeryczne traktowane jako symboliczne oraz 7 atrybutów symbolicznych. 14,9% obiektów posiada brakujące wartości.
- **dna2** — tabela zawiera 3186 obiektów z 3 klas decyzyjnych, 60 atrybutów symbolicznych. 14,1% obiektów posiada brakujące wartości.
- **hab2**$_n$ — tabela zawiera 306 obiektów z 2 klas decyzyjnych, 3 atrybuty numeryczne. 20,3% obiektów posiada brakujące wartości.
- **hab2**$_s$ — tabela zawiera 306 obiektów z 2 klas decyzyjnych, 3 atrybuty numeryczne traktowane jako symboliczne. 20,3% obiektów posiada brakujące wartości.
- **hco**$_n$ — tabela zawiera 368 obiektów z 2 klas decyzyjnych, 5 atrybutów numerycznych oraz 14 atrybutów symbolicznych. 89,4% obiektów posiada brakujące wartości.
- **hco**$_s$ — tabela zawiera 368 obiektów z 2 klas decyzyjnych, 5 atrybutów numerycznych traktowanych jako symboliczne oraz 14 atrybutów symbolicznych. 89,4% obiektów posiada brakujące wartości.
- **hep**$_n$ — tabela zawiera 155 obiektów z 2 klas decyzyjnych, 6 atrybutów numerycznych oraz 13 atrybutów symbolicznych. 48,4% obiektów posiada brakujące wartości.
- **hep**$_s$ — tabela zawiera 155 obiektów z 2 klas decyzyjnych, 6 atrybutów numerycznych traktowanych jako symboliczne oraz 13 atrybutów symbolicznych. 48,4% obiektów posiada brakujące wartości.
- **hin** — tabela zawiera 1000 obiektów z 3 klas decyzyjnych, 6 atrybutów symbolicznych. 40,5% obiektów posiada brakujące wartości.

- **hyp**$_n$ — tabela zawiera 3163 obiektów z 2 klas decyzyjnych, 6 atrybutów numerycznych oraz 9 atrybutów symbolicznych. 36,8% obiektów posiada brakujące wartości.
- **hyp**$_s$ — tabela zawiera 3163 obiektów z 2 klas decyzyjnych, 6 atrybutów numerycznych traktowanych jako symboliczne oraz 9 atrybutów symbolicznych. 36,8% obiektów posiada brakujące wartości.
- **pid2**$_n$ — tabela zawiera 768 obiektów z 2 klas decyzyjnych, 8 atrybutów numerycznych. 48,8% obiektów posiada brakujące wartości.
- **pid2**$_s$ — tabela zawiera 768 obiektów z 2 klas decyzyjnych, 8 atrybutów numerycznych traktowanych jako symboliczne. 48,8% obiektów posiada brakujące wartości.
- **smo2**$_n$ — tabela zawiera 2855 obiektów z 3 klas decyzyjnych, 3 atrybuty numeryczne oraz 5 atrybutów symbolicznych. 18,7% obiektów posiada brakujące wartości.
- **smo2**$_s$ — tabela zawiera 2855 obiektów z 3 klas decyzyjnych, 3 atrybuty numeryczne traktowane jako symboliczne oraz 5 atrybutów symbolicznych. 18,7% obiektów posiada brakujące wartości.
- **tumor** — tabela zawiera 339 obiektów z 22 klas decyzyjnych, 17 atrybutów symbolicznych.[8] 61,1% obiektów posiada brakujące wartości.

7.8.3 Implementacja

Jako podstawa do implementacji algorytmów wybrany został system analizy danych **Weka** [12] opisany w książce [61]. Wybór ten podyktowany został dostępnością pełnej implementacji algorytmu C4.5 wraz z jego dokładną dokumentacją. Oryginalna implementacja C4.5 pozbawiona jest dokumentacji technicznej, a w dodatku wykonana została w języku programowania C, więc nie nadaje się do łatwej modyfikacji i użycia wewnątrz innych programów. Jako, że Weka zaimplementowana została w języku Java, również do implementacji wszystkich opisanych wcześniej algorytmów użyty został ten język programowania. Pozwala to na szybką implementację eksperymentów oraz łatwą modyfikację zastosowanych rozwiązań. Oznacza to co prawda spowolnienie wykonania eksperymentów ok. 10 razy, jednak w dzisiejszych czasach, przy szerokiej dostępności dużych mocy obliczeniowych nie ma to aż tak dużego znaczenia. Dużą zaletą takiego rozwiązania jest również łatwość w uruchamianiu programu pod kontrolą różnych systemów operacyjnych.

Algorytmy implementujące różne warianty metody podziału wykonane zostały jako niezależna część programu, nie wymagająca ingerencji w kod źródłowy Weki, oraz korzystająca z własnych, zoptymalizowanych pod kątem eksperymentów struktur danych. Jedynym miejscem użycia systemu Weka był proces wnioskowania indukcyjnego metodą J48. Metoda J48 to pełna implementacja metody C4.5 Revision 8, która jest ostatnią niekomercyjną wersją rozwojową metody C4.5, przed wprowadzeniem metody C5.0 (patrz [42]).

Wszystkie eksperymenty wykonywane były na komputerach PC z procesorami AMD Duron 800Mhz lub Intel Pentium III 800Mhz pod kontrolą systemów operacyjnych Linux i Microsoft Windows. Czas wykonania eksperymentów zależał od użytego algorytmu.

[8] Dane dotyczące nowotworów pochodzą z Instytutu Onkologii w Centrum Medycznym Uniwersytetu w Ljubljanie dzięki życzliwości M. Zwittera oraz M. Skolica.

Różnice pomiędzy konfiguracjami konkretnych komputerów nie miały praktycznie żadnego wpływu na szybkość działania. Wykonanie pełnej serii eksperymentów dla najszybszej metody ga10 wyniosło 84 minuty, a dla najwolniejszej ev — 8 dni. Należy jednak pamiętać, że dla każdej z 21 tabel z danymi wykonywano stukrotne powtórzenie eksperymentu. Seria eksperymentów z metodą ev wykonaną tylko jednokrotnie zajęła by mniej niż 2 godziny przy wykorzystaniu wyżej opisanego sprzętu. Implementacja całego systemu w jednym z mniej uniwersalnych języków programowania, jak np. C, czy C++, pozwoliła by na jeszcze większe skrócenie tego czasu, do ok. 10–20 minut.

7.9 Wyniki eksperymentów

Eksperymenty przeprowadzano pod kontem weryfikacji przydatności metody podziału. Istotnym punktem badań, było stwierdzenie, która z implementacji metody podziału okaże się najlepsza. Podczas eksperymentów weryfikowano również hipotezy dotyczące mechanizmów działania poszczególnych komponentów metody. W szczególności, niezbędne było pokazanie skuteczności zastosowanego algorytmu genetycznego, który jest głównym składnikiem dekompozycji danych wejściowych na podtabele określone wzorcami.

7.9.1 Hipoteza statystyczna

Porównanie wyników algorytmów all i exact nie potwierdza słuszności hipotezy statystycznej, że większa liczba podziałów wpływa na poprawę jakości wnioskowania. Należy zauważyć, że wzorce wykorzystane w metodzie all, czyli wszystkie schematy wypełnienia, muszą być co najmniej tak szerokie (zawierać co najmniej tyle atrybutów), co wzorce użyte w metodzie exact. Nie jest zatem możliwe, aby w wzorce użyte w metodzie all ograniczały liczbę atrybutów uniemożliwiając tym samym wykrycie zależności pomiędzy atrybutami warunkowymi a decyzją. Wyniki jakie możemy zaobserwować, szczególnie dla tabel posiadających dużą liczbę schematów, jak np. hco, pokazują, że algorytm all cechuje nieco mniejsza dokładność klasyfikacji, niż algorytm exact. Istnieją co prawda zbiory danych, dla których to metoda all okazuje się być lepsza, niemniej jednak nie są to częste przypadki.

7.9.2 Algorytm genetyczny

Wyniki metod exact, ga50, ga20 i ga10 ilustrują efektywność zaprojektowanego algorytmu genetycznego. Nawet ograniczona do 10–40 osobników i 10 iteracji metoda ga10 umożliwia wnioskowanie tak skuteczne, jak sprawdzanie wszystkich wzorców[9]. Różnice w osiąganych wynikach różnią się z tabeli na tabelę, niemniej jednak nie są duże i nie rozstrzygają o przewadze żadnego z algorytmów. Dla pewnych tabel każda z tych czterech metod okazuje się być najlepsza. Wyniki te dobrze świadczą o jakości zastosowanego algorytmu genetycznego. W celu przekonania się o dużej skuteczności tego algorytmu możemy również porównać liczbę znalezionych wzorców, uwidocznioną na tabeli 7.4. Dla większości tabel uzyskano dokładnie taką samą liczbę wzorców pokrywających całą tablicę. Pewną zasługę w uzyskaniu tak dobrych wyników należy również przypisać niezbyt dużej wrażliwości liczby pokrywających tabelę wzorców na niewielką zmianę konkretnych wzorców i ich własności.

[9] Liczba wszystkich wzorców jest oczywiście równa 2^K, gdzie K to liczba atrybutów.

	J48	all	exact	ga50	ga20	ga10	ev1	ev2	ev4	ev8	ev
att_n	52.55	55.11	55.10	55.22	55.19	54.94	55.77	57.78	60.34	61.94	63.33
att_s	57.79	58.13	58.09	58.01	58.00	57.99	59.12	60.83	63.00	64.23	65.17
ban_n	62.14	65.56		65.26	65.20	65.82	68.51	70.43	72.69	74.91	76.30
ban_s	73.62	72.26		73.79	74.14	73.70	76.90	78.71	80.72	82.41	83.31
$cmc2_n$	45.72	47.23	44.66	44.96	45.03	44.92	47.28	48.61	50.09	51.33	51.41
$cmc2_s$	47.88	47.68	47.18	47.06	46.98	47.17	48.31	50.24	51.88	53.19	53.27
dna2	86.84	87.31		80.48	80.50	80.73	86.20	86.95	87.16	88.39	89.07
$hab2_n$	71.54	68.26	67.96	68.00	67.90	68.07	69.14	70.71	72.90	74.67	75.98
$hab2_s$	71.13	72.11	72.36	72.41	72.47	72.66	72.55	73.82	74.75	75.43	75.36
hco_n	81.68	78.73	79.61	79.85	79.90	79.50	81.96	83.60	85.02	85.99	86.00
hco_s	81.22	77.38	78.67	78.74	78.96	80.17	82.67	84.12	85.50	86.48	86.69
hep_n	80.12	76.18	76.40	76.33	76.19	75.88	79.53	81.48	83.70	85.29	86.59
hep_s	78.35	76.83	75.90	76.15	76.25	76.37	81.43	84.60	86.32	88.18	88.74
hin	70.47	66.65	69.98	69.84	70.04	69.96	70.16	70.49	70.98	71.10	70.53
hyp_n	95.82	96.44	96.71	96.70	96.68	96.72	96.76	96.79	96.80	96.81	97.09
hyp_s	99.05	98.78	97.94	97.96	97.93	97.96	98.98	99.00	98.99	99.00	99.21
$pid2_n$	60.81	62.06	61.97	61.96	61.94	61.98	62.19	63.84	66.24	67.11	68.29
$pid2_s$	73.50	72.90	73.38	73.20	73.43	73.26	73.47	74.16	75.38	76.70	77.20
$smo2_n$	60.75	57.63	56.17	56.08	56.14	56.14	57.92	65.48	68.47	68.95	69.66
$smo2_s$	62.64	57.83	61.30	61.17	61.11	61.21	66.16	66.80	68.00	69.02	69.89
tumor	38.89	36.19	36.48	36.42	36.57	36.28	40.17	42.20	43.05	43.89	43.30

Tablica 7.2: Wyniki eksperymentów. Liczba poprawnych odpowiedzi klasyfikatora w procentach.

Jednakże wyniki takie wyraźnie pokazują dużą efektywność zaprojektowanego algorytmu genetycznego.

Parametry tego algorytmu były strojone na podstawie badań eksperymentalnych i prezentowane tutaj wyniki zostały wykonane przy najlepszych, dobranych empirycznie, ustawieniach. Podlegające strojeniu parametry to wielkość populacji i liczba iteracji, które ostatecznie zostały ustalone na niezależne od wielkości badanych tablic, a także prawdopodobieństwa zastosowania operatorów genetycznych i wybór operatora selekcji.

7.9.3 Jakość predykcyjna wzorca

Włączenie jakości predykcyjnej wzorca do funkcji oceny było kluczowym punktem eksperymentów. Należy przypomnieć, że wszystkie metody ga10, ev1, ..., ev8 oraz ev jako optymalizatora używały tego samego algorytmu genetycznego. Jedyna różnica polegała na sposobie obliczania funkcji oceny wzorca. Zmieniający się wykładnik przy jakości predykcyjnej wzorca określa wpływ tej wartości na funkcję oceny. Zastosowanie metod ga10, ev1, ev2, ev4, ev8 i ev można interpretować jako użycie wykładników odpowiednio 0,1,2,4,8 i ∞, przy czym tę ostatnią wartość należy interpretować nieformalnie.

Podczas analizy wstępnych eksperymentów, gdzie porównywano znalezione wzorce z wszystkimi wzorcami występującymi w danych, zauważono pewien rozrzut ostatecznych wyników, pomimo zastosowania podobnej liczby wzorców o zbliżonych gabarytach. Wiąże się to z występowaniem w danych dużej liczby wzorców o podobnych szerokościach i wysokościach, które cechuje zdecydowanie odmienna jakość predykcyjna, czyli wpływ na skuteczność wygenerowanych hipotez. W oczywisty sposób nie wszystkie atrybuty i ich kombinacje w taki sam sposób nadają się do aproksymacji pojęć, zakodowanych w postaci atrybutu

	J48	all	exact	ga50	ga20	ga10	ev1	ev2	ev4	ev8	ev
att_n	52.55	+2.56	+2.55	+2.67	+2.64	+2.39	+3.22	+5.23	+7.79	+9.39	+10.78
att_s	57.79	+0.34	+0.30	+0.22	+0.21	+0.20	+1.33	+3.04	+5.21	+6.44	+7.38
ban_n	62.14	+3.42		+3.12	+3.06	+3.68	+6.37	+8.29	+10.55	+12.77	+14.16
ban_s	73.62	-1.36		+0.17	+0.52	+0.08	+3.28	+5.09	+7.10	+8.79	+9.69
$cmc2_n$	45.72	+1.51	-1.06	-0.76	-0.69	-0.80	+1.56	+2.89	+4.37	+5.61	+5.69
$cmc2_s$	47.88	-0.20	-0.70	-0.82	-0.90	-0.71	+0.43	+2.36	+4.00	+5.31	+5.39
dna2	86.84	+0.47		-6.36	-6.34	-6.11	-0.64	+0.11	+0.32	+1.55	+2.23
$hab2_n$	71.54	-3.28	-3.58	-3.54	-3.64	-3.47	-2.40	-0.83	+1.36	+3.13	+4.44
$hab2_s$	71.13	+0.98	+1.23	+1.28	+1.34	+1.53	+1.42	+2.69	+3.62	+4.30	+4.23
hco_n	81.68	-2.95	-2.07	-1.83	-1.78	-2.18	+0.28	+1.92	+3.34	+4.31	+4.32
hco_s	81.22	-3.84	-2.55	-2.48	-2.26	-1.05	+1.45	+2.90	+4.28	+5.26	+5.47
hep_n	80.12	-3.94	-3.72	-3.79	-3.93	-4.24	-0.59	+1.36	+3.58	+5.17	+6.41
hep_s	78.35	-1.52	-2.45	-2.20	-2.10	-1.98	+3.08	+6.25	+7.97	+9.83	+10.39
hin	70.47	-3.82	-0.49	-0.63	-0.43	-0.51	-0.31	+0.02	+0.51	+0.63	+0.06
hyp_n	95.82	+0.62	+0.89	+0.88	+0.86	+0.90	+0.94	+0.97	+0.98	+0.99	+1.27
hyp_s	99.05	-0.27	-1.11	-1.09	-1.12	-1.09	-0.07	-0.05	-0.06	-0.05	+0.16
$pid2_n$	60.81	+1.25	+1.16	+1.15	+1.13	+1.17	+1.38	+3.03	+5.43	+6.30	+7.48
$pid2_s$	73.50	-0.60	-0.12	-0.30	-0.07	-0.24	-0.03	+0.66	+1.88	+3.20	+3.70
$smo2_n$	60.75	-3.12	-4.58	-4.67	-4.61	-4.61	-2.83	+4.73	+7.72	+8.20	+8.91
$smo2_s$	62.64	-4.81	-1.34	-1.47	-1.53	-1.43	+3.52	+4.16	+5.36	+6.38	+7.25
tumor	38.89	-2.70	-2.41	-2.47	-2.32	-2.61	+1.28	+3.31	+4.16	+5.00	+4.41

Tablica 7.3: Wyniki eksperymentów. Różnica osiągniętych wyników w stosunku do metody J48.

decyzyjnego. Użycie jakości predykcyjnej przy wyliczaniu funkcji oceny wzorca umożliwia selekcję tych wzorców, które wpłyną na polepszenie wyników klasyfikacji. Przykładowo przy wykładniku 1 wzorzec o jakości predykcyjnej 0.55 może posiadać o 10% mniejsze gabaryty niż wzorzec o jakości 0.50, a i tak zostanie oceniony jako lepszy (porównaj także tabelę 7.1).

Zdecydowana poprawa wyników metody ev1 w porównaniu do ga10 wykazuje słuszność zastosowania takiej metodologii. Porównując liczbę znalezionych wzorców (patrz tabela 7.4) znajdujemy potwierdzenie empiryczne obserwacji o dużej liczbie podobnych gabarytami wzorców. W większości przypadków liczba znalezionych wzorców nie zwiększyła się znacznie, a czasami nawet zmalała. Zauważmy zatem, że zdecydowana poprawa jakości klasyfikacji uzyskana została przy praktycznie identycznej liczbie podtabel użytych do dekompozycji danych.

Porównując wyniki kolejnych metod, ev2, ev4, ev8 i ev, obserwujemy stopniowy wzrost jakości klasyfikacji. Dla niektórych tabel liczba wzorców, które posłużyły do dekompozycji, niewiele wzrasta, lub stabilizuje się na poziomie zbliżonym do uzyskanego w metodach exact i ga10. Istnieją również tabele, gdzie występuje drastyczny wzrost liczby podtabel, aż do wielkości porównywalnych z liczbą schematów. Nie istnieje jednak szczególny związek, pomiędzy szybkością wzrostu liczby wzorców, a uzyskaną poprawą (pogorszeniem) jakości wnioskowania.

Wyniki uzyskane przy zastosowaniu metody ev są najlepsze ze wszystkich, oraz jako jedyne pozostają lepsze od wyników metody J48 dla każdej tabeli. Chociaż pierwotnym zamysłem eksperymentów było porównanie jakości mechanizmów radzenia sobie z brakującymi wartościami, zbyt naiwnym stwierdzeniem było by, gdybyśmy przyjeli, że uzyskana

	all	exact	ga50	ga20	ga10	ev1	ev2	ev4	ev8	ev
att_n	17.15	3.90	3.93	3.90	3.94	4.00	4.10	4.15	4.09	5.35
att_s	17.19	3.90	3.93	3.93	3.96	3.97	4.00	4.03	3.88	4.97
ban_n	56.80		5.33	6.98	9.00	8.09	8.28	9.01	10.08	22.14
ban_s	56.83		5.27	6.85	8.92	8.18	8.49	9.26	10.62	23.10
$cmc2_n$	6.96	2.00	2.00	2.00	2.00	2.41	2.59	2.91	3.51	3.92
$cmc2_s$	6.94	2.00	2.00	2.00	2.00	2.15	2.47	3.25	4.11	5.25
dna2	7.80		1.00	1.01	1.06	2.54	2.63	2.61	3.55	7.08
$hab2_n$	5.00	3.83	3.67	3.65	3.69	3.20	3.01	2.78	2.50	1.84
$hab2_s$	5.00	3.82	3.65	3.66	3.65	3.67	3.52	3.24	3.08	2.33
hco_n	164.65	5.03	5.03	5.16	5.46	5.80	6.16	6.89	9.70	67.54
hco_s	164.58	5.00	5.02	5.20	5.51	6.08	6.38	7.39	9.97	65.26
hep_n	18.48	3.84	3.83	3.85	4.03	4.12	4.30	4.70	5.30	8.27
hep_s	18.47	3.83	3.81	3.86	4.02	4.15	4.39	4.81	5.50	8.77
hin	25.97	4.11	3.90	3.87	3.83	4.91	5.74	7.21	8.77	13.22
hyp_n	17.96	2.00	2.00	2.00	2.01	2.01	2.02	2.01	2.01	4.55
hyp_s	17.96	2.00	2.00	2.00	2.00	2.02	2.04	2.04	2.14	7.53
$pid2_n$	6.77	2.97	2.97	2.98	2.98	2.99	3.11	3.41	3.48	3.89
$pid2_s$	6.76	2.97	2.99	2.97	2.98	2.98	2.87	3.01	3.26	4.81
$smo2_n$	4.00	2.00	2.00	2.00	2.00	2.42	1.80	1.33	1.39	2.14
$smo2_s$	4.00	2.00	2.00	2.00	2.00	1.26	1.15	1.15	1.26	2.06
tumor	6.40	1.99	1.99	1.99	2.17	2.53	3.03	3.58	3.84	4.37

Tablica 7.4: Średnia liczba użytych wzorców. Wartość ta odpowiada liczebności lokalnych podmodeli użytych w metodzie podziału.

poprawa jest tylko i wyłącznie zasługą lepszego potraktowania brakujących wartości. Metoda podziału oferuje dużo większe możliwości analizy danych, poprzez wielokrotne zastosowanie klasyfikatora. Zastosowanie algorytmu ev oznacza nie tylko inteligentną filtrację brakujących wartości, ale również dobór cech znaczących, czyli atrybutów istotnych do aproksymacji pojęć. Poprawa jakości klasyfikacji jest rezultatem wielu różnych czynników, podobnie jak ma to miejsce w innych metodach opartych na wielokrotnej klasyfikacji, takich jak np. Bagging i Boosting (patrz np. [43]). Niemniej jednak, jako całość metoda ta umożliwia radzenie sobie z brakującymi wartościami i to z końcową skutecznością zdecydowanie lepszą, od jednej z najlepszych metod potrafiących analizować dane z niekompletnym opisem obiektów, jaką jest C4.5.

Rozdział 8

Zakończenie

Zaprezentowana w niniejszej pracy metoda podziału jest skutecznym i uniwersalnym rozwiązaniem umożliwiającym wnioskowanie w oparciu o dane z niekompletnym opisem obiektów. Źródłem jej wysokiej sprawności jest zarówno uniemożliwienie systemom decyzyjnym wnioskowania w oparciu o brak informacji, jak i zastosowanie wielokrotnej, etapowej klasyfikacji, która pozwala na konstrukcję bardziej złożonych hipotez dotyczących badanego pojęcia. Ma ona jednak pewną przewagę nad innymi metodami uczenia się pojęć w oparciu o przykłady stosującymi złożony model hipotez.

Wyniki teorii maszynowego uczenia się pokazują, że gdy podczas procesu uczenia się przeszukujemy bardziej skomplikowaną przestrzeń hipotez w celu odnalezienia tej pasującej do badanego pojęcia, wzrasta znacznie liczba niezbędnych przykładów do prawidłowego wyuczenia się pojęcia. To zjawisko opisuje tzw. wymiar Vapnika-Chervonenkisa (patrz [8, 57]). W metodzie podziału unika się tego problemu stosując dwuetapową konstrukcję opisu pojęcia na zbiorze wszystkich przykładów.

Ta własność w połączeniu ze skuteczną eliminacją brakujących wartości z procesu wnioskowania pozwala na uzyskanie dobrej skuteczności klasyfikacji. Jak pokazują wyniki eksperymentalne metoda podziału przewyższa swoją skutecznością metodę C4.5 uznawaną za najlepszą metodę wnioskowania w oparciu o dane z niekompletnym opisem obiektów.

Metoda podziału została zaprojektowana pod kątem jej zastosowania w systemach decyzyjnych opartych na teorii zbiorów przybliżonych. Planowana jest implementacja metody podziału w ramach biblioteki RSES-lib wykonanej w Zakładzie Logiki Matematycznej Uniwersytetu Warszawskiego pod opieką naukową prof. dra hab. Andrzeja Skowrona przez zespół ludzi pod kierownictwem dra Jana Bazana.

Dalszym kierunkiem do badań nad brakującymi wartościami atrybutów powinno być skonstruowanie algorytmu umożliwiającego odkrywanie wiedzy dotyczącej brakujących wartości bezpośrednio z danych. Wiedza taka powinna umożliwiać algorytmiczne wyznaczenie optymalnej relacji nierozróżnialności dla rozpatrywanych danych. Od takiej relacji oczekuje się, że powinna maksymalizować jakość wnioskowania przez generowanie aproksymacji pojęć o jak najmniejszym brzegu, przy jednoczesnym zachowaniu poprawności wnioskowania indukcyjnego i jego zdolności do generalizacji.

Bibliografia

[1] *Encyklopedia popularna PWN*. Państwowe Wydawnictwo Naukowe, Warszawa, wydanie piąte, 1982.

[2] J. G. Bazan. *Metody wnioskowań aproksymacyjnych dla syntezy algorytmów decyzyjnych*. Praca doktorska, Uniwersytet Warszawski, Wydział Matematyki, Informatyki i Mechaniki, 1998.

[3] G. Bińczak. *Charakteryzacja klas algebr częściowych definiowanych przez słabe równości*. Praca doktorska, Uniwersytet Warszawski, Wydział Matematyki, Informatyki i Mechaniki, Warszawa, 2000.

[4] A. Birkendorf, N. Klasner, C. Kuhlman, and H. U. Simon. Structural results about exact learning with unspecified attribute values. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 144–153, 1998.

[5] C. L. Blake and C. J. Merz. *UCI Repository of machine learning databases*. http://www.ics.uci.edu/˜mlearn/MLRepository.html, Univerity of California, Departament of Information and Computer Science, Irvine, CA, 1998.

[6] N. H. Bshouty and D. K. Wilson. On learning in the presence of unspecified attribute values. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory, COLT'99*, pages 81–87. ACM, 1999.

[7] P. Burmeister. *A model — theoretic oriented approach to partial algebras*. Akademie-Verlag, Berlin, 1986.

[8] P. Cichosz. *Systemy uczące się*. WNT, 2000.

[9] J. Cytowski. *Algorytmy genetyczne. Podstawy i zastosowania*. Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1996.

[10] T. Dietterich, M. Kearns, and Y. Mansour. Applying the week learning framework to understand and improve C4.5. In L. Saitta, editor, *Proceedings of the Thirteenth International Conference on Machine Learning, ICML'96*, pages 96–104. Morgan Kaufmann, 1996.

[11] D. Driankov, H. Hellendoorn, and M. Reinfrank. *Wprowadzenie do sterowania rozmytego*. WNT, Warszawa, 1996.

[12] E. Frank, L. Trigg, and M. Hall. *Weka 3.1.9, Waikato Environment for Knowledge Analysis*. http://www.cs.waikato.ac.nz/ml/weka, The University of Waikato, Hamilton, New Zealand, 2000.

[13] J. H. Friedman, R. Kohavi, and Y. Yun. Lazy decision trees. In Shrobe and Senator [47], pages 717–724.

[14] Y. Fujikawa and T. Ho. Scalable algorithms for dealing with missing values. 2001.

[15] Z. Ghahramani and M. I. Jordan. Supervised learning from incomplete data via an EM approach. In J. D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 120–127. Morgan Kaufmann, 1994.

[16] D. E. Goldberg. *Algorytmy genetyczne i ich zastosowania*. WNT, Warszawa, 1995.

[17] S. A. Goldman, S. Kwek, and S. D. Scott. Learning from examples with unspecified attribute values. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 231–242, 1997.

[18] G. Grätzer. *Universal Algebra*. Springer-Verlag, New York, 1979.

[19] S. Greco, B. Matarazzo, and R. Słowiński. Fuzzy similarity relation as a basis for rough approximations. In L. Polkowski and A. Skowron, editors, *Rough sets and current trends in computing, Proceedings of the RSCTC'98*, pages 283–289. Springer-Verlag, 1998.

[20] S. Greco, B. Matarazzo, and R. Słowiński. Handling missing values in rough set analysis of multi-attribute and multi-criteria decision problems. In Zhong et al. [63], pages 146–157.

[21] S. Greco, B. Matarazzo, and R. Słowiński. Rough sets processing of vague information using fuzzy similarity relations. In C. S. Caldue and G. Paun, editors, *Finite vs. infinite: contribution to an eternal dilemma*, pages 149–173, Berlin, 2000. Springer-Verlag.

[22] S. Greco, B. Matarazzo, R. Słowiński, and S. Zanakis. Rough set analysis of information tables with missing values. In *Proceedings of 5th International Conference Decision Sciences Institute, July 4–7, Athens-Greece*, volume 2, pages 1359–1362, 1999.

[23] J. W. Grzymała-Busse, W. J. Grzymała-Busse, and L. K. Goodwin. A closest fit approach to missing attribute values in preterm birth data. In Zhong et al. [63], pages 405–413.

[24] J. W. Grzymała-Busse, W. J. Grzymała-Busse, and L. K. Goodwin. An approach to missing attribute values based on closest fit in preterm birth data. 2000.

[25] J. W. Grzymała-Busse and M. Hu. A comparison of several approaches to missing attribute values in data mining. In Ziarko and Yao [65], pages 180–187.

[26] J. Komorowski, Z. Pawlak, L. Polkowski, and A. Skowron. Rough sets: A tutorial. In S. K. Pal and A. Skowron, editors, *Rough Fuzzy Hybridization. A New Trend in Decision Making*, pages 3–98. Springer-Verlag, 1998.

[27] K. Krawiec, R. Słowiński, and D. Vanderpooten. Learning decision rules from similarity based rough approximation. In N. Zhong, A. Skowron, and S. Ohsuga, editors, *Rough sets in knowledge dicovery, Applications, Case studies and software systems*, volume 2, pages 37–54, Heidelberg, 1998. Physica-Verlag.

[28] R. Kruse, J. Gebhardt, and F. Klawonn. *Foundation of Fuzzy Systems*. John Wiley & Sons, 1994.

[29] M. Kryszkiewicz. Properties of incomplete information systems in the framework of rough sets. In L. Polkowski and A. Skowron, editors, *Rough Sets in Data Mining and Knowledge Discovery*, pages 422–450. Physica-Verlag, 1998.

[30] W. Z. Liu, A. P. White, S. G. Thompson, and M. A. Bramer. Techniques for dealing with missing values in classification. In X. Liu, P. Cohen, and M. R. Berthold, editors, *Advances in Intelligent Data Analysis*, pages 527–536. Springer-Verlag, 1997.

[31] Z. Michalewicz. *Algorytmy genetyczne + struktury danych = programy ewolucyjne*. WNT, 1999.

[32] H. S. Nguyen. From optimal hyperplanes to optimal decision trees. *Fundamenta Informaticae*, 34:145–174, 1998.

[33] H. S. Nguyen and S. H. Nguyen. Rough sets and association rule generation. *Fundamenta Informaticae*, 40:383–405, 1999.

[34] S. H. Nguyen. *Regularity Analysis and its Application in Data Mining*. Praca doktorska, Warsaw University, Faculty of Mathematics, Computer Science and Mechanics, 1999.

[35] S. H. Nguyen, A. Skowron, and P. Synak. Discovery of data patterns with applications to decomposition and classification problems. In L. Polkowski and A. Skowron, editors, *Rough Sets in Knowledge Discovery*, volume 2, pages 55–97, Heidelberg, 1998. Physica-Verlag.

[36] O. Ortega Lobo and M. Numao. Ordered estimation of missing values. In Zhong and Zhou [64], pages 499–503.

[37] Z. Pawlak. Rough sets. *International Journal of Computer and Information Sciences*, 11:341–356, 1982.

[38] Z. Pawlak. *Rough sets: Theoretical aspects of reasoning about data*. Kluwer, Dordrecht, 1991.

[39] L. Polkowski, A. Skowron, and J. M. Żytkow. Tolerance based rough sets. In T. Y. Lin and A. M. Wildberger, editors, *Soft Computing*, pages 55–58. San Diego Simulation Councils Inc., 1995.

[40] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[41] J. R. Quinlan. Unknown attribute values in induction. In A. M. Segre, editor, *Proceedings of the Sixth International Machine Learning Workshop*, pages 31–37. Morgan Kaufmann, 1989.

[42] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, San Mateo, 1993.

[43] J. R. Quinlan. Bagging, Boosting, and C4.5. In Shrobe and Senator [47], pages 725–730.

[44] J. R. Quinlan and R. L. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80:227–248, 1989.

[45] D. B. Rubin. *Multiple Imputation for Nonresponse in Surveys*. John Wiley & Sons, New York, 1987.

[46] L. Rudak. *Słabe rozmaitości algebr częściowych*. Praca doktorska, Uniwersytet Warszawski, Wydział Matematyki, Informatyki i Mechaniki, Warszawa, 1986.

[47] H. Shrobe and T. Senator, editors. *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference, AAAI96, IAAI96*, volume 1. AAAI Press / The MIT Press, 1996.

[48] A. Skowron. Boolean reasoning for decision rules generation. In J. Komorowski and Z. Raś, editors, *Proceedings of the 7th International Symposium ISMIS'93, Trondheim, Norway*, pages 295–305. Springer-Verlag, 1993.

[49] A. Skowron. Extracting laws from decision tables. *Computational Intelligence*, 11 (2):371–388, 1995.

[50] A. Skowron and C. Rauszer. *The Discernibility Matrices and Functions in Information Systems*, pages 331–362. Kluwer, Dordrecht, 1992.

[51] R. Słowiński and D. Vanderpooten. Similarity relation as a basis for rough approximations. Research Report 53/95, Institute of Computer Science, Warsaw University of Technology, 1995.

[52] R. Słowiński and D. Vanderpooten. A generalized definition of rough approximations based on similarity. *IEEE Transactions on Data and Knowledge Engineering*, 12:331–336, 2000.

[53] J. Stefanowski. *Algorytmy indukcji reguł decyzyjnych w odkrywaniu wiedzy*. Rozprawa Habilitacyjna, Politechnika Poznańska, 2001.

[54] J. Stefanowski and A. Tsoukiàs. On the extension of rough sets under incomplete information. In Zhong et al. [63], pages 73–81.

[55] J. Stefanowski and A. Tsoukiàs. Decision rules and valued tolerance. In Ziarko and Yao [65], pages 180–187.

[56] J. Stefanowski and A. Tsoukiàs. Incomplete information tables and rough classification. *International Journal of Computational Intelligence*, 2001.

[57] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[58] G. I. Webb. The problem of missing values in decision tree grafting. In *Proceedings of the Tenth Australian Joint Conference on Artificial Intelligence*, pages 273–283, 1998.

[59] S. M. Weiss and N. Indurkhya. Decision-rule solutions for data mining with missing values. IBM Research Report RC-21783, IBM T. J. Watson Research Center, 2000.

[60] S. M. Weiss and N. Indurkhya. Lightweight rule induction. In *Proceedings of the International Conference on Machine Learning ICML'2000*, 2000.

[61] I. H. Witten and E. Frank. *Data Mining: Practical Mashine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 2000.

[62] Z. Zheng and B. T. Low. Classifying unseen cases with many missing values. In Zhong and Zhou [64], pages 370–374.

[63] N. Zhong, A. Skowron, and S. Ohsuga, editors. *New Directions in Rough Sets, Data Mining and Granular-Soft Computing, Proceedings of 7th International Workshop RSFD-GrC'99*. Springer-Verlag, 1999.

[64] N. Zhong and L. Zhou, editors. *Methodologies for Knowledge Discovery and Data Mining, Third Pacific-Asia Conference, PAKDD-99*. Springer-Verlag, 1999.

[65] W. Ziarko and Y. Y. Yao, editors. *Proceedings of 2nd International Conference on Rough Sets and Current Trends in Computing, RSCTC-2000*, 2000.