SKRYPT DO PRZEDMIOTU

SZTUCZNA INTELIGENCJA W MEDYCYNIE

autorzy:

prof. dr hab. inż. Bożena Kostek,

dr inż. Piotr Szczuko

Gdańsk, 2015

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt *„Przygotowanie i realizacja kierunku inżynieria biomedyczna – studia międzywydziałowe”*
współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Spis treści

1 Wprowadzenie

Celem niniejszego skryptu jest zapoznanie studentów kierunku Inżynieria Biomedyczna z podstawami metod i algorytmów sztucznej inteligencji oraz wybranymi przykładami zastosowań w obszarze medycyny.

W nowoczesnym systemie opieki zdrowotnej można zauważyć szybko rosnącą złożoność procesów decyzyjnych i ich koszt. Dzieje się to na skutek dużej ilości strumieni informacji (nowo powstająca tworzona aparatura, procedury i terapie, różnorodność leków i generyków, itp.), które przekładają się na nowe opcje leczenia, ale jednocześnie mogą utrudniać wybór optymalnych decyzji dotyczących leczenia w przypadkach konkretnego pacjenta. W takiej sytuacji rozwiązaniem jest zastosowanie systemów wspomagających, wykorzystujących systemy decyzyjne.

W pierwszej kolejności przywołano zagadnienia związane ze sztucznymi sieciami neuronowymi, zwłaszcza, że mają swoje odniesienie do modelu komórki nerwowej, czyli budowy neuronu. Klasyfikacji sieci neuronowych dokonuje się według czterech podstawowych kryteriów: metod trenowania, kierunków propagacji sygnałów w sieci, typów funkcji przejścia, rodzajów danych wprowadzanych na wejścia, te właśnie zagadnienia zostaną omówione w rozdziale 2. Zwrócono również uwagę na własności generalizacyjne sieci neuronowych, co definicyjnie oznacza zdolność do wyznaczania poprawnych wartości wyjściowych po wprowadzeniu na wejście sieci neuronowej wektorów danych wejściowych, które nie były wykorzystywane w trakcie treningu (ale pochodzą z tego samego źródła co dane do treningu), a w praktyce - zdolności od uogólniania wyników. Ten aspekt jest szczególnie istotny w warunkach baz medycznych, gdy pojawiają się nowe przypadki dotyczące danej jednostki chorobowej.

Bazy medyczne przechowują często informacje opisowe lub nieprecyzyjne, zaś przetwarzanie takich danych mogłoby prowadzić do wielu niejednoznaczności, dlatego w rozdziale 3 przedstawiono zagadnienia dotyczące wnioskowania rozmytego. Podano najważniejsze definicje w odniesieniu do zbioru klasycznego i zbioru rozmytego, rozmyty opis atrybutu, a także przedstawiono wszystkie elementy proces wnioskowania rozmytego. Logika rozmyta wykorzystuje wiedzę eksperta - ekspert na podstawie zdobytego wcześniej doświadczenia może określić sposób postępowania dla poszczególnych przypadków, które mogą się zdarzyć w trakcie procesu, ustalać granice funkcji przynależności dla każdego przypadku, przypisywać etykiety w procesie rozmywania atrybutów. Ten ostatni aspekt ma ogromne znaczenie w przypadku baz medycznych, gdzie atrybuty nie poddają się łatwo kwantyfikacji, z kolei w przypadku danych ciągłych szczegółowych może zaistnieć potrzeba rozmycia poszczególnych atrybutów. Zadaniem eksperta może też być zarówno konstrukcja reguł wnioskowania, jak również późniejsza weryfikacja reguł uzyskanych w procesie uczenia.

W kolejnym rozdziale przywołano algorytmy budowania i klasyfikacji za pomocą drzew decyzyjnych. Szczególnie istotna w przypadku klasyfikacji danych ciągłych i potrzeby dyskretyzacji jest metoda zbiorów przybliżonych. W tym miejscu należałoby zwrócić uwagę na kilka ważnych aspektów tej metody, jak brak potrzeby formułowania założeń wstępnych, ale również umożliwienie wykrywania związków i relacji występujących w zbiorze danych. Podobnie, jak w przypadku wnioskowania rozmytego wiedza ekspercka może być wykorzystana do tworzenia i weryfikacji reguł.

Jak wspomniano wcześniej dane medyczne są często niespójne, nieprecyzyjne czy wręcz zawierają dane sprzeczne. W takim przypadku do przetwarzania danych tego typu szczególnie dobrze nadaje się metoda oparta na zbiorach przybliżonych. Teoria zbiorów przybliżonych, wprowadzana przez Pawlaka a latach 80. poprzedniego stulecia, jest często stosowana w kontekście redukcji danych nadmiarowych, selekcji ważnych atrybutów odkrywania wzorców z danych, jak i odkrywania zależności w bazach danych, co jest szczególnie istotne w przypadku baz medycznych. W rozdziale 5 przytoczono najważniejsze pojęcia związane z teorią zbiorów przybliżonych, definicje, jak również przykłady wnioskowania i przetwarzania oraz szeroko stosowany system decyzyjny RSES, wykorzystujący zbiory przybliżone.

Rozdział 6 odwołuje się, podobnie, jak w przypadku sztucznych sieci neuronowych, do bezpośredniej analogii systemów biologicznych, a w szczególności do zjawiska ewolucji populacji biologicznej, na którym oparte są założenia algorytmów genetycznych. W rozdziale tym podane zostały definicje, zasada działania algorytmu genetycznego, metody selekcji, pojęcia krzyżowania oraz mutacji.

Końcowe dwa rozdziały zawierają przykłady możliwości praktycznego wykorzystania metod sztucznej inteligencji w przetwarzaniu sygnałów i danych medycznych oraz podsumowanie niniejszego skryptu.

W skrypcie przyjęto konwencję braku szczegółowych cytowań do poszczególnych pozycji Literatury, ale zarówno rozdziały przeglądowe, jak i przykłady zastosowań zawierają podrozdział Literatura, w którym znajdują się wykorzystywane źródła. Opracowany materiał powstał przy szerokim wykorzystaniu wymienionych źródeł, z których pochodzą m.in. rysunki i fragmenty opisów. W skrypcie zostały również zawarte materiały zamieszczone w prezentacjach do wykładu „Sztuczna inteligencja w medycynie” dla międzywydziałowego kierunku Inżynieria Biomedyczna Politechniki Gdańskiej. Zalecaną pozycją dla studentów specjalności Inżynieria Biomedyczna mogą też być następujące pozycje literatury: Informatyka medyczna R. Tadeusiewicza, Lublin 2010 r. i wyd. Tadeusiewicz R., Korbicz J., Rutkowski L., Duch W. (Edytorzy), Sieci neuronowe w inżynierii biomedycznej, Wyd. Exit, Warszawa 2013, pp. 775.

2 Sztuczne sieci neuronowe

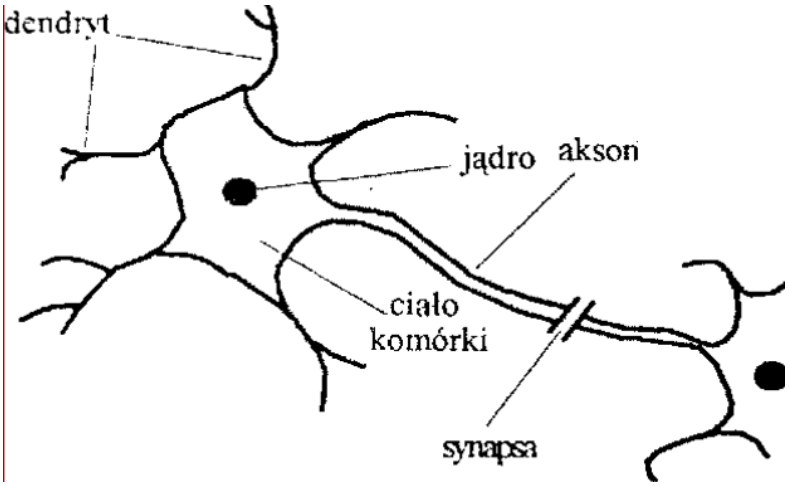
W niniejszym rozdziale zostaną pokrótce przedstawione wybrane zagadnienia sztucznych sieci neuronowych.

Mózg to bardzo duża (w literaturze przyjmuje się, że ok. 10 miliardów) liczba elementarnych komórek nerwowych, neuronów połączonych w formie skomplikowanej sieci. Średnio na jeden neuron przypada kilka tysięcy połączeń, ale dla poszczególnych komórek liczby połączeń mogą być różne. Zasada działania każdego neuronu jest identyczna, a dopiero ze złożoności struktur tworzonych przez miliony komórek wynika specjalizacja mózgu, zdolność nauki, zapamiętywania, rozwiązywania problemów.

Zakłada się, że poprzez częściowe naśladownictwo ludzkiego mózgu model komputerowy może uzyskać pewne zdolności nauki, generalizacji i klasyfikacji danych.

Początek teorii sztucznych sieci neuronowych wyznacza praca McCullocha i Pittsa z 1943 r. [5], która zawiera pierwszy matematyczny opis komórki nerwowej i powiązanie tego opisu z problemem przetwarzania danych.

Biologiczny neuron kojarzyć można z jednostką obliczeniową posiadającą wejścia, moduł przetwarzania i wyjścia, będące wejściami do innych neuronów (rys. 2.1).



Rys. 2.1. Budowa neuronu biologicznego

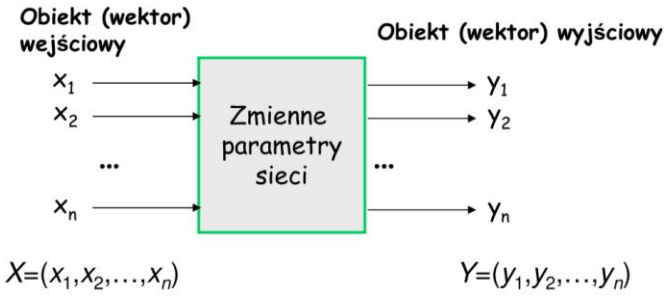
2.1 Model matematyczny sztucznego neuronu

W modelu komputerowym symuluje się wyłącznie podstawowe funkcje biologicznego systemu nerwowego.

2.1.1 Sieć neuronowa jako „czarna skrzynka”

Sieci neuronowe, podobnie jak ludzki mózg, w wyniku uczenia automatycznie dostosowują swoją strukturę dla danego zagadnienia na podstawie danych uczących. Dane uczące muszą mieć postać obiektów (rekordów, wierszy w tabeli, itp.), które opisane są atrybutami wyrażonymi liczbami rzeczywistymi. Dla każdego obiektu uczącego należy także dostarczyć do sieci informacji o oczekiwanej od sieci odpowiedzi (metoda treningu z nauczycielem. Więcej szczegółów dotyczących tej metody oraz innych podejściach zostanie podanych w części dotyczącej nauki sieci neuronowych – rozdział 2.4).

Wytrenowana sieć neuronowa jest „czarną skrzynką” rozwiązującą wyuczony problem. Sieci w trakcie treningu umieją odkrywać i odwzorować w swojej strukturze różne złożone zależności pomiędzy obiektami (danymi, sygnałami, bodźcami) wejściowymi **x** i wyjściowymi **y** (rys. 2.2).



Rys. 2.2. Sieć neuronowa jako „czarna skrzynka”, przetwarzająca wejście **x** na wyjście **y**

Do nauki sieci neuronowej potrzebne są wektory (obiekty) wejściowe x_i wraz z właściwymi im wektorami (odpowiedziami, klasami) wyjściowymi y_j. Długości wektorów **x** i **y** mogą się różnić, np. sieć klasyfikująca obiekty x_i do dwóch klas może mieć: 1) jedno wyjście, którego stan wysoki oznaczać będzie klasę pierwszą, a niski drugą; 2) lub dwa wyjścia „konkurujące” ze sobą, a wówczas wynikowa klasa to numer wyjścia, na którym obserwowana jest większa wartość y_i.

2.2 Model neuronu

Neuron biologiczny jest komórką o wejściach, elemencie przetwarzającym i wyjściach. Biologiczny neuron zostaje zamieniony na model matematyczny. Z wielu sztucznych neuronów budowane są skomplikowane struktury decyzyjne naśladujące pewne funkcjonalności ludzkiego mózgu. Model taki (rys. 2.3) ma następującą postać:

$$\mathbf{y}=f(\mathbf{w}^T \mathbf{x}) \tag{2.1}$$

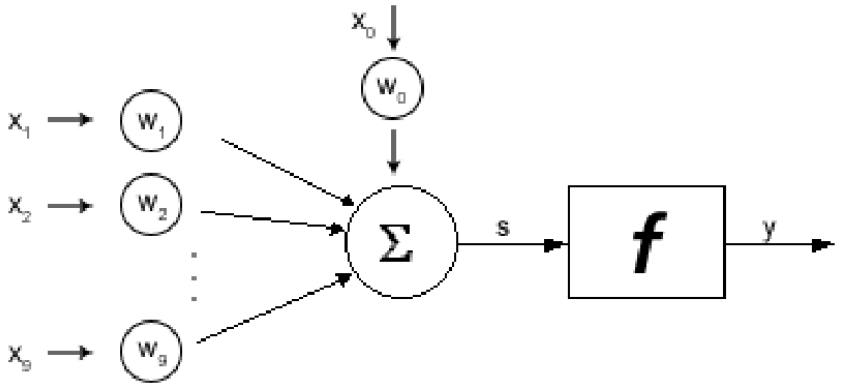
gdzie:

y – skalar lub wektor, wartość wyjściowa neuronu,

x – wektor wartości x_i sygnału wejściowego, związanych np. z atrybutami obiektu lub kolejne próbki sygnału (zależnie od zastosowania sieci)

w – wektor wag w_i połączeń wejściowych, i-te wejście wymnażane jest przez i-tą wagę, a następnie sumowane. W ten sposób wyliczana jest suma ważona z wszystkich wartości wejściowych.

f – tzw. **funkcja aktywacji neuronu**, przyjmująca jako argument wynik sumy ważonej.



Rys. 2.3. Schemat działania sztucznego neuronu

Wartości x_i dla i=1, …, n to elementy wektora sygnału wejściowego, natomiast dodatkowe połączenie to wartość progowa x_0, zwykle równa -1, wymnażana przez wagę w_0. Jej zadaniem jest zmniejszanie wartości sumy ważonej tak, aby dobrze odpowiadała dziedzinie zmienności argumentu funkcji f. Suma ważona wyliczana jest zgodnie z (2.1) z iloczynu wektorów **w** transponowane i **x** (2.2):

$$\mathbf{w}=[w_0, w_1, w_2, ..., w_n],\ \mathbf{x}=[-1, x_1, x_2, ..., x_n] \qquad (2.2)$$

Suma ważona $\mathbf{w}^T\ \mathbf{x}$ jest wartością skalarną, oznaczana jest jako *net*, a wyjście neuronu krótko jako f(net).

Wyjście **y** jednego neuronu podawane jest na wejście drugiego neuronu. Sposób połączeń między neuronami oraz wykorzystywane w nich funkcje aktywacji dobierane są w zależności od zagadnienia.

Wyuczoną sieć można badać, określać skuteczność jej działania, jej zdolności generalizacyjne, porównywać między sobą różne sieci. Jednakże parametry wewnętrzne sieci, wagi **w** i funkcje aktywacji f nie dają się wprost zinterpretować. Parametry te można wizualizować, odczytywać, jednak nie mają one sensu fizycznego, ani prostego powiązania z danymi (porównać można tę cechę z drzewami decyzyjnymi i zbiorami przybliżonymi, gdzie wartość atrybutu w regule wpływa na podejmowaną decyzję, co znajduje odzwierciedlenie w doświadczeniu eksperta i jest łatwe w interpretacji – sieć takiej zalety nie posiada).

2.3 Rodzaje sieci neuronowych

Sieci neuronowe różnią się wykorzystywanymi funkcjami aktywacji, sposobami połączeń neuronów między sobą oraz metodą nauki.

2.3.1 Funkcje aktywacji

Aktywność neuronu to wartość pośrednio związana z jego wartością wyjściową. Zależność ta przedstawia się w postaci funkcji f nazywanej funkcją aktywacji neuronu. Funkcje te można podzielić na trzy zasadnicze grupy (rys. 2.4):

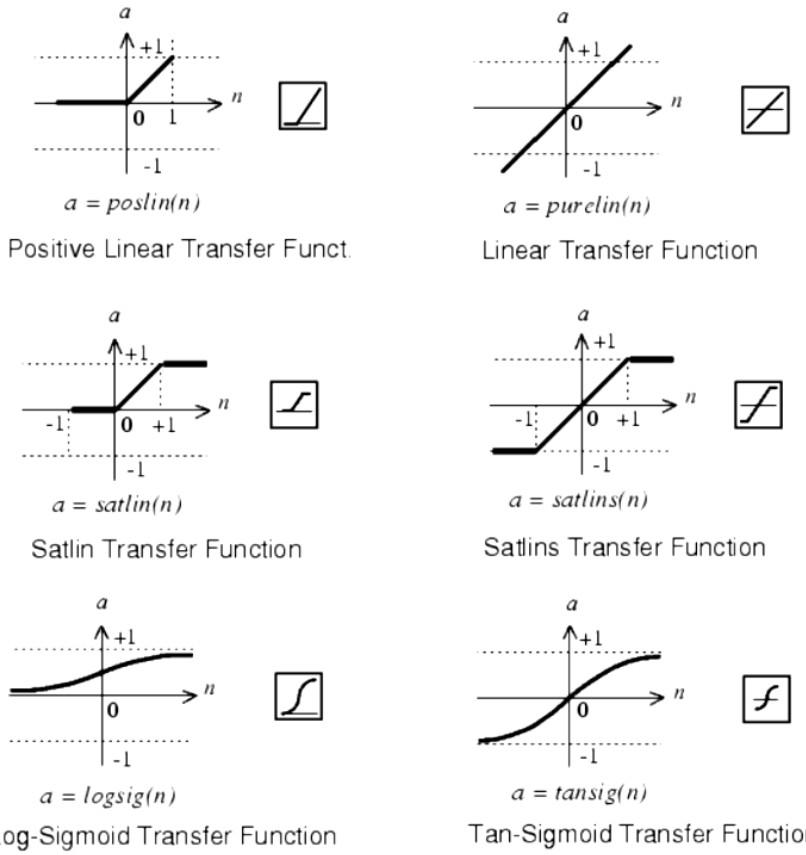
- **progowe** – wartość na wyjściu przybiera jeden z dwóch poziomów (0/1 lub -1/1),
- **liniowe** – wartości wyjściowe zmieniają się proporcjonalnie do wartości wejściowych,
- **nieliniowe** – wartości funkcji zmieniają się nieliniowo.

Wybrane popularne funkcje aktywacji przedstawione są poniżej (tab. 2.1).

Tab. 2.1. Często stosowane funkcje aktywacji. Prosta postać pochodnej wymagana jest do liczenia gradientów w procesie nauki

funkcja	Wzór funkcji	Wzór pochodnej
Sigmoida	$f(x)=\frac{1}{1+e^{-(beta*x)}}$	$f'(x)=beta*(1-f(x))*f(x)$
Tangens hiperboliczny	$f(x)=\tanh(beta*x)$	$f'(x)=beta(1-f^2(x))$
Sinusoida	$f(x)=\sin(beta*x)$	$f'(x)=beta\sqrt{1-f^2(x)}$
Cosinusoida	$f(x)=\cos(beta*x)$	$f'(x)=-beta\sqrt{1-f^2(x)}$
$\frac{x}{(1+\lvert x\rvert)}$ (bez nazwy)	$f(x)=\frac{beta*x}{(1+\lvert beta*x\rvert)}$	$f'(x)=\frac{beta}{1+\lvert beta*x\rvert}-\frac{\lvert beta*x\rvert}{(1+\lvert beta*x\rvert)^2}$

Kilka innych typów dostępnych jest w oprogramowaniu naukowym MATLAB (rys. 4.4).



Rys. 2.4. Przegląd funkcji aktywacji dostępnych w oprogramowaniu MATLAB. Pod wykresem *a=nazwa(n)* to sposób wywołania funkcji

Zadaniem funkcji aktywacji jest zamiana wejściowej sumy ważonej $n=\mathbf{w}^T\mathbf{x}$, która przyjąć może dowolne wartości na osi poziomej, na wartość wyjściową z zakresu akceptowanego na wejściach innych neuronów. Zaobserwować można na osiach pionowych (rys. 2.4), że zwykle jest to ograniczenie do zakresów (-1, 1) lub (0, 1).

Dla sprawnego działania sieci i skutecznego jej treningu wymagane jest, aby funkcje aktywacji cechowały:

- ciągłe przejście pomiędzy jej wartością maksymalną a minimalną (np. (0; 1)),
- łatwa do obliczenia i ciągła pochodna (tab. 2.1. przytacza wzory na obliczenie pochodnych kilku funkcji).
- możliwość wprowadzenia do argumentu/parametru *beta* do ustalania kształtu krzywej.

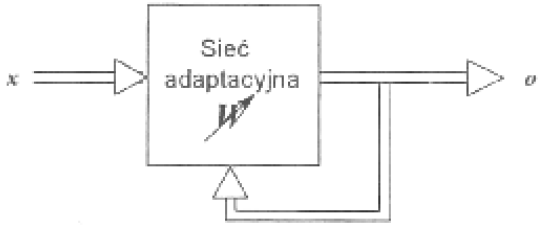
Dla funkcji ciągłej istnieje pochodna w każdym jej punkcie. Pochodna określa nachylenie funkcji, co jest wykorzystywane w treningu sieci metodą gradientową. Pochodna odpowie na pytanie: „w którą stronę zmienić argument, aby uzyskać wzrost/spadek wartości”. Dokładniej: jak zmienić wagi **w**, aby dla konkretnego **x** uzyskać wynik $\mathbf{y}=f(\mathbf{w}^T\mathbf{x})$ jak najbardziej zbliżony do oczekiwanego **y**.

2.3.2 Metoda treningu

To, jak sieć neuronowa realizuje swoje zadanie zależy od wag i funkcji aktywacji każdego neuronu. Ustalanie odpowiednich wartości wag dla wszystkich neuronów, a tym samym ustalenie żądanych odpowiedzi sieci neuronowej na konkretne pobudzenia odbywa się w procesie treningu sieci. Metody treningu można podzielić na dwie zasadnicze grupy: z nauczycielem/nadzorem i bez nauczyciela/nadzoru.

Trening bez nadzoru

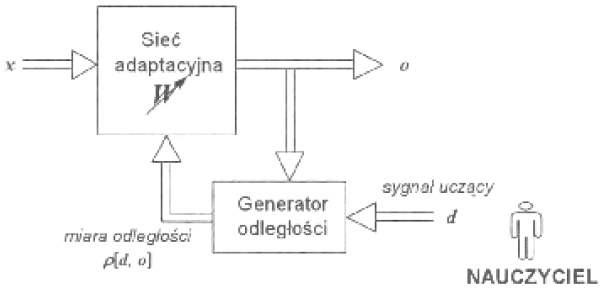
Sieci wykorzystujące trening bez nauczyciela określane są często mianem samoorganizujących (rys. 2.5). Ich zadaniem jest poszukiwanie w nieopisanym zbiorze obiektów/danych pewnych zależności nieokreślanych z góry przez twórcę sieci. W wyniku nauki sieć nauczy się dawać zbliżone/identyczne odpowiedzi y_1 i y_2 na podobne do siebie obiekty x_1 i x_2. Oznacza to, że sieć dokonuje klasteryzacji, łączenia w grupy obiektów. Taka sieć wykorzystywana jest np. do wyszukiwania obrazów podobnych.



Rys. 2.5. Schemat nauki sieci bez nadzoru

Trening z nadzorem

Prostsza w użyciu, a tym samym bardziej popularna jest metoda treningu z nadzorem (rys. 2.6).



Rys. 2.6. Schemat nauki sieci z nadzorem

W takim przypadku do treningu sieci neuronowej konieczne jest przygotowanie odpowiednio dużego zbioru danych wraz z oczekiwanymi odpowiedziami sieci (decyzjami). Istnieje wiele algorytmów treningu i minimalizowania błędu, czyli odległości/różnicy między odpowiedzią a oczekiwaniem. W wyniku długiego treningu nie zawsze odpowiedź sieci jest w 100% poprawna. Co więcej, z powodu zagrożenia **przetrenowaniem**, nie oczekuje się od sieci idealnego działania (więcej na ten temat w rozdziale 2.7, poświęconym własnościom generalizacyjnym).

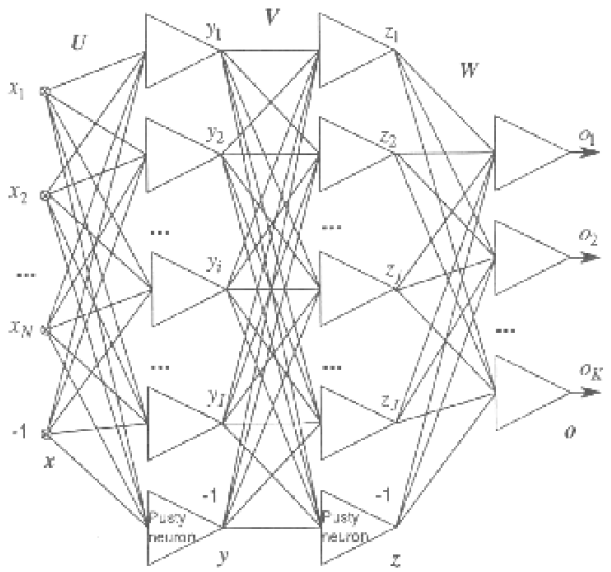
2.3.3 Podział ze względu na strukturę

Ze względu na **topologię** sieci neuronowe można podzielić na 2 grupy. Pierwsza grupa to sieci ze sprzężeniem zwrotnym, gdzie występują połączenia wsteczne między warstwami. Druga to sieci jednokierunkowe, w których sygnał przechodzi przez każdy neuron jednokrotnie.

Sieci jednokierunkowe

Modelowanie złożonych zadań, odbywa się poprzez zwielokrotnianie pojedynczych neuronów. Neurony te zorganizowane są w warstwy, z których połączenia powstaje sieć. Najprostsza sieć neuronowa składa się z warstwy neuronów wejściowych i warstwy neuronów wyjściowych. Każde z wyjść warstwy wejściowej połączone jest poprzez odpowiednie wagi i sumowanie, z wszystkimi wejściami warstwy wyjściowej. Typowe sieci neuronowe posiadają zwykle także dodatkowe warstwy pomiędzy wejściową i wyjściową, zwane warstwami ukrytymi (rys. 2.7).

Macierze **U,V,W** (rys. 2.7) zawierają współczynniki wagowe dla wszystkich połączeń synaptycznych.



x=[x_1 ,x_2 ,...,x_N ,-1] wektor wejściowy

y=[y_1 ,y_2 ,...,y_I ,-1] wektor wyjściowy pierwszej warstwy ukrytej

z=[z_1 ,z_2 ,...,z_J ,-1] wektor wyjściowy drugiej warstwy ukrytej

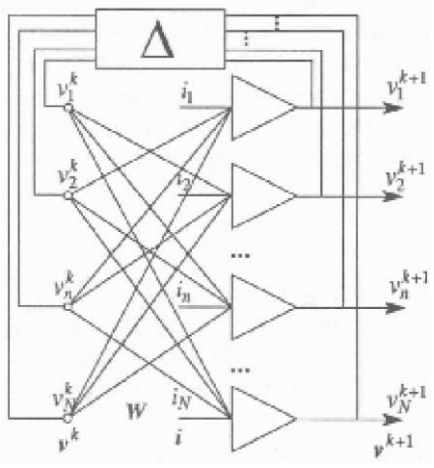
o=[o_1 ,o_2,...,o_K] wektor wyjściowy sieci

Rys. 2.7. Budowa sieci jednokierunkowej

Wynikową odpowiedź sieci określa się analizując pobudzone neurony na jej wyjściu. Przykładowo, jeżeli sieć posiada 5 neuronów w warstwie wyjściowej, z których każdy może znajdować się w 2 możliwych stanach (pobudzony/niepobudzony), liczba klas, które mogą zostać w ten sposób opisane to 2^5=32. Częściej jednak, stosuje się podejście, gdzie każdy z neuronów warstwy wyjściowej przyporządkowany jest do jednej odpowiadającej mu klasy. Stworzona w ten sposób sieć staje się bardziej rozbudowana, jednak znacznie upraszcza to proces analizy odpowiedzi.

Sieci ze sprzężeniem zwrotnym

W tej strukturze wyjście przynajmniej jednego neuronu jest połączone bezpośrednio lub pośrednio z jego wejściem przez np. opóźnienie lub tłumienie (rys. 2.8).



Rys. 2.8. Budowa sieci ze sprzężeniem zwrotnym

Przykładem sieci ze sprzężeniem zwrotnym jest sieć Hopfielda. Sieć ta może mieć:

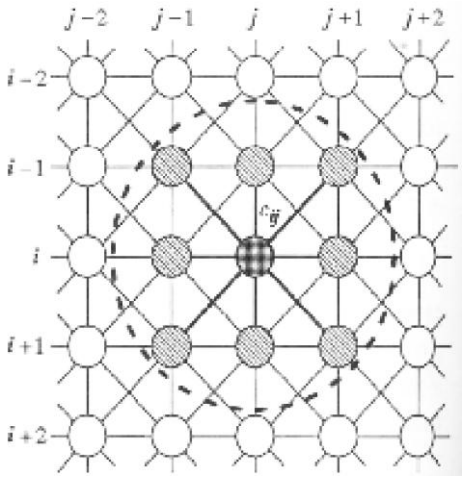
- *k* neuronów w warstwie wejściowej (wektor *k*-elementowy jest analizowanym sygnałem),
- *n* < *k* neuronów w warstwie ukrytej,
- *k* neuronów w warstwie wyjściowej.

Skoro wewnątrz sieci następuje zmniejszenie reprezentacji danych do wektora *n*-elementowego, to wynikiem jest uogólnienie danych wejściowych, zmniejszenie ilości danych. Jeżeli sieć jest w stanie na wyjściu uzyskać te same *k* wartości co na wejściu, to uzyskiwana jest w ten sposób kompresja danych w stosunku *n* : *k*. Takie podejście spotyka się w sieciach nazywanych autokoderami (ang. *autoencoders*) stosowanych dla przetwarzania i rozpoznawania grafiki. Sieć reprezentuje wejściowe obrazy w warstwach ukrytych w postaci uproszczonych cech szczególnych tych obrazów. Możliwe jest zinterpretowanie wag jako elementów obrazu – np. charakterystycznych układów linii lub części twarzy. Następnie w warstwie wyjściowej sieć dokonuje rekonstrukcji z cech szczególnych do oryginalnego obrazu. Jest to część metody tzw. głębokiej nauki sieci (ang. *deep learning*), implementowanej m.in. przez Google[1].

Sieci komórkowe

Topologia sieci komórkowej oparta jest na regularnej strukturze geometrycznej. Przykładem takiej sieci jest topologia płaskiej siatki prostokątnej (rys. 2.9):

- Neurony zgrupowane w I wierszach i J kolumnach
- Dowolna komórka połączona jest tylko z neuronami w najbliższym sąsiedztwie (konieczna definicja promienia sąsiedztwa – np. promień = 1)



Rys. 2.9. Budowa sieci komórkowej. Dana komórka połączona jest ze swoim najbliższym sąsiedztwem

Tego typu sieci stosowane są do przetwarzania obrazów, gdzie wartość wynikowego piksela zależy od wartości oryginalnej i pikseli z pewnego zadanego sąsiedztwa.

2.4 Budowanie i trening sieci neuronowych

Sieci neuronowe dostępne są w wielu postaciach w oprogramowaniu naukowym MATLAB, w pakiecie WEKA, w postaci bibliotek dla popularnych języków programowania. We wszystkich tych

[1] http://www.technologyreview.com/featuredstory/513696/deep-learning/

przypadkach konieczne jest poznanie prostych zasad określania struktury sieci, przygotowania danych i trenowania.

2.4.1 Dobór struktury sieci

Sieci neuronowe wymagają od użytkownika wyboru właściwej architektury (struktury sieci), dostosowanej do danego zagadnienia i przygotowania danych do nauki.

Sieć jednokierunkowa musi mieć w warstwie wejściowej tyle neuronów ile wynosi wymiar wektora danych wejściowych. Dla sygnałów będących pomiarami wartości zmiennych w czasie, np. zapisów EEG, EKG, w praktyce do sieci dostarcza się wycinek takiego sygnału, próbki z okna czasowego o określonej długości. Próbka pierwsza trafi do pierwszego neuronu, druga do drugiego, itd. Następnie konieczne jest określenie liczby warstw ukrytych (zwykle jedna warstwa jest wystarczająca, może także nie być warstwy ukrytej) i liczb neuronów w tych warstwach. Zalecane jest użycie mniejszej liczby neuronów ukrytych niż wejściowych. Warstwa wyjściowa z kolei musi mieć tyle neuronów, aby łatwa była interpretacja jej odpowiedzi, np. równą liczbie rozpoznawanych klas.

Ponadto od użytkownika wymaga się podstawowych umiejętności wyboru i przygotowania danych do treningu i testowania sieci. Wszystkie dostępne, sklasyfikowane (przez eksperta lub w wyniku eksperymentów) dane podzielone zostają na trzy rozłączne grupy, a próbki/obiekty między tymi grupami nie powinny się powtarzać:

a) dane uczące (ang. *training data*) – do właściwego treningu, automatycznej zmiany wag w sieci neuronowej

b) dane weryfikujące (ang. *validation data*) – do treningu, stosowane w celu sprawdzania uzyskiwanych decyzji sieci i zapobiegania przetrenowaniu (opisane w części kolejnej)

c) dane testowe (ang. *test data*) – do testowania gotowej sieci po treningu, symulują przypadki jeszcze nie znane, które napotkać można w przyszłości.

Wielkość tych zbiorów danych może być różna, np. w stosunku 1:1:1 lub 3:1:1. Trening może być powtarzany kilkukrotnie, dla większych i mniejszych zbiorów danych uczących, a wygenerowane sieci porównywane między sobą.

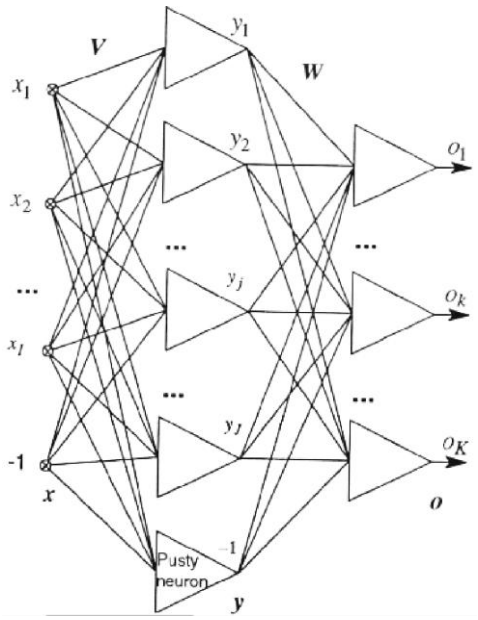
Interpretacja odpowiedzi sieci. Przykładowa sieć klasyfikująca dane do 2 klas może mieć jeden neuron, którego odpowiedź **y** o niskiej wartości (np. bliskiej zeru) będzie interpretowana jako klasa pierwsza, a odpowiedź **y** o wysokiej wartości (np. bliska jedynce) jako klasa druga. Możliwe jest także użycie dla dwóch klas 2 neuronów wyjściowych, wówczas za decydujący uznawany jest ten, którego odpowiedź jest „silniejsza”. Oczywiście nastąpić może niejednoznaczna odpowiedź sieci, gdy dla wersji sieci z jednym neuronem wyjściowym wartość jego odpowiedzi jest zbliżona do 0,5, a dla sieci z

dwoma – gdy dwie odpowiedzi mają podobne wartości. Sieć dająca takie niejednoznaczne odpowiedzi wymaga dostarczenia dodatkowych danych trenujących.

Sieci neuronowe **nie wymagają** od użytkownika posiadania specjalistycznej wiedzy teoretycznej niezbędnej do zbudowania modelu matematycznego (sieć buduje model sama w procesie treningu). Poziom wiedzy teoretycznej niezbędnej do zbudowania skutecznego modelu jest przy stosowaniu sieci neuronowych znacznie niższy niż w przypadku stosowania tradycyjnych metod statystycznych lub innych metod wnioskowania, stąd też sieci są popularną metodą klasyfikacji i wspierania podejmowania decyzji.

2.4.2 Trening sieci neuronowej

Rozważmy jednokierunkową sieć neuronową (rys. 2.10), przyjmującą na wejście wektor **x** o liczbie elementów I, (plus dodatkowy element – wejście progowe równe „-1”), składającą się z warstw: wejściowej (wektor **y**), zawierającej J neuronów, wyjściowej (wektor **o**), zawierającej K neuronów.



$\mathbf{x} = [x_1, \ldots, x_I, -1]$

$\mathbf{y} = [y_1, \ldots, y_J, -1]$

$\mathbf{o} = [o_1, \ldots, o_K]$

Rys. 2.10. Przykładowa sieć neuronowa ilustrująca zasadę treningu

Z powodu dużej liczby neuronów i połączeń między nimi wagi wygodnie jest reprezentować w postaci macierzy (2.3):

$$\mathbf{V} = \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1I} \\ v_{21} & v_{22} & \cdots & v_{2I} \\ \cdots & \cdots & \cdots & \cdots \\ v_{J1} & v_{J2} & \cdots & v_{JI} \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1J} \\ w_{21} & w_{22} & \cdots & w_{2J} \\ \cdots & \cdots & \cdots & \cdots \\ w_{K1} & w_{K2} & \cdots & w_{KJ} \end{bmatrix} \tag{2.3}$$

Waga v_{ji} określa, że na wejście j-tego neuronu warstwy wejściowej trafi wartość wejściowa x_i pomnożona przez v_{ij}.

Waga w_{kj} określa, że do k-tego neuronu wyjściowego trafi y_j pomnożony przez w_{kj}.

Zastosowanie indeksów dolnych pozwala wskazywać na neuron odpowiedniej warstwy (np. $f_{o1}(net)$ to neuron pierwszy warstwy o).

Wektory pochodnych funkcji aktywacji (2.4):

$$f_y' = [f_{y1}'(net_{y1}), f_{y2}'(net_{y2}),..., f_{yJ}'(net_{yJ})]^T$$
$$f_y' = [f_{o1}'(net_{o1}), f_{o2}'(net_{o2}),..., f_{oK}'(net_{oK})]^T \quad (2.4)$$

gdzie: $net_y = \mathbf{v}^T\mathbf{x}$ $net_o = \mathbf{w}^T\mathbf{y}$

oraz: $net_{ok} = w_{k1}y_1+w_{k2}y_2 + \ldots + w_{kJ}y_J$

Wykorzystać można operator $\Gamma(.)$, który uprości zapis do postaci macierzowej, zniweluje potrzebę używania wprost nazw funkcji f_i od argumentów q_1 (2.5).

$$\Gamma[q] = \begin{bmatrix} f_1(q_1) & 0 & \ldots & 0 \\ 0 & f_2(q_2) & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & \ldots & f_Q(q_Q) \end{bmatrix} \quad (2.5)$$

wtedy odpowiedź sieci zapisać można jako (2.6):

$$\mathbf{o} = \Gamma[\mathbf{W}\mathbf{y}] = \Gamma[\mathbf{W} \cdot \Gamma[\mathbf{V}\mathbf{x}]] \quad (2.6)$$

Odczytując ten zapis od prawej strony: $\Gamma(\mathbf{Vx})$, to wynik działania pierwszej warstwy, elementy wektora **x** pomnożone przez wagi **V**, trafiają na funkcje aktywacji neuronów, uzyskiwana jest odpowiedź z całej pierwszej warstwy ($\Gamma(\mathbf{Vx}) = \mathbf{y}$). Ta z kolei wymnażana jest przez wagi **W** i podawana na funkcje drugiej warstwy, uzyskiwane jest **o**.

Błąd uczenia

Metoda uczenia jest metodą nadzorowaną, tj. odpowiedź sieci porównuje się z oczekiwaną odpowiedzią, dlatego można określić miarę wyrażającą różnicę między wskazaniem sieci (symbol **o**) a oczekiwaną odpowiedzią sieci (symbol **d**), czyli tzw. funkcję błędu E.

Najczęściej do wyliczenia różnicy między dwoma wektorami stosuje się odległość średniokwadratową:

$$E^n = \sum_{p=1}^{P}\sum_{k=1}^{K}\frac{1}{2}\cdot\left(d_k^{(p)} - o_k^{(p)}\right)^2 = \frac{1}{2}\cdot\sum_{p=1}^{P}\sum_{k=1}^{K}\left(d_k^{(p)} - o_k^{(p)}\right)^2 \quad (2.7)$$

co można zapisać także jako:

$$E^n = \frac{1}{2}\cdot\sum_{p=1}^{P}\left\|\mathbf{d}^{(p)} - \mathbf{o}^{(p)}\right\|^2 \quad (2.8)$$

gdzie: *p* jest numerem próbki trenującej, dla której oczekiwana jest odpowiedź $\mathbf{d}^{(p)}$ a uzyskiwane jest $\mathbf{o}^{(p)}$.

Błąd ten liczony jest dla wszystkich P wektorów ze zbioru uczącego – błąd skumulowany. Podczas treningu prezentowane są kolejno wektory uczące, wówczas funkcja błędu dla *p* – tego wektora przyjmuje postać:

$$E(p)^n = \rho\left(\mathbf{d}^{(p)}, \mathbf{o}^{(p)}\right) = \frac{1}{2} \cdot \left\| \mathbf{d}^{(p)} - \mathbf{o}^{(p)} \right\|^2 \qquad (2.9)$$

W dalszej części przyjęto, że rozważania dotyczą właśnie pojedynczego *p*-tego wektora ze zbioru uczącego.

Aktualizacja wag

Funkcja błędu skumulowanego wyliczana jest dla każdej warstwy osobno. Szczególnie interesujące jest sprawdzenie pochodnej funkcji błędu, która odpowie na pytanie, w jakim kierunku zmienia się błąd i jak modyfikować wagi sieci aby uzyskać zmniejszenie tego błędu. W tym celu w warstwach wylicza się gradienty (operator nabla - ∇)(2.10)(42.11). W gradiencie δ_o *k*-ty składnik to pochodna cząstkowa **E** po *k*-tym wejściu neuronu net_k:

- warstwa wyjściowa

$$\delta_o = -\nabla E(\mathbf{o}) \quad \text{, gdzie:} \quad \delta_{ok} = -\frac{\partial E}{\partial net_k} \qquad (2.10)$$

- warstwa ukryta

$$\delta_y = -\nabla E(\mathbf{y}) \quad \text{, gdzie:} \quad \delta_{yj} = -\frac{\partial E}{\partial net_j} \qquad (2.11)$$

Dla łatwiejszego zapisania funkcji błędu posłużyć się można pomocniczym operatorem:

$$\Phi[q] = \begin{bmatrix} q_1 & 0 & \dots & 0 \\ 0 & q_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & q_Q \end{bmatrix} \qquad (4.12)$$

Wtedy składniki [**d – o**] z funkcji błędu:

$$\delta_o = \Phi[\mathbf{d} - \mathbf{o}] \cdot \mathbf{f}_o' \qquad \text{gdzie:} \quad \delta_{ok} = (d_k - o_k) \cdot f'(net_k) \qquad (2.13)$$

W warstwie wyjściowej, gdzie jak pokazano wcześniej $\mathbf{o}=\mathbf{w}^T\mathbf{y}$, uzyska się wówczas:

$$\delta_y = \mathbf{w}_j^T \cdot \delta_o \cdot \mathbf{f}_y' \quad \text{lub:} \quad \delta_{yj} = f_j'(net_j) \cdot \sum_{k=1}^{K} (d_k - o_k) \cdot f_k'(net_k) \cdot w_{kj} \qquad (2.14)$$

Podstawiając do powyższych równań obecny stan sieci (wytrenowane wagi **w** i **v**) oraz kolejne próbki testowe **x**, oczekiwane dla nich odpowiedzi **d** i uzyskane odpowiedzi **o**, uzyskuje się gradienty

funkcji błędu δ. Korzystanie z pochodnej f'_k narzuca wspominany wcześniej warunek, aby funkcje aktywacji neuronów f były ciągłe i różniczkowalne.

Następnie można modyfikować wagi tak, aby poruszać się zgodnie z kierunkiem malejącym tego gradientu, czyli krok po kroku zmniejszać błąd – różnicę między oczekiwanymi odpowiedziami **d** i uzyskanymi odpowiedziami **o**. Taka modyfikacja wag zwykle wykonywana jest zgodnie z **regułą delta**, która uzależnia nowe wagi w n+1 kroku nauki ($\mathbf{V}^{n+1}$, $\mathbf{W}^{n+1}$) od gradientów błędów uzyskanych przez sieć, która korzystała z wag w kroku n (wagi $\mathbf{V}^n$, $\mathbf{W}^n$):

$$\begin{cases} \Delta \mathbf{V}^{n+1} = -\eta \cdot \nabla E(\mathbf{V}^n) \\ \Delta \mathbf{W}^{n+1} = -\eta \cdot \nabla E(\mathbf{W}^n) \end{cases} \tag{2.15}$$

gdzie: η jest współczynnikiem szybkości treningu.

Zapis (2.15) oznacza, że korekta (Δ - delta) dla kroku n+1 wyliczona jest z gradientów ∇E. Współczynnik szybkości treningu η reguluje jak bardzo gradient ma wpływ na korektę wag.

W wyniku podstawienia (2.10)(2.11) do (2.15) otrzymujemy ostatecznie wzór na zaktualizowane wagi:

$$\begin{cases} \mathbf{V}^{n+1} = \mathbf{V}^n + \eta\, \delta_y\, \mathbf{x}^T \\ \mathbf{W}^{n+1} = \mathbf{W}^n + \eta\, \delta_o\, \mathbf{y}^T \end{cases} \tag{2.16}$$

Zapis (2.16) należy rozumieć następująco: wagi w kolejnym kroku nauki są równe wagom w poprzednim kroku powiększonym proporcjonalnie do iloczynów wektorów wejściowych warstw i wektorów błędu działania sieci.

2.5 Zbieżność procesu nauki

Minimalizacja funkcji błędu opisana powyżej jest oparta o metody gradientowe i nie gwarantuje zbieżności nauki. W zadaniach optymalizacji (w tym przypadku minimalizacji błędu) zawsze napotyka się na ryzyko nieznalezienia optimum. Metoda gradientowa szczególnie jest podatna na występowanie **lokalnych minimów**, z których się nie „wydostaje”, gdyż w najbliższym sąsiedztwie aktualnego punktu w wielowymiarowej przestrzeni wag gradienty we wszystkich kierunkach są dodatnie.

Dla małych wartości współczynnika szybkości treningu η nauka może nie przynosić poprawy, jeżeli funkcja błędu jest płaska w analizowanym obszarze.

Aby poprawić właściwości nauki stosuje się dodatkowy składnik **momentu**, uzależniający przyrost wag od przyrostu wag w poprzednim kroku nauki. Po uwzględnieniu w (2.16) składnika momentu α, wzory wartości macierzy wag w kroku n +1 mają następującą postać:

$$\begin{cases} \mathbf{V}^{n+1} = \mathbf{V}^{n} + \eta\,\delta_{y}\,x^{T} + \alpha\cdot\Delta\mathbf{V}^{n} \\ \mathbf{W}^{n+1} = \mathbf{W}^{n} + \eta\,\delta_{o}\,y^{T} + \alpha\cdot\Delta\mathbf{W}^{n} \end{cases} \tag{2.17}$$

Moment wprowadza do algorytmu element bezwładności, który zmniejsza chwilowe i gwałtowne zmiany kierunku wskazywanego przez gradient funkcji błędu. Dzięki temu uczenie nie wchodzi w płytkie minima lokalne, a to znacznie przyspiesza naukę dla płaskich obszarów funkcji błędu oraz pozwala „wyjść” z minimów lokalnych. **Właściwy dobór współczynników nauki** η i α umożliwia wyjście z minimów lokalnych funkcji błędu i szybkie osiągnięcie wartości bliskich jej minimum globalnego.

2.6 Algorytm wstecznej propagacji błędu

Opisane powyżej postępowanie pokazało, w jaki sposób korzystając z wektorów/próbek uczących i z **błędu** (różnicy między aktualną odpowiedzią sieci a oczekiwaną odpowiedzią) można aktualizować wagi sieci, aby uzyskać zmniejszenie tego błędu. Takie podejście – od sygnału wyjścia, do warstwy ostatniej, następnie do przedostatniej, itd., – nazywane jest wsteczną propagacją błędu. Podsumowując można proces ten podzielić na kilka, łatwych do zalgorytmizowania kroków:

Krok 1:

• Inicjalizacja macierzy wag **V** i **W** (lub większej ich liczby, jeżeli warstw sieci jest więcej) małymi, losowymi wartościami z zakresu (-1,1)

• Ustawianie parametrów nauki sieci:

- funkcji aktywacji neuronów
- parametrów nauki - η i α

Krok 2:

• Ustawienie wartości błędu skumulowanego na **E**=0 dla każdego nowego cyklu treningowego. W wyniku oceny wszystkich wejściowych wektorów **x**, w każdym cyklu treningowym, ten błąd wyliczany jest od nowa – weryfikowane jest czy **E** spada i jaki jest jego gradient.

Krok 3:

• Wybór dowolnego wektora ze zbioru uczącego, najlepiej wybór losowy. Odczytanie z danych uczących oczekiwanej odpowiedzi sieci **d**.

Krok 4:

• Wyznaczanie odpowiedzi warstw sieci **y**, **o** lub większej liczby odpowiedzi, jeżeli warstw jest więcej, poprzez obliczanie wyjść z neuronów.

Krok 5:

• Obliczanie sygnałów błędów dla kolejnych warstw: różnic między oczekiwaną odpowiedzią a uzyskaną.

Krok 6:

• Obliczanie nowych wartości wag, z uwzględnieniem momentu lub bez (w zależności od wyboru metody).

Krok 7:

• W krokach od 3. do 6. obliczana jest w ten sposób wartość funkcji błędu dla wektora **x**. Wartość ta dodawana jest do wartości błędu skumulowanego **E**.

Krok 8:

• Jeśli wektor **x** nie jest ostatnim wykorzystywanym wzorcem, to algorytm wraca do kroku 3. Jeśli wektor **x** jest ostatnim wektorem uczącym to przechodzi się do kroku 9.

Krok 9:

• Sprawdzany jest warunek czy wartość błędu skumulowanego jest mniejsza od zadanej progowej wartości $\mathbf{E}_{min}$. Jeśli tak, to trening się zatrzymuje. Jeśli warunek ten nie jest spełniony to następuje kolejny cykl treningowy i powrót do kroku 2.

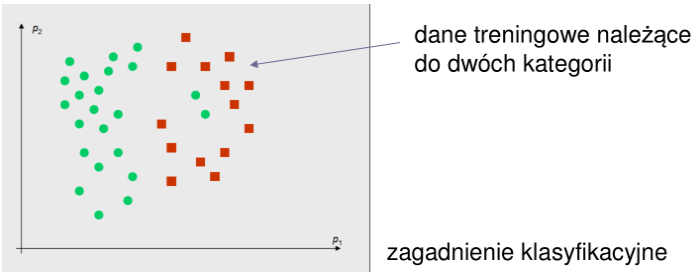
Należy mieć na uwadze, że odpowiednio długo wykonywany trening może doprowadzić do sytuacji, że wszystkie próbki uczące x_i będą klasyfikowane w pełni prawidłowo i błąd wyniesie 0. Poniżej pokazane zostanie, że nie jest to jednak sytuacja pożądana, gdyż może być symptomem **przetrenowania**.

2.7 Generalizacyjne własności sieci

Gdy algorytm treningu powtarzany jest w zbyt wielu krokach dojść może do przeuczenia/przetrenowania. Sieć przetrenowana bardzo dobrze klasyfikuje obiekty ze zbioru treningowego, ale osiąga znaczne gorsze wyniki dla obiektów spoza tego zbioru. Pożądane jest, aby sieć równie dobrze rozpoznawała i klasyfikowała obiekty, którymi była trenowana oraz dowolne inne (oczywiście reprezentujące klasy, których wyuczona została sieć). Taka cecha sieci nazywana jest zdolnością generalizacji – nazwa ma oznaczać, że w wyniku treningu sieć była w stanie z wielu danych wydobyć istotę problemu, „zrozumieć” faktyczne różnice między klasami i próbkami i ostatecznie wykorzystać tę nabytą wiedzę. Generalizacja oznacza również nieprzywiązywanie wagi do atrybutów nieistotnych[2].

[2] Dokładniejsza dyskusja zamieszczona jest w rozdziale dotyczącym drzew decyzyjnych.

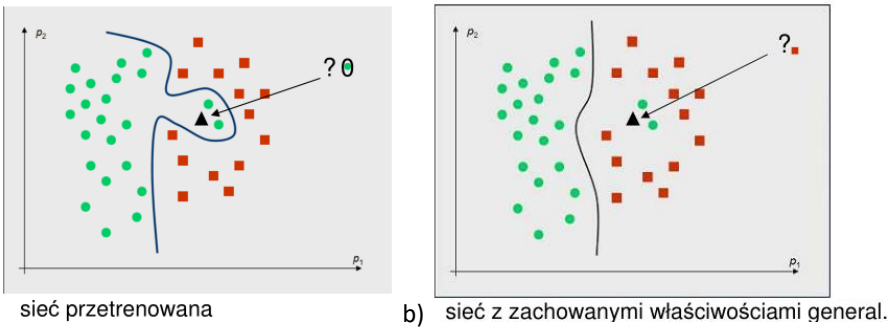
Z czym należy kojarzyć przetrenowanie? Poniższa graficzna interpretacja (rys. 2.11 – 2.12) w prosty sposób wyjaśnia ryzyko zbyt dokładnego wyuczenia się danych treningowych. W zbiorze treningowym atrybuty próbek mogą być obarczone błędami pomiaru: może zawiodło urządzenie zbierające te dane, może ekspert czy lekarz pomylił się, może nie uwzględniono nieznanego zmiennego czynnika, przez co doszło do pojawienia się przypadków odbiegających od grupy (zielone kropki leżące w prawej części wykresu, wśród czerwonych kwadratów).



Rys. 2.11. Przykład zbioru danych z błędami pomiaru: dwa zielone punkty w prawej części mają błędnie pomierzoną wartość p1 lub są błędnie rozpoznane jako klasa zielona

Sieć trenowana dostatecznie długo dostosuje swoją strukturę w taki sposób, aby te dwa problematyczne przypadki rozpoznać prawidłowo (tzn. zgodnie z decyzją podaną w zbiorze treningowym). Można to graficznie przedstawić jako nieregularny kształt krzywej rozdzielającej na płaszczyźnie punkty, które sieć przydzieli do grupy pierwszej i do drugiej. Nowy przypadek oznaczony symbolem czarnego trójkąta, zostanie przypisany przez sieć przetrenowaną do klasy kółek (rys. 2.12a).

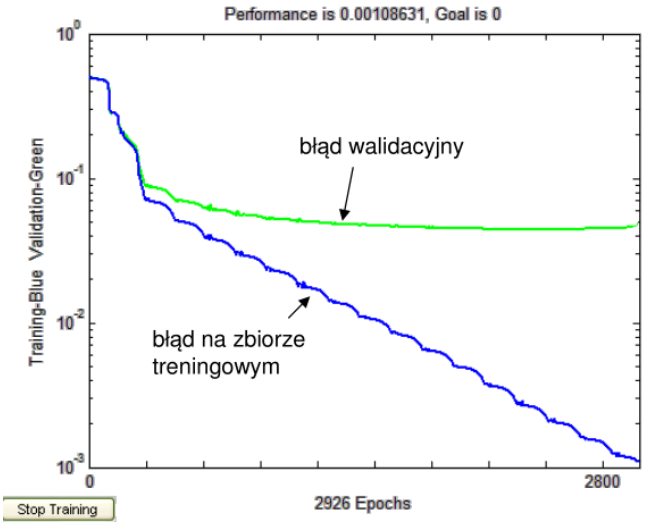
Tymczasem nieprzetrenowana sieć, nie nauczy się zaliczać dwóch problematycznych próbek do klasy kółek i wówczas nieznany przypadek zaklasyfikuje do kwadratów (rys. 2.12b). Według miary błędu treningowego będzie to gorsza sieć, ale dla danych nowych, na których nie była trenowana, działać będzie lepiej.



Rys. 2.12. Porównanie działania sieci: a) sieć przetrenowana, b) sieć nieprzetrenowana

Na początku rozdziału wspomniane zostało, że dostępne dane dzielone są na zbiór treningowy, walidacyjny i testowy. Ma to na celu w każdym kroku treningu sprawdzanie działania sieci dla przypadków nieznanych. Po wykonaniu jednej iteracji nauki, na wejście sieci podaje się próbki walidacyjne i sprawdza się uzyskiwany wynik, nazywany błędem walidacyjnym (wylicza się tylko błąd, bez aktualizacji wag, gdyż na danych walidacyjnych sieć nie ma się uczyć). Uzyskany błąd walidacyjny

jest zapamiętywany, jego zmiany śledzone w czasie nauki, gdyż na tej podstawie podejmowana jest decyzja o wcześniejszym jej zakończeniu (rys. 2.13).



Rys. 2.13. Interfejs MATLAB prezentujący porównanie błędu treningowego i walidacyjnego w kolejnych iteracjach treningu (ang. *epochs*)

W każdym kroku nauki uzyskiwana jest poprawa działania dla danych treningowych (wykres niebieski), **błąd treningowy spada**. Jednocześnie także **spada błąd walidacyjny** (wykres zielony), ale zauważyć można, że od pewnej iteracji sieć nie poprawia skuteczności działania, błąd walidacyjny spada coraz wolniej i nawet zaczyna wzrastać. W tym przykładzie około kroku 2200 wykres staje się płaski, a powyżej 2800 rośnie. Trening należy przerwać przed tym wzrostem.

Kontrola błędu walidacyjnego w każdym kroku nauki uniemożliwia przetrenowanie sieci. Ostatecznie dla wytrenowanej sieci, która dla danych treningowych i walidacyjnych uzyskuje zadawalające wyniki dokonuje się jeszcze pomiaru **błędu testowego**, na danych testowych. To jest prawdziwy sprawdzian sieci – działanie na danych niedostępnych dla algorytmu nauki – co symuluje rzeczywiste wykorzystanie sieci w przyszłości.

2.8 Przegląd zastosowań

Sieć neuronowa pełni zawsze rolę aproksymatora pewnej idealnej funkcji wielu zmiennych, która (gdyby było możliwe jej zdefiniowanie) dokonywałaby optymalnego rozpoznania, klasyfikacji, lub innego przetworzenia danych treningowych, walidacyjnych i testowych. W procesie nauki sieci parametry funkcji aproksymującej, tej rzeczywiście realizowanej przez sieć, są tak zmieniane, aby popełniane błędy były jak najmniejsze – przybliżając ją do poszukiwanej funkcji idealnej. Duża liczba zadań modelowania, identyfikacji, przetwarzania sygnałów da się sprowadzić do takiego zagadnienia aproksymacyjnego.

2.8.1 Najważniejsze zastosowania

Przy **klasyfikacji i rozpoznawaniu** wzorców sieć uczy się podstawowych cech tych wzorców, takich jak odwzorowanie geometryczne układu pikseli uczonego obrazu, rozkładu składników cech statystycznych wzorca, czy jego innych parametrów. Dobre uczenie polega na podawaniu wzorców o dużych różnicach, stanowiących podstawę podjęcia decyzji przypisania ich do odpowiedniej klasy. Należy „pokazać” sieci dostatecznie dużo różnorodnych próbek, które trafiają do tej samej klasy oraz do klas różnych.

Przy **predykcji** zadaniem sieci jest określenie przyszłych odpowiedzi systemu na podstawie ciągu wartości z przeszłości (np. próbek sygnału). Mając informacje o wartościach zmiennej **x** w chwilach poprzedzających predykcję x(k-1), x(k-2) … x(k-N), sieć podejmuje decyzję, jaka będzie estymowana wartość x(*k*) badanego ciągu w chwili aktualnej *k*. Może to mieć zastosowanie w rekonstrukcji sygnałów akustycznych, redukcji szumu w obrazie, dźwięku i innych danych („uszkodzone” próbki zastępowane są właściwymi, predykowanymi przez sieć na podstawie dobrych próbek wcześniejszych).

W zagadnieniach **identyfikacji i sterowania** procesami dynamicznymi sieć neuronowa pełni zwykle kilka funkcji. Stanowi model nieliniowy tego procesu, pozwalający na wypracowanie odpowiedniego sygnału sterującego. Pełni również funkcję układów śledzącego i nadążnego, adaptując się do warunków środowiskowych – w tej dziedzinie najczęściej stosuje się sieci ze sprzężeniem zwrotnym.

W zadaniach **asocjacji** sieć neuronowa pełni rolę pamięci skojarzeniowej. Można wyróżnić pamięć asocjacyjną, w przypadku której skojarzenie dotyczy tylko poszczególnych składowych wektora wejściowego oraz pamięć heteroasocjacyjną, gdzie zadaniem sieci jest skojarzenie ze sobą dwóch wektorów. Jeśli na wejście sieci podany będzie wektor odkształcony (np. o elementach zniekształconych szumem bądź pozbawiony pewnych elementów danych), sieć neuronowa jest w stanie odtworzyć wektor oryginalny, pozbawiony szumów, generując przy tym pełną postać wektora stowarzyszonego z nim. Przykładowo obraz zeskanowanego tekstu z zabrudzonej, uszkodzonej kartki, zawierający wyuczone czcionki może zostać przywrócony do pierwotnej postaci, gdyż w miejsce czcionek „przypominających” i nasuwających sieci pewne skojarzenia podstawione zostaną obrazy czcionek oryginalnych.

2.8.2 Klasyfikator neuronowy - dyskretny dychotomizator

W pierwszej kolejności należy wyjaśnić terminy:

- „Dyskretny” - zwracający decyzję w formie liczb całkowitych „1” i „2”;

- „Dychotomizator” - dokonujący rozróżnienia wszystkich obiektów na dwie klasy.

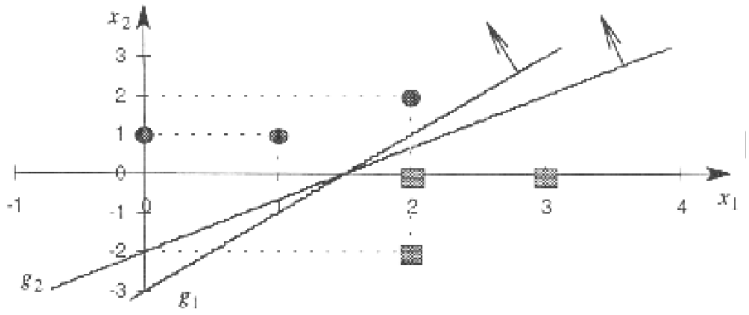
Klasyfikator taki jest w stanie przydzielić dowolny n–wymiarowy obiekt do jednej z dwóch klas. Ma to szczególne praktyczne zastosowanie, gdyż klasy te mogą dotyczyć „akceptacji” lub „odrzucenia” jakiegoś obiektu, co pożądane jest w wielu rzeczywistych aplikacjach.

Klasyfikator ten może składać się tylko z jednego neuronu o (n+1) wejściach: n wejść na które podawane są wartości atrybutów i jedno wejście progowe.

Decyzja podejmowana jest na podstawie uzyskanej wartości wyjście neuronu:

- jeżeli **y** ≥ 0 - klasyfikacja do klasy „1”
- jeżeli **y**<0 - klasyfikacja do klasy „2”

Dla 2-wymiarowego zagadnienia łatwo to zobrazować na płaszczyźnie: rozrzucone punkty dwóch różnych klas rozdzielić można skutecznie wieloma prostymi (przykładowo g_1 i g_2), które tak samo dobrze realizować będą to zadanie (rys. 2.14). Strzałki pokazują półpłaszczyznę, dla której wartości y są dodatnie – klasa „1”.



Rys. 2.14. Przykład klasyfikacji dwóch klas: proste g_1 i g_2 tak samo skutecznie realizują to zadanie

Proste g_1 i g_2 mają równania g_1 : $-2x_1 + x_2 + 3 = 0$; g_2: $-4x_1 + 3x_2 + 6 = 0$. Wagi wejściowe neuronu, który dokonać ma takiej samej klasyfikacji będą równe współczynnikom powyższych wielomianów. Zauważyć można, że $g_i=\mathbf{w}^T\mathbf{x}$, gdzie np. dla g_1: $\mathbf{w}=[-2, 1, 3]$ i $\mathbf{x}=[x_1, x_2, 1]$, czyli prosta definiuje wagi dla jednego neuronu. I odwrotnie – jeden neuron interpretowany może być jako prosta na płaszczyźnie.

Jeżeli obiekty opisane będą większą liczbą atrybutów, np. dowolne *k*, to wyobrażać można sobie przestrzenie *k*-wymiarowe, w których rozdzielenie klas realizowane jest przez hiperpłaszczyznę.

Dychotomizator taki można wykorzystywać kilkukrotnie w formie wielowarstwowej sieci neuronowej, np. równolegle w celu klasyfikacji najpierw grup: klasa A kontra inne, klasa B kontra inne, itd., lub kaskadowo np. klasy (A+B) kontra klasy (C+D), a następnie wynik podać na dwa kolejne neurony klasa A kontra klasa B i klasa C kontra klasa D. Proponowane jest aby czytelnik samodzielnie wykonał schemat blokowy takich struktur i wykreślił na płaszczyźnie przykłady czterech klas i sposobów ich rozdzielania prostymi.

W praktyce nie ma potrzeby narzucania sieci jednego z powyższych podejść. W trakcie treningu wieloneuronowa i wielowarstwowa sieć sama określi swoje wagi i nie będzie potrzeby analizowania

jakiej klasyfikacji dokonuje pojedynczy neuron (co więcej, zadanie pojedynczego neuronu może być nawet bardzo trudne w interpretacji, o czym pisano na początku rozdziału).

2.9 Literatura

[1] Hertz J. et al, *Wstęp do teorii obliczeń neuronowych*, WNT, Warszawa, 1995

[2] Korbicz J., Obuchowicz A., Uciński D., *Sztuczne sieci neuronowe. Podstawy i zastosowania*, Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1994

[3] Tadeusiewicz R., *Sieci neuronowe*, Akademicka Oficyna Wydawnicza RM, Warszawa, 1993

[4] Żurada J., Barski M., Jędruch W., *Sztuczne sieci neuronowe*, PWN, Warszawa, 1996

[5] McCulloch W. S., Pitts W., *A logical calculus of the ideas immanent in nervous activity*, Bulletin of Mathematical Biophysics, No 5, 1943, pp. 115-133.

3 Logika rozmyta

3.1 Wprowadzenie

Wraz z postępem metod drążenia danych, wspomagania decyzji i dziedzin podobnych, okazało się, że klasyczne wnioskowanie oparte na dwuwartościowej logice Arystotelesa oraz na klasycznej definicji zbioru według Georga Cantora nie zawsze są adekwatne do badanych problemów. Szczególnie ma to miejsce w projektowaniu i eksploatacji systemów sterowania, gdy uzyskuje się z mierników i sensorów rzeczywiste parametry, obarczone błędami lub niejednoznaczne, nie dające się zinterpretować, które ponadto wskazywać mogą na sprzeczne decyzje. Dlatego zachodzi potrzeba wykorzystania innych narzędzi niż obliczenia komputerowe o wysokiej precyzji i logika *prawda-fałsz*.

W klasycznej teorii zbiorów stopień przynależności danego elementu do zbioru można określić za pomocą jednej z dwóch wartości: 0 – gdy element nie należy do danego zbioru i 1 – gdy element należy do danego zbioru. Wówczas trudno jest jednoznacznie określić stopień przynależności każdego parametru rzeczywistego, gdy jego wartość pochodzi z ciągłej dziedziny zmienności i ulokowana jest w pobliżu granicy zbiorów [6]. Przykładowo, temperatura ciała osoby zdrowej to 36,6°C. Wynik pomiaru 36,3°C albo 37,0°C może także nie niepokoić lekarza, ale czy 37,1°C już wskazuje na chorobę? W tym wypadku bardziej intuicyjna będzie możliwość określenia stopnia, w jakim dany pomiar wskazuje na stan chorobowy: 36,6 – nie; 37,0 – możliwe, 37,1 – pewne, itp.

W praktyce zaobserwować można kilka **rodzajów niepewności**:

Niepewność stochastyczna:

Np. rzut kostką, wypadek, ryzyko w ubezpieczeniach (zastosowanie ma rachunek prawdopodobieństwa) – celem jest stwierdzanie, jakie jest prawdopodobieństwo zajścia ściśle określonego zdarzenia.

Niepewność pomiarowa:

Około 3 cm; 20 punktów (zastosowanie ma statystyka) – celem jest stwierdzenie poziomu istotności uzyskanej wartości, estymacja wartości, ocena jakości pomiaru.

Niepewność informacyjna:

Ocena wiarygodności kredytobiorcy (zastosowanie ma drążenie danych, ang. *data mining*) – celem jest poszukanie zależności między atrybutami i decyzjami.

Niepewność lingwistyczna

Wyrażenie wartości w sposób słowny: mały, szybki, zimno, ciepło, drogo, tanio (zastosowane ma logika rozmyta) – celem jest wnioskowanie i otrzymywanie ścisłego

wyniku z wejściowych danych nieprecyzyjnych, podawanych w sposób słowny (często danych liczbowych, ale zamienianych na opis słowny).

Na potrzeby ostatniego typu niepewności, tj. dla przetwarzania danych lingwistycznych prof. Lofti Zadeh w 1965 roku zaproponował podejście nazywane logiką rozmytą (ang. *fuzzy logic* - FL) [8] lub przetwarzaniem wyrażeń języka naturalnego (ang. *computing with words*) [9]. Może być ona traktowana jako rozwinięcie dwuwartościowej logiki do postaci logiki wielowartościowej, gdyż istnieje prosta redukcja logiki rozmytej do logiki *prawda-fałsz*.

Logika rozmyta (FL) znalazła szerokie zastosowanie w szeroko rozumianej technice, zwłaszcza w systemach sterowania [3][6]. Systemy FL charakteryzują się prostotą i łatwością rozbudowy i modyfikacji. Zastosowanie intuicyjnego, lingwistycznego opisu atrybutów, wartości i reguł modelujących dany proces, upraszcza proces projektowania i walidacji [1][5][7].

3.2 Zbiór klasyczny a zbiór rozmyty

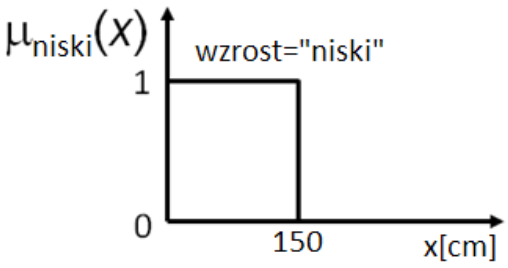
Przed wprowadzeniem pojęcia zbioru rozmytego, przypomnieć należy sposób określania zbioru tradycyjnego. Niech przykładowy zbiór zawiera wszystkie osoby niskie, np. o wzroście poniżej 150cm. Definicja tak rozumianego zbioru jest następująca:

$$\textbf{niski} = \{x \mid wzrost(x) < 150\} \tag{3.1}$$

Gdzie, x oznacza osobę, a *wzrost*() oznacza funkcję pomiaru wzrostu, która zwraca wartości w centymetrach. Funkcja przynależności do tego zbioru klasycznego jest określona jako odwzorowanie, przypisujące każdej wartości wejściowej x wartości wynikowe 1 lub 0:

$$\mu_{niski}(x) = \begin{cases} 1 : wzrost(x) < 150 \\ 0 : wzrost(x) \geq 150 \end{cases} \tag{3.2}$$

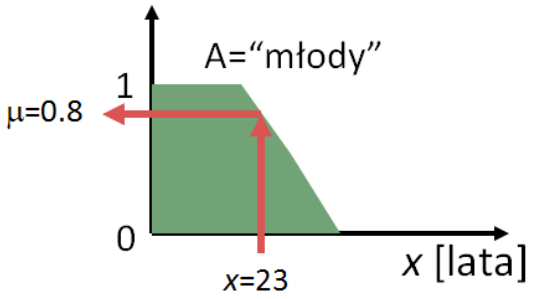
Wykreślona na osi funkcja przynależności nazywana jest funkcją charakterystyczną (rys. 3.1).



Rys. 3.1. Przykład funkcji charakterystycznej zbioru klasycznego – zawsze posiada ona kształt prostokątny, przyjmuje wyłącznie wartości 0 lub 1 (odpowiednio: nie należy, należy do zbioru).

Na osi poziomej oznaczane są wartości z dziedziny analizowanego atrybutu X, wartości x, opisującej obiekty z uniwersum; $x \in X$. Zmienna lingwistyczna – w tym przykładzie „wzrost”, o wartości lingwistycznej „niski”. Inne możliwe wartości lingwistyczne zmiennej „wzrost” to „wysoki”, „średni”, „bardzo wysoki” i inne, odpowiadające problemowi lub badanej grupie osób.

Jak podkreślono wcześniej, dla zbioru tradycyjnego przynależność wynosi zawsze 0 lub 1. Z kolei dla rozmytego – przyjąć może ona dowolną wartość pomiędzy 0 a 1 (rys. 3.2):



Rys. 3.2. Przykładowa funkcja przynależności dla zmiennej lingwistycznej „wiek”, wartości lingwistycznej „młody”. Osoba o liczbie lat równej *x* przynależy częściowo do zbioru. Stopień przynależności określa wartość funkcji

Zauważyć można, że o przynależności wyjściowej, czyli wartości funkcji μ decyduje kształt krzywej opisującej wartość lingwistyczną. Istnieją przedziały wartości *x* gdzie przynależność równa jest 1 i przedziały gdzie wartość ta to 0 – stanowi to analogię do zbioru tradycyjnego. Jednak występują wartości nie w pełni przynależące do zbioru – jest to istotą zbioru rozmytego.

Formalizując powyższe spostrzeżenia mówi się, że w teorii zbiorów rozmytych **element może należeć częściowo** do pewnego zbioru. Stopień przynależności elementów do danego zbioru rozmytego opisuje funkcja przynależności (ang. *membership function*) (3.3).

$$\mu_A: U \rightarrow [0,1] \tag{3.3}$$

Przy czym zapis ten oznacza, że funkcja obiektom ze zbioru U przyporządkowuje liczby z przedziału od 0 do 1.

Podsumowując:

- **zmienna lingwistyczna** to zwykle nazwa cechy (wzrost, temperatura, waga, itd.),
- **wartość lingwistyczna** to nazwa kojarzona w sposób intuicyjny z przedziałem wartości (niski, wysoki, zimno, ciepło, itd.)

Należy mieć na uwadze, że stopień przynależności nie ma nic wspólnego z prawdopodobieństwem: osoba raczej niska, której przynależność do zbioru niski wynosi 0.8 to nie to samo, co osoba niska spotykana w 4 na 5 przypadkach (prawdopodobieństwo spotkania 0.8).

3.3 Cechy zbiorów rozmytych

Z kształtem funkcji przynależności związane jest kilka pojęć, opisujących cechy zbioru rozmytego (rys. 3.3).

Nośnikiem zbioru rozmytego (ang. *support*) jest zbiór elementów, których stopień przynależności do danego zbioru jest większy od 0 (3.4):

$$\text{support}(A) = \{ x \in X : \mu_A(x) > 0 \} \tag{3.4}$$

Jądro zbioru rozmytego (ang. *kernel*) to zbiór elementów *x* o przynależności równej 1 (3.5). Gdy tylko jeden element należy do jądra zbioru, to element ten nazywany jest wartością szczytową zbioru, co zachodzi np. dla funkcji o kształcie trójkątnym (czytelnik odpowiedzieć może na pytanie, czy także dla innych funkcji).

$$\ker(A) = \{ x \in X : \mu_A(x) = 1 \} \tag{3.5}$$

α -cięcie (ang. α -cut) zbioru rozmytego A, to zbiór elementów *x* o przynależności większej od zadanego progu α (3.6):

$$A_\alpha = \{ x \in X : \mu_A(x) > \alpha \} \tag{3.6}$$

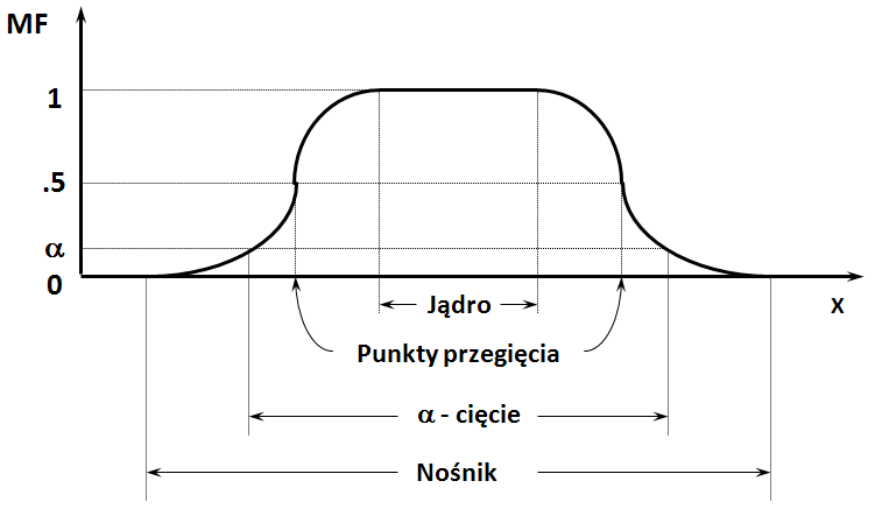
Wysokość zbioru (ang. *height*) to maksymalna osiągana wartość przynależności (zawsze jest ≤ 1) (3.7):

$$hgt(A) = \sup_x \mu_A(x) \tag{3.7}$$

Zbiór rozmyty **normalny** to zbiór, którego wysokość hgt(A) równa jest 1, czyli $\sup_x \mu_A(x) = 1$. Najczęściej w praktyce stosuje się właśnie zbiory normalne, jednak możliwe jest definiowanie i wykorzystywanie funkcji przynależności, których maksimum jest mniejsze od 1. Często w procesie projektowania systemów logiki rozmytej dokonuje się normalizacji zbiorów rozmytych poprzez dzielenie wartości funkcji przynależności danego zbioru przez jego wysokość (3.8).

$$\mu_{An}(x) = \mu_A(x) / hgt(A) = \mu_A(x) / \sup_x \mu_A(x) \tag{3.8}$$

Punktem rozgraniczającym (punktem przegięcia) zbioru rozmytego jest taki element *x* zbioru, dla którego wartość funkcji przynależności $\mu(x)=½$. Zbiór rozmyty może nie posiadać punktu rozgraniczającego lub może posiadać jeden lub wiele punktów rozgraniczających.



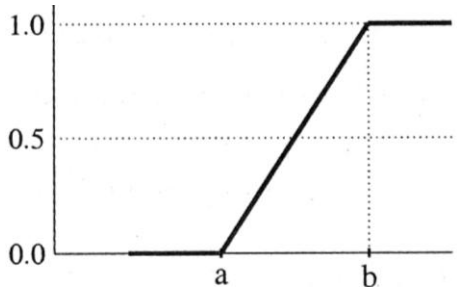
Rys. 3.3. Graficzna reprezentacja cech funkcji przynależności

3.4 Typy funkcji przynależności

Kształt i nośnik funkcji przynależności mogą być określone albo arbitralnie przez eksperta, albo poprzez analizę statystyczną rzeczywistych wartości danego parametru, obserwowanych w eksperymentach [4][5]. W praktycznych zastosowaniach często korzysta się z kilku rodzajów popularnych kształtów funkcji przynależności [6] (funkcje posiadają kilka parametrów, pozwalających określać ich położenia na osi OX i kształt, m.in. nachylenie).

• funkcje klasy Γ (rys. 3.4)(3.8):

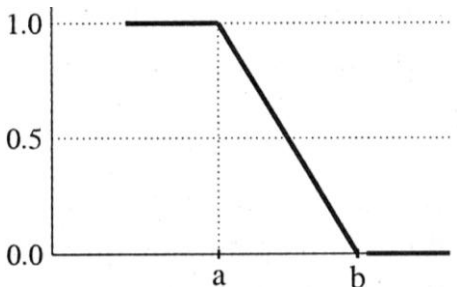
$$\Gamma_{a,b}(x)=\begin{cases}0 & \text{dla } x \le a \\ \dfrac{x-a}{b-a} & \text{dla } a < x \le b \\ 1 & \text{dla } x > b\end{cases} \tag{3.8}$$



Rys. 3.4. Funkcja klasy Γ

• funkcje klasy L (rys. 3.5)(3.9):

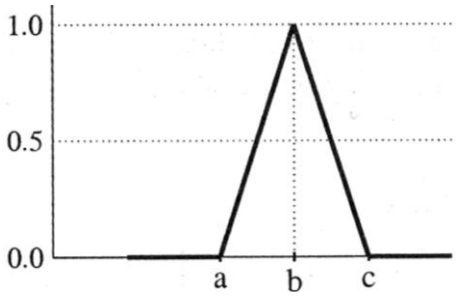
$$L_{a,b}(x)=\begin{cases}0 & \text{dla } x \le a \\ \dfrac{b-x}{b-a} & \text{dla } a < x \le b \\ 1 & \text{dla } x > b\end{cases} \tag{3.9}$$



Rys. 3.5. Funkcja klasy L

• funkcje klasy Λ , zwane również trójkątnymi (rys. 3.6)(3.10):

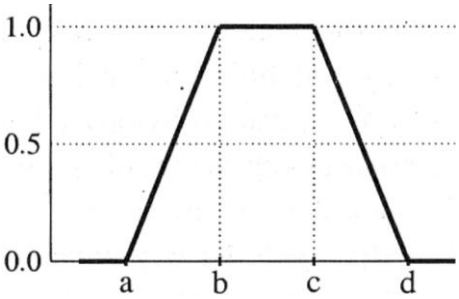
$$\Lambda_{a,b,c}(x)=\begin{cases}0 & \text{dla } x\le a \text{ i } x\ge c\\ \dfrac{x-a}{b-a} & \text{dla } a<x\le b\\ \dfrac{c-x}{c-b} & \text{dla } b<x<c\end{cases} \tag{3.10}$$



Rys. 3.6. Funkcja klasy Λ

• funkcje klasy Π , zwane również trapezowymi (rys. 3.7)(3.11):

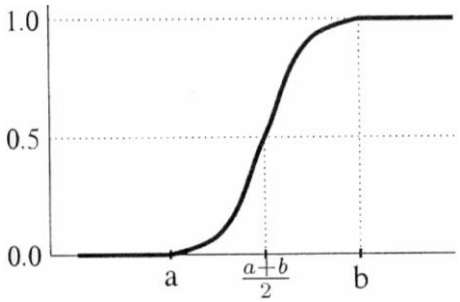
$$\Pi_{a,b,c,d}(x)=\begin{cases}0 & \text{dla } x\le a \text{ i } x\ge d\\ \dfrac{x-a}{b-a} & \text{dla } a<x\le b\\ 1 & \text{dla } b<x\le c\\ \dfrac{d-x}{d-c} & \text{dla } c<x<d\end{cases} \tag{3.11}$$



Rys. 3.7. Funkcja klasy Π

• funkcje klasy s (rys. 3.8)(3.12):

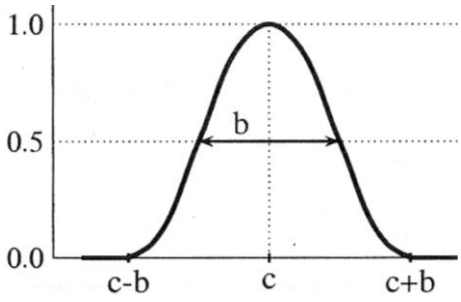
$$s_{a,b}(x)=\begin{cases}0 & \text{dla } x\le a\\ 2\cdot\left(\dfrac{x-a}{b-a}\right)^2 & \text{dla } a<x\le\dfrac{a+b}{2}\\ 1-2\cdot\left(\dfrac{x-b}{b-a}\right)^2 & \text{dla } \dfrac{a+b}{2}<x<b\\ 1 & \text{dla } x\ge b\end{cases} \tag{3.12}$$



Rys. 3.8. Funkcja klasy s

• funkcje klasy π, uzyskane przez dwustronne **złożenie funkcji s** (rys. 3.9)(3.13):

$$\pi_{b,c}(x) = \begin{cases} s_{c-b,c}(x) & \text{dla } x < c \\ 1 - s_{c,c+b}(x) & \text{dla } x \geq c \end{cases} \tag{3.13}$$



Rys. 3.9. Funkcja klasy π

3.5 Podstawowe działania na zbiorach rozmytych

Z punktu widzenia przetwarzania rozmytego najważniejsze są operacje na zbiorach rozmytych, które są analogią do działań na zbiorach (część wspólna, suma, dopełnienie) i do działań logicznych (AND, OR, NOT czyli iloczyn, suma i negacja) (rys. 3.10). Ich znajomość pozwala zaimplementować kompletny proces wnioskowania rozmytego.

- iloczyn zbiorów rozmytych A i B na tym samym uniwersum U to zbiór rozmyty A∩B określony funkcją przynależności (3.14):

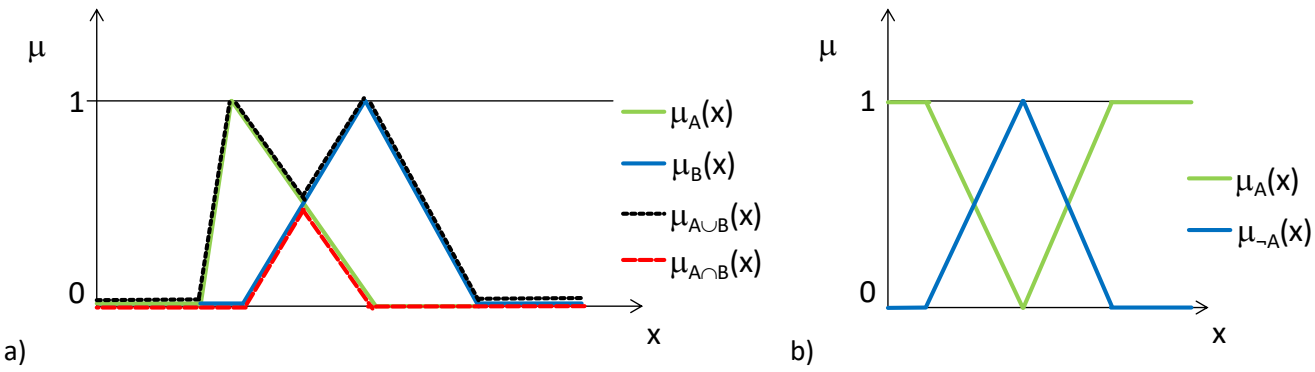
$$\mu_{A \cap B}(x) = min(\mu_A(x), \mu_B(x)) \tag{3.14}$$

- suma zbiorów rozmytych A i B na tym samym uniwersum U to zbiór rozmyty A∪B określony funkcją przynależności (3.15):

$$\mu_{A \cup B}(x) = max(\mu_A(x), \mu_B(x)) \tag{3.15}$$

- dopełnienie zbioru rozmytego A na uniwersum U to zbiór rozmyty ¬A (3.16):

$$\mu_{\neg A}(x) = 1 - \mu_A(x) \tag{3.16}$$

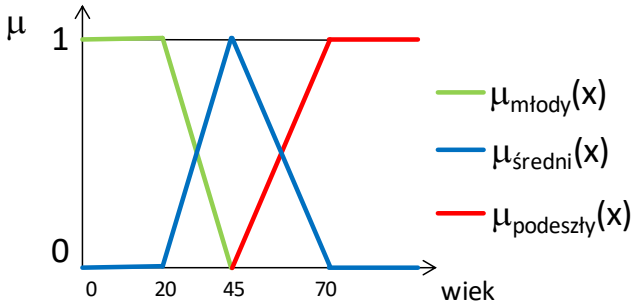


Rys. 3.10. Graficzna reprezentacja działań na zbiorach: a) iloczyn i suma, b) dopełnienie

3.6 Rozmyty opis atrybutu

Rozmyte pojęcia są subiektywne i zależne od kontekstu i natury rozwiązywanego problemu. Dla jednych zagadnień temperatury: 36,6°C jest średnia, a 37,2°C jest wysoka (medycyna), dla innych obie są niskie (np. temperatura wody w gastronomii). Wobec tego rozpoczynając tworzenie modelu rozmytego, należy określić, w jaki sposób dziedziny wartości wejściowych i wyjściowych zostaną podzielone na zbiory rozmyte.

Korzystając z popularnych kształtów funkcji przynależności (rozdział 3.4), oznacza się na osi wartości lingwistyczne każdej zmiennej. Przykładowy zestaw funkcji dla zmiennej „wiek” przedstawiono poniżej:

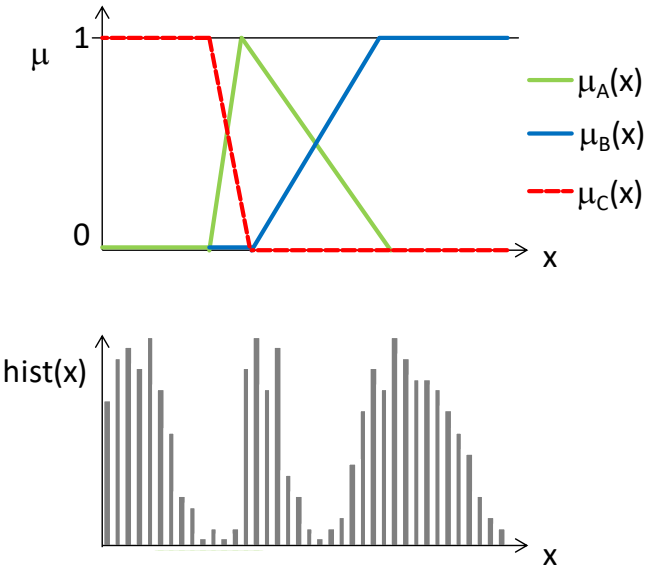


Rys. 3.11. Funkcje przynależności wartości lingwistycznych zmiennej *wiek*

Nośniki i jądra funkcji oraz nachylenie zboczy dobrać należy:

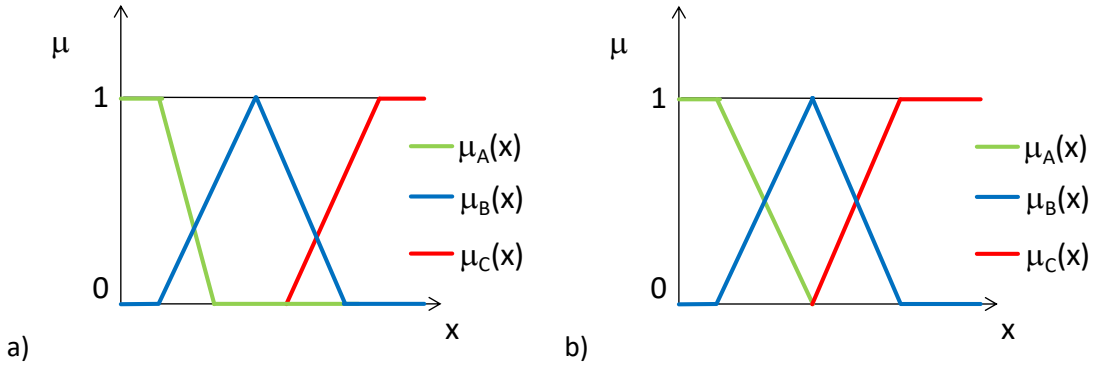
- zgodnie z **intuicją** eksperta, posiłkując się doświadczeniem i znajomością dziedziny. Warto dobierać przedziały rozmyte, które rzeczywiście mają wpływ na decyzję. Jeżeli przykładowo wśród osób powyżej 40 roku życia występuje zwiększone ryzyko raka prostaty, to grupa zdecydowanie poniżej 40, np. 0-30 lat może być opisana jedną funkcją, grupa krytyczna, np. 30-60 drugą funkcją, a grupa starsza, zmniejszonego ryzyka, powyżej 50 roku życia, trzecią funkcją. Wówczas opis rozmyty uprości zastosowanie reguły: „jeżeli wiek pacjenta zbliżony jest do 40 to prowadzić trzeba badanie krwi” (przykładowo).

- **automatycznie**, przydzielając równym przedziałom na osi identyczne kształty funkcji, np. co 5°C trójkątne, zachodzące na siebie funkcje przynależności opisujące temperaturę powietrza. Wówczas zakładamy, że dla sąsiednich przedziałów mogą występować podobne zależności – np. „gdy jest bardzo zimno ubierz kurtkę zimową”, „gdy jest zimno ubierz kurtkę zimową”, „gdy jest gorąco załóż T-shirt”, „gdy jest bardzo gorąco załóż T-shirt”.
- w wyniku **analizy statystycznej** i obserwacji histogramu wartości. Jeżeli bardzo często mierzone/obserwowane wartości skupiają się w łatwym do zdefiniowania przedziale, to należy je opisać jedną funkcją przynależności. Na wykresie histogramu „wysokie słupki” skupione są w podprzedziałach i oddzielone są podprzedziałami o „niskich słupkach” lub pustymi.



Rys. 3.12. Przykładowy histogram i proponowane funkcje przynależności

Bez względu na przyjęty sposób określania kształtu i rozmieszczenia funkcji przynależności wskazane jest zapewnianie warunku **sumowania do jedności przynależności**, dla każdej wartości na osi *x*. Proponuje się taki warunek poprzez analogię do zbioru tradycyjnego, gdzie każdy obiekt należy do możliwych zbiorów określonych na danej dziedzinie, z sumaryczną przynależnością równą 1 (do jednego tylko zbioru należy a do innych nie, co łącznie daje 1+0…+0=1).



Rys. 3.13. Funkcje przynależności dla warunku sumowania do jedności: a) brak spełnienia warunku, b) spełnienie warunku

3.7 Wnioskowanie rozmyte

Wprowadzone powyżej pojęcia pozwalają za pomocą funkcji rozmytych opisać dziedzinę zmienności wybranej zmiennej lingwistycznej. Na tak rozmytych zmiennych przeprowadzone mogą być operacje w logice rozmytej, których wynikiem jest nowa wartość lingwistyczna – wynik reguły logicznej.

Przetwarzanie danych w typowym systemie logiki rozmytej przebiega w następujących krokach:

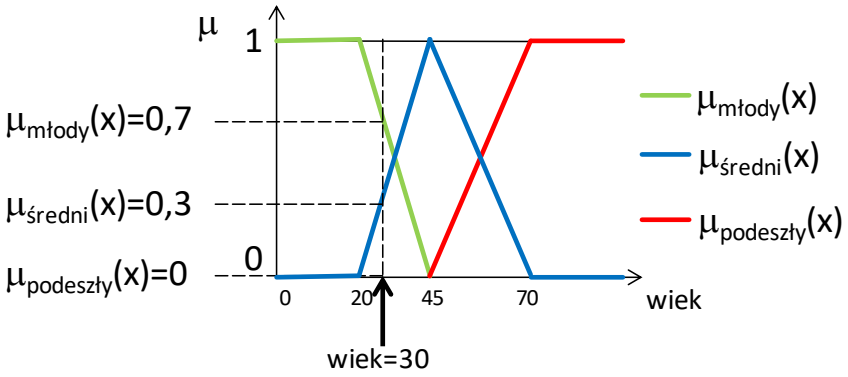
1. przetwarzanie wstępne (ang. *preprocessing*);
2. rozmywanie, fuzyfikacja (ang. *fuzzyfication*);
3. interpretacja reguł (ang. *inferrence*);
4. wyostrzanie (ang. *defuzzyfication*);
5. przetwarzanie końcowe (ang. *postprocessing*).

3.7.1 Przetwarzanie wstępne

Celem **przetwarzania wstępnego** jest m.in. konwersja danych wejściowych do formatu akceptowanego przez system wnioskowania FL. Może polegać na zmianie formatu zapisu, precyzji, na zaokrąglaniu wartości, normalizacji do zadanego przedziału zmienności, zamianie przecinka dziesiętnego na kropkę, itp. System logiki rozmytej oczekuje na wejściu parametrów w postaci liczb rzeczywistych i zwraca wyniki również w postaci liczb rzeczywistych (ang. *crisp value*), które nazywa się „ostrymi”, w przeciwieństwie do rozmytych.

3.7.2 Rozmywanie

Kolejny etap przetwarzania to **rozmywanie** [3][6], które polega na wyznaczeniu wartości lingwistycznych w oparciu o wartości zwracane przez **funkcje przynależności** dla danej zmiennej wejściowej (rys. 3.14).



Rys. 3.14. Graficzna interpretacja rozmywania.
Wejściowa ostra wartość *wiek*=30 zamieniana jest na wartości rozmyte *młody* i *średni*

3.7.3 Interpretacja reguł

Poprzez odpowiednią kombinację wartości lingwistycznych uzyskuje się możliwość wnioskowania – wiedza wyrażona w postaci reguł logicznych może być użyta do określenia wyniku przy zadanych wartościach wejściowych. Reguły w logice tradycyjnej, dwuwartościowej, pochodzić mogą z metod drążenia danych, np. mogą testować rozgałęzienia w drzewie decyzyjnym. Rozmyte reguły skonstruowane są bardzo podobnie do reguł logiki *true-false*, z tym, że testują zwykle więcej niż dwie wartości danej zmiennej lingwistycznej.

Typowa reguła w logice rozmytej ma postać wyrażenia złożonego z poprzednia (przesłanek reguły) i następnika (decyzji) (3.12). Typowo przesłanki połączone są warunkami AND, jednak możliwe jest stosowanie OR oraz NOT (zależnie od typu systemu FL).

IF przesłanka 1 **AND** przesłanka 2 **AND** ... **AND** przesłanka n **THEN** decyzja (3.12)

Reguł o takiej postaci jest dla danego zagadnienia wiele. Zwykle powinny one testować wszystkie możliwe kombinacje wartości lingwistycznych tak, aby system mógł działać w sposób deterministyczny dla danych analizowanych w przyszłości. Jeżeli temperatura ciała przyjąć może 3 różne wartości lingwistyczne, a nasilenie kaszlu 2 wartości, to tworzone jest 2x3=6 reguł, uwzględniających wszystkie kombinacje i determinujących decyzję (chory / zdrowy lub podobną), w każdym z możliwych przypadków.

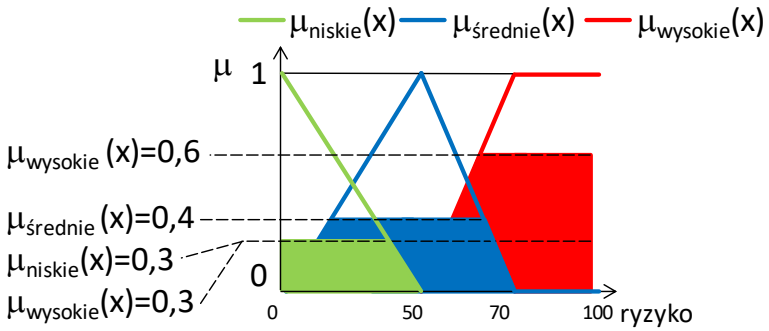
Interpretacja reguł przebiega w dwóch fazach. Najpierw oblicza się **moc reguły** (ang. *rule evaluation*), czyli określa jak silna jest decyzja uzyskana przez obliczenie reguły dla danych wartości wejściowych. W tym celu w miejsce przesłanek podstawia się wartości odpowiadających im zmiennych lingwistycznych. Ponieważ w logice rozmytej operacja AND równoważna jest funkcji minimum, dlatego moc reguły oblicza się jako minimum wartości przesłanek, występujących w tej regule. Jeżeli moc reguły jest zerowa, to reguła ta jest uznawana za nieaktywną.

Wyznaczona moc reguły interpretowana jest jako stopień przynależności wynikowej wartości decyzji (rozmytej lingwistycznej wartości z dziedziny decyzji)

Po wyznaczeniu mocy wszystkich reguł występujących w systemie FL następuje faza **agregacji reguł** (ang. *rule aggregation*), która polega na sumowaniu wszystkich wynikowych zbiorów rozmytych, reprezentujących poszczególne reguły [6][8]. Decyzje z wielu reguł tworzą zdanie logiczne o postaci:

Decyzja = dec 1 **OR** dec 2 **OR** … **OR** dec n (3.13)

Wobec tego, że zdanie to zawiera alternatywy, czyli warunki OR, to do wyliczenia końcowej decyzji stosuje się funkcję maksimum (sumę logiczną).



Rys. 3.15. Graficzna interpretacja procesu agregacji reguł – wynikiem jest pole powierzchni pod trzema kolorowymi trapezami

3.7.4 Wyostrzanie

Dla wynikowego zbioru rozmytego przeprowadza się wyostrzanie (defuzyfikację). Jest to operacja odwrotna do rozmywania, której zadaniem jest zamiana rozmytego wyniku na liczbę rzeczywistą, ostrą. Wyostrzanie uwzględnia kształt funkcji przynależności wynikowego zbioru decyzji rozmytej. Wobec tego, do określenia na postawie kształtu (pola, krzywizny) jednej liczby rzeczywistej, stosuje się podejścia analizujące funkcję lub pole pod funkcją i zwracające jedną wartość ostrą.

Wyostrzanie można przeprowadzić na kilka sposobów:

• Metoda środka przedziału o największej wartości funkcji przynależności (ang. *mean of maximum*).

Wynikiem tego typu wyostrzania jest wartość punktu x_0, który jest środkiem przedziału, w którym wyznaczona funkcja przynależności przyjmuje maksymalną wartość (rys. 3.16a). Jest to najprostszy sposób defuzyfikacji, sprowadza się on do wyboru tej reguły, której moc była największa. Wadą tego rozwiązania jest nieuwzględnianie pozostałych reguł. Ponadto w trakcie wolnej zmiany wartości zmiennych wejściowych, gdy reguły aktywowane są z różnymi stopniami dochodzi do sytuacji zmiany z jednej dominującej funkcji przynależności na inną, czyli nieciągłego przeskoku wynikowej wartości wyostrzonej. Gdy wystąpi wiele reguł aktywowanych z równą, maksymalną wartością aktywacji, to jako wynik przyjmuje się położenie x_0 najbardziej z lewej strony (wartość najmniejszą).

• Metoda centrowego środka ciężkości (ang. *center average*).

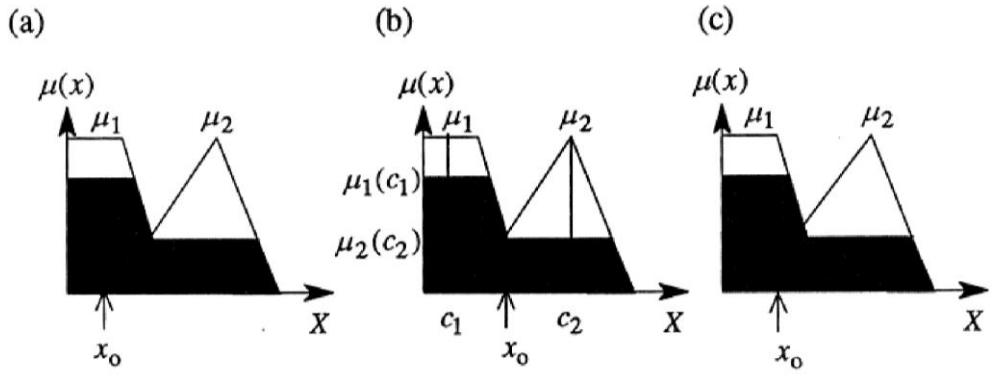
Wartość wyjściową x_0 oblicza się według zależności (3.14) (rys. 3.16b). Wartość c_i określa środek przedziału x, dla którego i-ta funkcja przynależności $\mu(x)$ przyjmuje wartość maksymalną. Parametr N określa liczbę wszystkich wyjściowych zbiorów rozmytych. W metodzie tej brane są pod uwagę wszystkie aktywowane reguły. Należy tu dostrzec podobieństwo do sposobu wyliczania średniej ważonej. Wadą tej metody jest nieuwzględnianie informacji o kształcie funkcji przynależności, a bazuje ona tylko na położeniu środków przedziałów.

$$x_0 = \frac{\sum_{i=1}^{N} c_i \cdot \mu_i(c_i)}{\sum_{i=1}^{N} \mu_i(c_i)} \tag{3.14}$$

• metoda wyznaczania środka ciężkości (ang. *center of gravity*).

Wartość wyjściową x_0 oblicza się według zależności (3.15) (rys. 3.15c). Metoda ta jest najbardziej uniwersalna, gdyż uwzględnia wszystkie aktywne reguły oraz kształt funkcji przynależności. Jej wadą jest większa złożoność obliczeniowa, wynikająca z konieczności wyliczenia całek (pól powierzchni pod krzywymi funkcji).

$$x_0 = \frac{\int_X \mu(x) \cdot x dx}{\int_X \mu(x) dx} \tag{3.15}$$



Rys. 3.16. Graficzna interpretacja metod wyostrzania: a) metoda środka przedziału o największej wartości funkcji przynależności, b) metoda centrowego środka ciężkości, c) metoda wyznaczania środka ciężkości [2]

3.7.5 Przetwarzanie końcowe

Przetwarzanie końcowe, analogicznie do wstępnego, konwertuje wyniki systemu logiki rozmytej do formatu akceptowanego przez zewnętrzne moduły podłączone do tego systemu. Tu odbywać może się normalizacja, uśrednianie w czasie wyników za okres kilku próbek wstecz lub podobne operacje.

3.8 Prawdopodobieństwo a przynależność

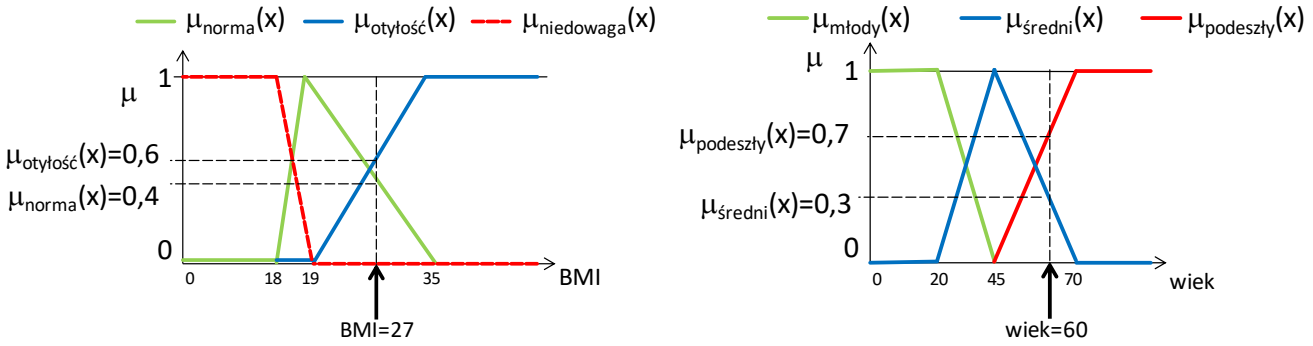
Teoria prawdopodobieństwa i teoria zbiorów rozmytych mogą wydawać się podobne do siebie, gdyż przyjmują wartości określane liczbami rzeczywistymi z domkniętego przedziału [0;1].

Prawdopodobieństwo jest związane z faktami zachodzenia zdarzeń, a niepewność, wyrażana miarą prawdopodobieństwa, dotyczy przypadkowości pojawiania się tych zdarzeń. Natomiast **rozmytość** to niepewność związana z określeniem przynależności danego elementu do zbioru rozmytego.

Prawdopodobieństwo jest miarą, która spełnia warunek addytywności (prawdopodobieństwo zdarzenia, które jest sumą rozłącznych zdarzeń równe jest sumie prawdopodobieństw tych zdarzeń). Prawdopodobieństwo jest także unormowane, co oznacza, że suma prawdopodobieństw dla wszystkich możliwości zawsze wynosi 1. Rozmytość nie musi spełniać ani warunku addytywności ani unormowania [1][4][6].

3.9 Przykładowe zadanie określania ryzyka zawału

Poniżej opisane jest hipotetyczne zagadnienie określania ryzyka zawału na podstawie wieku i indeksu masy ciała (BMI – *body mass index*). Dwa wejściowe parametry mają zdefiniowane po trzy funkcje przynależności, trójkątne, z sumowaniem do jedności (rys. 3.17). Wiek podany liczbą rozmywany jest to lingwistycznych wartości: *młody, średni, podeszły*, a BMI do wartości *niedowaga, norma* i *otyłość.* Przykładowy pacjent ma 60 lat i BMI=27.



Rys. 3.17. Funkcje przynależności dla BMI i wieku.
Wyniki rozmywania wartości BMI=27 i wiek=60

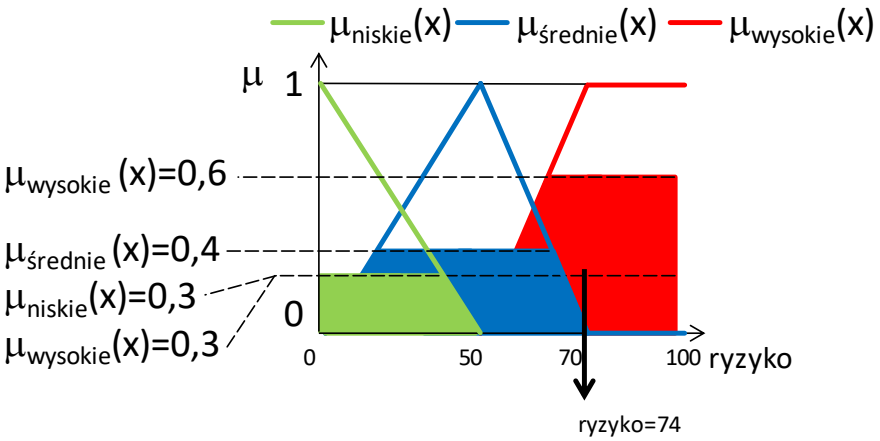
Stworzono zbiór reguł uwzględniających wszystkie 3·3=9 przypadków (tab. 3.1). Zauważyć należy, że w zbiorze reguł do analizowanego przypadku pasują cztery reguły, których stopnie aktywacji określa się poprzez wyliczanie funkcji iloczynu (AND) argumentów w przesłance. Przykładowa reguła: IF *wiek=średni* AND *tusza=norma* THEN *ryzyko=niskie* po uwzględnieniu przynależności (rys. 3.17) przyjmie postać: IF *średni*(0,3) AND *norma*(0,4) THEN *niskie*(0,3). Wartość aktywacji stopnia reguły *niskie*(0,3) wynika z wyliczenia iloczynu (3.14) czyli min(0,3; 0,4)=0,3. W ten sposób wyliczone zostały stopnie aktywacji dla wszystkich czterech aktywowanych reguł.

Tab. 3.1. Reguły wyrażające zależność ryzyka zawału od tuszy i wieku. Podano stopnie aktywacji dla przykładowego pacjenta

Ryzyko zawału		Tusza		
		niedowaga	norma (0,4)	otyłość (0,6)
Wiek	młody	niskie	niskie	średnie
	średni (0,3)	niskie	niskie (0,3)	wysokie (0,3)
	podeszły (0,7)	niskie	średnie (0,4)	wysokie (0,6)

Wynik wyliczany jest poprzez zagregowane wszystkich aktywowanych reguł. Zapisać można to następująco: ryzyko = niskie(0,3) OR średnie(0,4) OR wysokie(0,3) OR wysokie(0,6). Zgodnie z opisanymi metodami agregacji (rozdział 3.7.4) dokonuje się w sposób graficzny:

1. Obcięcia funkcji przynależności na poziomie odpowiadającym stopniowi aktywacji (rys. 3.18).
2. Wyliczenia np. środka ciężkości figury powstałej przez zsumowanie powierzchni trapezów.



Rys. 3.18. Wynik aktywacji reguł, wyostrzanie metodą środka ciężkości

3.10 Literatura

[1] Czogała E., Pedrycz W., *Elementy i metody teorii zbiorów rozmytych*, PWN, Warszawa 1982.

[2] Czyżewski A., *Dźwięk cyfrowy. Podstawy teoretyczne, technologia, zastosowania*, Akademicka Oficyna Wydawnicza, Warszawa, 1998.

[3] Driankov D., Hellendoom H., Reinfrank M., *Wprowadzenie do sterowania rozmytego*, WNT, Warszawa, 1996.

[4] Kosko B., *Fuzzy Engineering*, Prentice-Hall, 1997.

[5] Kostek B., *Soft Computing in Acoustics, Applications of Neural Networks*, Fuzzy Logic and Rough Sets to Musical Acoustics, Studies in Fuzziness and Soft Computing, Physica Verlag, Heilderberg, New York, 1999.

[6] Łachwa A., *Rozmyty świat zbiorów, liczb, relacji, faktów, reguł i decyzji*, Akademicka Oficyna Wydawnicza, Warszawa, 2001.

[7] Mendel J.M., *Fuzzy Logic Systems for Engineering: A Tutorial*, IEEE, 1995.

[8] Zadeh L.A., *Fuzzy Sets*, Information and control, pp. 338-353, 1965.

[9] Zadeh L. A., *Fuzzy logic = computing with words*, IEEE Trans. on Fuzzy Systems, vol. 4, pp. 103-111, 1996.

4 Drzewa decyzyjne

4.1 Klasyfikacja danych

Drążenie danych (ang. *data mining*), to dziedzina skupiająca się na analizie danych w celu wydobywania wiedzy o zależnościach między cechami obiektów będących w obszarze zainteresowania, którego dotyczy wybrana aplikacja. Zwykle obiektom przypisywana jest decyzja w postaci atrybutu symbolicznego, opisu słownego (rzadziej liczby rzeczywistej). Decyzja pochodzić może z badań statystycznych, z eksperymentów, od ekspertów, którzy badają rzeczywiste przypadki i na podstawie doświadczenia odpowiednio je opisują (np. przypadki chorobowe, prognozę wyzdrowienia, typ schorzenia, itd.). Zadaniem narzędzi klasyfikujących jest zastąpić takie metody wydawania decyzji, w ich miejsce stosując automatyczne pomiary obiektywnych cech, atrybutów, np. parametrów biologicznych pacjenta, a następnie wykorzystać reguły określające zależność między cechami a decyzją wynikową.

Klasyfikacja polega więc na przewidywaniu wyniku na podstawie posiadanych parametrów opisujących obiekt (w domyśle nie wszystkich możliwych, tylko tych, których pomiar jest możliwy, prosty, uzasadniony ekonomicznie, nieinwazyjny).

4.2 Tablice kontyngencji

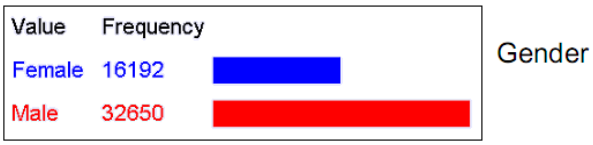
Wspominane dane i parametry obiektów mogą mieć różną postać: atrybutów słownych, liczbowych, odnośników, itd. Interesujące jest obserwowanie, które z atrybutów występują w przypadku różnych decyzji, co może pomóc wnioskować na temat zależności.

W celu obserwacji współzależności stosować można tzw. tablice kontyngencji (współwystępowania) cech. Za przykład niech posłuży badanie statystyczne zamożności społeczeństwa amerykańskiego. Zebrane dane to informacja o 48000 osób, opisanych 16 atrybutami [Koh95]. Atrybutem decyzyjnym, którego zależność od innych atrybutów chcemy poznać jest stan majątkowy (ang. *wealth*), określony dwuwartościowo „*poor*” i „*rich*” (ubogi i bogaty) (tab. 4.1).

Tab. 4.1. Przykład systemu decyzyjnego opisującego cechy społeczeństwa amerykańskiego

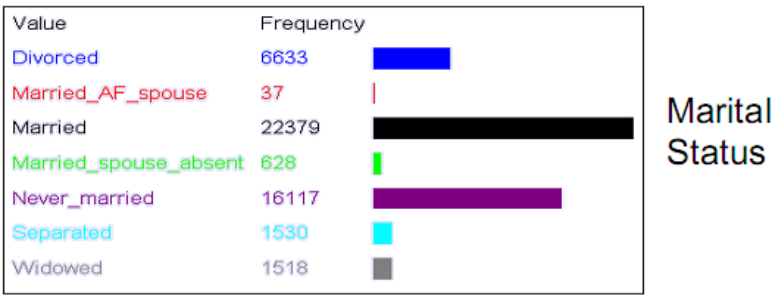
age	employment	education	edu_yrs	marital	…	job	relation	race	gender	work_h	wealth
39	State_gov	Bachelors	13	Never_mar	…	Adm_cleric	Not_in_fam	White	Male	40	poor
51	Self_emp_	Bachelors	13	Married	…	Exec_man	Husband	White	Male	13	poor
39	Private	HS_grad	9	Divorced	…	Handlers_c	Not_in_fam	White	Male	40	poor
54	Private	11th	7	Married	…	Handlers_c	Husband	Black	Male	40	poor
…											
52	Self_emp_	HS_grad	9	Married	…	Exec_man	Husband	White	Male	45	rich
31	Private	Masters	14	Never_mar	…	Prof_speci	Not_in_fam	White	Female	50	rich

Analiza danych tylko poprzez przeglądanie takiej tabeli nie jest możliwa. W tym celu najprościej jest na początek sprawdzić i na wykresach **histogramów** przedstawić liczbę obiektów cechujących się jakąś określoną kombinacją wartości cech i odpowiednią decyzją. Histogram to jednowymiarowa tablica kontyngencji, która przykładowo pokazać może liczbę kobiet i liczbę mężczyzn w całej badanej grupie (rys. 4.1):



Rys. 4.1. Tablica kontyngencji dla jednej cechy – liczba osób danej płci w badanej grupie

Inna wizualizowana cecha – stan cywilny (rys. 4.2):



Rys. 4.2. Tablica kontyngencji dla jednej cechy – liczba osób o określonym stanie cywilnym

Słupki histogramu graficznie reprezentują liczność tych grup, pozwalając wyciągać intuicyjnie pewne wnioski, np. o nierówności płci w tym badaniu czy o rzadkości przypadków separacji i wdowieństwa.

Jednakże dopiero porównanie liczności niezależnie dla osób ubogich i bogatych dostarczy interesujących danych. W tym celu wprowadźmy precyzyjnie pojęcie **tablicy kontyngencji**.

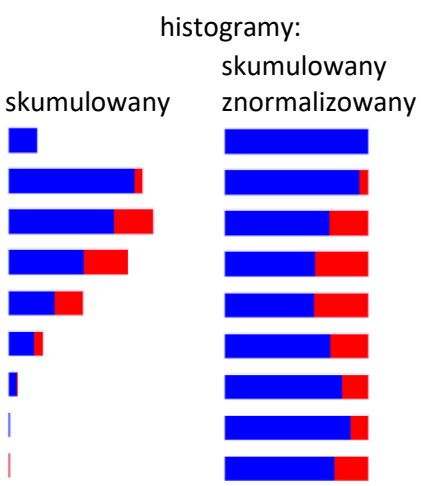
Kontyngencja jest to współzależność statystyczna między cechami, z których przynajmniej jedna jest cechą jakościową.

Z kolei histogram jest to jednowymiarowa tablica kontyngencji. K-wymiarowa tablica zbudowana może być w następujący sposób:

- wybierz *k* interesujących atrybutów, a_1, a_2, … a_K,
- dla każdej możliwej kombinacji wartości $a_1=x_1$, $a_2=x_2$, … $a_K=x_K$, określ ile razy pojawia się w zbiorze danych obiekt o takich wartościach.

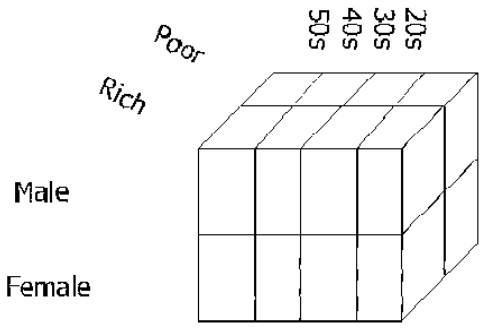
Przykładowo, dla każdej możliwej kombinacji wartości wiek (*age*) i zamożność (*wealth*) określana jest liczba obiektów (rys. 4.3). Ponadto graficznie przedstawić można liczności obiektów (histogramy skumulowane), które intuicyjne w szybki sposób pozwalają stwierdzić, że grupa 30-latków jest najliczniejsza, że wśród 30- i 40-latków osób zamożnych jest dużo. Wykreślić można także liczności znormalizowane do 100% (histogramy skumulowane 100%), które ujawniają, że w grupach 40- i 50-latków procentowo osób zamożnych jest tyle samo, choć grupy te nie są równoliczne i wśród 90-latków osób bogatych jest także stosunkowo dużo, czego nie widać było na wykresie histogramu.

age	wealth: poor	wealth: rich
10s	2507	3
20s	11262	743
30s	9468	3461
40s	6738	3986
50s	4110	2509
60s	2245	809
70s	668	147
80s	115	16
90s	42	13



Rys. 4.3. Tablica kontyngencji dla dwóch cech: przedział wiekowy i zamożność

Tablice kontyngencji są dobrym narzędziem szybko dostarczającym informacji o badanych obiektach. Jednakże podczas badania cech, które mają wiele różnych możliwości i badania więcej niż 2 cech jednocześnie, wizualizowanie i wykorzystanie takich tablic staje się trudne. Trójwymiarową tablicę można interpretować jako kostkę podzieloną na mniejsze kostki, w których zapisane są liczności grup, ale posługiwanie się taką tabelą są niewygodne (rys. 4.4).



Rys. 4.4. Sposób graficznej reprezentacji tablicy kontyngencji dla trzech cech

Do wykreślania histogramów i tablic kontyngencji często korzysta się z oprogramowania typu OLAP (*on-line analytical processing*), które posiadają kreatory tablic, podglądy przekrojów, histogramy, itp. Popularnymi i łatwo dostępnymi przykładami są wykresy w programie MS Excel: histogram, skumulowany i 100-procentowy skumulowany, nadające się do przedstawiania tablic dwuwymiarowych.

W przykładzie ankietowania amerykańskiego społeczeństwa, każda osoba opisana jest 16 atrybutami. Możliwych do wykreślenia tablic 1-wymiarowych jest 16, 2-wymiarowych jest $16 \cdot 15/2=120$, itd. Różnych tablic 3 wymiarowych dla danych, w których byłoby 100 atrybutów będzie ponad 160 tysięcy. Ich wyliczenie nie jest problemem, jednak konieczne jest posiadanie narzędzia, które pozwoli stwierdzić, która spośród nich może przedstawiać **istotne zależności** między atrybutami a decyzją. Tym właśnie zajmuje się drążenie danych – odpowiedzią na pytanie, które zależności są interesujące i przydatne dla prognozowania i wspomagania decyzji, a które są tylko pozornie znaczące oraz jak je wykorzystać.

4.3 Istotność informacji – entropia

Stosując teorię informacji Shannona, uzyskać można bardzo istotną miarę jakości informacji zawartej w zbiorze danych lub w przetworzonym jej wycinku, jakim jest tablica kontyngencji. W ten sposób można będzie stwierdzić, które zależności i tablice są nienadmiarowe, uporządkowane lub nie, istotnie lub nie.

Entropia jest to najmniejsza średnia ilość jednostek informacji (bitów) potrzebna do zakodowania faktu zajścia pewnego zdarzenia (ze zbioru zdarzeń o danych prawdopodobieństwach).

Dla przypadku **równych prawdopodobieństw**, kiedy każde ze zdarzeń zachodzi tak samo często, **entropia jest największa** i obserwując ich ciąg trudno jest przewidzieć, które zajdzie jako kolejne. Niech X przyjmuje cztery różne wartości z prawdopodobieństwami:

$$P(X=A) = 1/4$$

$$P(X=B) = 1/4$$

$$P(X=C) = 1/4$$

$$P(X=D) = 1/4$$

Przykładowy rejestrowany (zapisywany, przesyłany, obserwowany) ciąg zdarzeń zakodowany może być binarnie w taki sposób, że:

$$A = 00,\ B = 01,\ C = 10,\ D = 11$$

Wystąpienie ośmiu zdarzeń, np. ABBDCADC daje ciąg 16 bitów: 0001011110001110. Bardzo długa obserwacja zdarzeń zawsze wykaże, że średnio jedno zdarzenie zapisane jest na dwóch bitach. Nie ma możliwości użycia mniejszej liczby bitów i entropia wynosi dokładnie 2.

Dla przypadku **różnych prawdopodobieństw**, są zdarzenia następujące częściej, tj. takie, których można się spodziewać lub je przewidzieć i wówczas **entropia jest mniejsza** niż w powyższym przypadku. Niech prawdopodobieństwa wynoszą:

P(X=A) = 1/2

P(X=B) = 1/4

P(X=C) = 1/8

P(X=D) = 1/8

Okazuje się, że poprzez odpowiednie kodowanie binarne zdarzeń uzyskać można sytuację, że średnio potrzebne będzie tylko 1.75 bita na zdarzenie! Jedno z możliwych kodowań dających taki wynik jest następujące:

A = 0, B = 10, C = 110, D = 111

Można zauważyć, że krótki ciąg bajtów używa się do kodowania informacji najczęstszej, a dłuższy dla rzadko występującej.

Przypadek ogólny

Zmienna losowa X przyjmuje *m* różnych wartości V_1 , V_2 , ..., V_m z prawdopodobieństwami równymi p_1, p , ..., p_m .

Najmniejsza możliwa średnia liczba bitów na symbol, potrzebna do przetransmitowania ciągu symboli o dystrybucji losowej X wyraża się wzorem:

$$H(X) = – p_1 \cdot \log_2 p_1 – p_2 \cdot \log_2 p_2 \ldots – p_m \cdot \log_2 p_m = – \sum_{j=1\ldots m} p_j \log_2 p_j \qquad (4.1)$$

H(X) to entropia X.

Czytelnik może **samodzielnie policzyć entropię** dla powyższych przypadków czterech zdarzeń o równych i różnych prawdopodobieństwach. Dla przypomnienia, $\log_2$ z liczby 2^{-1} to –1, $\log_2(2^{-2})$=–2 i ogólnie $\log_2(2^n)$=n, a 1/2 to 2^{-1}, 1/4 to 2^{-2}, itd., To zadanie można rozwiązać bez kalkulatora.

Interpretacja entropii

Duża entropia oznacza równomierną dystrybucję, płaski histogram, rozrzucenie wartości w całej dziedzinie wartości atrybutu, trudniejsze do przewidzenia zdarzenia. Mała entropia oznacza nierównomierną dystrybucję, pofalowany histogram, skupiska i luki w przedziale wartości, łatwiejsze do przewidzenia zdarzenia.

4.4 Entropia warunkowa

Tablica kontyngencji reprezentuje liczbę obiektów o danej decyzji w grupach o danych cechach. Potrzebne jest pomierzenie przydatności informacji zawartych w takich tablicach, czyli tego, **jak**

znajomość cechy wpływa na łatwość przewidzenia decyzji. W tym celu zostanie wykorzystana **entropia warunkowa**.

Dla przykładu można rozpatrzyć zbiór ośmiu osób studiujących matematykę, informatykę lub historię, które z powodu długotrwałego używania komputerowej myszy mają symptomy zespołu cieśni nadgarstka (c.n.) (tab. 4.2):

Tab. 4.2. Przykładowe dane prezentujące związek między specjalizacją a możliwym występowaniem zespołu cieśni nadgarstka

X (kierunek studiów)	Y (możliwy zespół c.n.)
Mat.	Tak
Hist.	Nie
Inf.	Tak
Mat.	Nie
Mat.	Nie
Inf.	Tak
Hist.	Nie
Mat.	Tak

W takim przypadku określić można prawdopodobieństwa:

– P(Y = Tak) = 0.5 (połowa osób skarży się na ból i osłabienie chwytu)

– P(X = Mat. && Y = Nie) = 0.25 (1/4 z wszystkich osób jest matematykami bez symptomów)

– P(X = Mat.) = 0.5 (połowa osób jest matematykami)

– P(Y = Tak | X = Hist.) = 0 (prawdopodobieństwo, że osoba ma zespół c.n. pod warunkiem, że jest historykiem jest zerowe)

Dalej:

H(X) = 1.5

H(Y) = 1 (równe prawdopodobieństwa diagnozy „Tak” i „Nie”, tylko dwa zdarzenia, co wyrażane jest dokładnie 1 bitem)

Postawione zostaje zadanie: **przewidywanie wartości wyjściowej Y pod warunkiem znajomości wartości wejściowej X**.

Entropia warunkowa Y pod warunkiem, że X=v, zostaje zdefiniowana jako H(Y|X=v), tj. entropia liczona po tych tylko Y, dla których X=v.

Czytelnik może samodzielne wyliczyć, ile wynoszą H(Y|X=Mat.), H(Y|X=Inf.), H(Y|X=Hist.).

Z kolei średnia entropia warunkowa Y|X to H(Y|X) (4.2). Odpowiada ona na pytanie, jaka będzie entropia warunkowa Y-ka, jeżeli wybierze się wiersz ze zbioru danych losowo i odczyta się (pomierzy) w tym wierszu wartość X (średnio, bez zakładania do obliczeń konkretnej wartości).

$$H(Y|X) = \sum_j P(X=v_j)\cdot H(Y|X = v_j) \qquad (4.2)$$

W telekomunikacji H(Y|X) interpretowane jest jako oczekiwana liczba bitów wymaganych do przesłania faktu Y, pod warunkiem, że nadawca i odbiorca znają X. Dla przykładu studentów trzech kierunków wyliczone zostają następujące entropie:

Tab. 4.3. Wartości entropii warunkowych przykładu zespołu c.n.

v_j	P (X=v_j)	H(Y|X = v_j)
Mat	0,5	1
Hist.	0,25	0
Inf.	0,25	0

Po podstawieniu do wzoru (4.2) otrzymuje się (4.3):

$$H(Y|X) = 0{,}5\cdot 1 + 0{,}25\cdot 0 + 0{,}25\cdot 0 = 0{,}5 \quad (4.3)$$

W tym przypadku średnia entropia warunkowa jest mniejsza od entropii zwykłej. W dalszej części wyjaśnione zostaną korzyści tego faktu.

4.5 Zysk informacyjny

Można zadać pytanie czy znajomość kierunku studiowania w powyższym przykładzie jest przydatna w preferencji przewidywaniu diagnozy dla tych osób? W przypadku, gdy kierunek nie jest znany, należy posługiwać się miarą entropii H(Y), która miała wartość 1 (równe prawdopodobieństwa „Tak” i „Nie”, tylko dwa zdarzenia, co wyrażane jest dokładnie 1 bitem). Z kolei znajomość X obniża entropię do wartości 0,5, dając zysk równy różnicy H(Y) – H(Y|X) = 1 – 0,5 = 0,5. Im większa jest ta różnica, tym więcej zaoszczędzonych bitów i tym większa wartość nazywana **zyskiem informacyjnym IG** (ang. *information gain*).

Wartość IG(Y|X) mówi o tym, ile oszczędza się bitów średnio w przypadku przesłania Y, jeżeli obie strony znać będą X (informację w jakiś sposób powiązaną z Y) (4.4).

$$IG(Y|X) = H(Y) - H(Y|X) \quad (4.4)$$

W podobny sposób analizować można IG dla zbioru danych przedstawiających społeczeństwo amerykańskie i wartości IG(wealth|X) dla kilku atrybutów X.

H(wealth) = 0,793844

H(wealth|gender) = 0,757154

IG(wealth|gender) = H(wealth) – H(wealth|gender) = 0,03669

H(wealth|age) = 0,709463

IG(wealth|age) = H(wealth) – H(wealth|age) = 0,084381

Znajomość atrybutu *age* daje większy zysk IG. Atrybut ten jest przydatniejszy od *gender* w zadaniu przewidywania zamożności. Analizując IG wszystkich atrybutów można oczywiście uszeregować je w kolejności malejącej, od najbardziej przydatnych do najmniej przydatnych.

Wiedza o przydatności atrybutu do przewidywania decyzji jest podstawą budowania algorytmu decyzyjnego nazywanego **drzewem decyzyjnym**.

4.6 Budowanie drzewa decyzyjnego

Drzewo decyzyjne ma postać odwróconego drzewa, które u góry posiada korzeń i, idąc w dół, w każdym kroku rozdziela się na warunki dotyczące wartości wybranego atrybutu. Na końcu na dole drzewa znajdują się decyzje – liście. Obiekt o danych wartościach parametrów jest testowany przez drzewo, tj. kolejno przechodzi przez rozgałęzienia, kierowany jest jedną ze ścieżek od korzenia do jednego z liści.

W zależności od przyjętej kolejności testowania atrybutów różne drzewa, realizujące te same zadanie klasyfikacji, mogą mieć różną wielkość i „rozpiętość”. Aby określić, które z atrybutów najkorzystniej jest testować, należy wybrać atrybut do tej pory nieprzetestowany o najwyższej wartości IG, wprowadzić do drzewa rozgałęzienie na wartości, które przyjmuje ten atrybut. Kolejne rozgałęzienia wynikają z powtórzenia tego procesu – wyboru następnego atrybutu, jeszcze nieprzetestowanego, o największym IG.

Aby zademonstrować wynik tworzenia drzewa metodą doboru atrybutów według ich wartości IG, można się posłużyć przykładem danych opisujących ryzyko zawału (osoba zdrowa „*nie*” i potencjalnie chora „*tak*”). Cały zbiór danych treningowych zawiera 40 obiektów opisanych 7 atrybutami (tab. 4.4) Atrybuty mają wartości liczbowe lub symboliczne (etykiety słowne, wartości lingwistyczne, nazwy przedziałów). Dla kolejnych atrybutów przedstawione są tablice kontyngencji i zyski informacyjne (tab. 4.5). Wiek osoby, ciśnienie krwi, cholesterol – to trzy pierwsze atrybuty w drzewie decyzyjnym (można możliwe jest posortowanie atrybutów i sprawdzenie jak ich kolejność wykorzystana jest w drzewie).

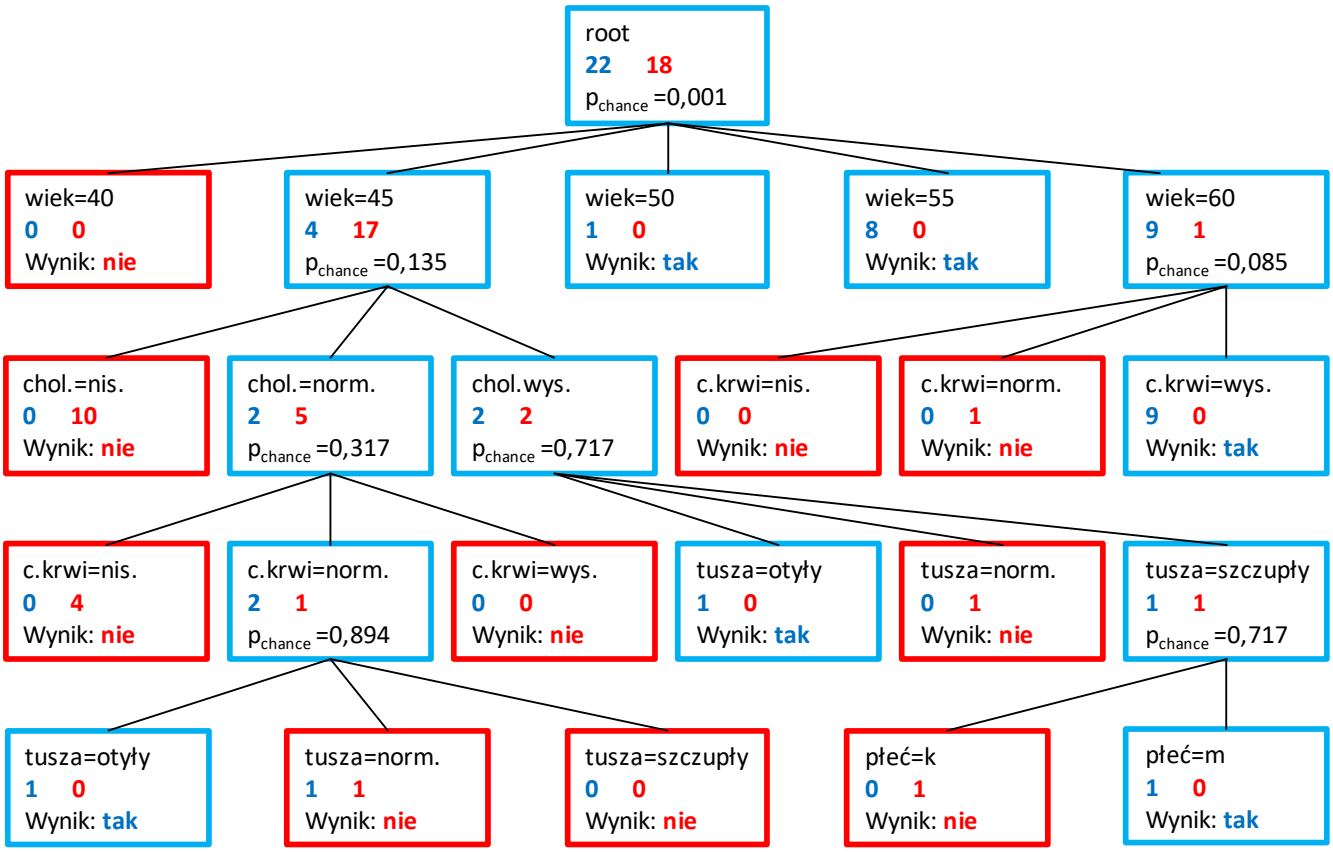
Tab. 4.4. System decyzyjny opisujący ryzyko zawału dla 40 osób

wiek	palenie	c.krwi	cukrzyca	tusza	płeć	cholesterol	ryzyko zawału
40	Nie	Nis.	Nie	Szczupły	K	Norm.	Nie
45	Tak	Norm.	Nie	Norm.	K	Niski	Nie
50	Tak	Wys.	Tak	Norm.	M	Wys.	Tak
55	Nie	Wys.	Nie	Otyły	M	Wys.	Tak
60	Nie	Norm.	Nie	Norm.	K	Norm.	Nie
…	…	…	…	…	…	…	…

Tab. 4.5. Charakterystyka zbioru danych opisujących parametry pojazdów

Atrybut	Wartość	Dystrybucja	IG1
wiek	40		0,506731
	45		
	50		
	55		
	60		
palenie	tak		0,223144
	nie		
	w przeszłości		
c.krwi	niskie		0,387605
	normalne		
	wysokie		
cukrzyca	tak		0,064018
	nie		
tusza	szczuply		0,342088
	normalny		
	otyły		
płeć	M		0,267964
	K		
cholesterol	niski		0,437265
	normalny		
	wysoki		

Uzyskane zostało drzewo decyzyjne, które testuje wszystkie atrybuty zgodnie z kolejnością malejącego IG (rys. 4.5).



Rys. 4.5. Drzewo decyzyjne uwzględniające wszystkie atrybuty

Zauważyć należy następujące elementy uzyskanego drzewa:

- w każdym węźle drzewa zapisano dwoma kolorami liczbę obiektów o danej wartości cechy. Przyjęto konwencję, że kolor niebieski oznacza obiekty o decyzji „tak”, czerwony o decyzji „nie”. Tam, gdzie obie wartości są niezerowe, konieczne jest dalsze rozgałęzianie drzewa, gdyż decyzja na danym etapie jest niejednoznaczna. Z kolei tam, gdzie tylko jedna z nich jest zerem (np. „chol.=nis.”), podejmowana jest końcowa decyzja.

- kolejność testowania nie jest identyczna dla każdego obiektu. Przykładowo: obiekt o wartości „wiek=45” w następnym kroku testowaną ma wartość „cholesterol”, a „wiek=60” testowane ma „ciśnienie krwi”. Jest tak, ponieważ testowanie danej cechy rozdziela obiekty na różne, rozłączne zbiory. Te są w dalszych krokach przetwarzane z osobna i w takich różnych zbiorach inne atrybuty mogą mieć większe wartości IG .

- w liściach, w których brak reprezentantów („wiek=40” i „tusza=szczupły”) podejmowana jest decyzja mniej ‘kosztowna” – tzn. przy braku przesłanek, osobę uznaje się za zdrową i okresowo poddaje tylko rutynowym badaniom.

- po przetestowaniu wszystkich atrybutów, może wystąpić kilka obiektów o tych samych wartościach cech, ale różnych decyzjach („tusza=norm.” na samym dole drzewa, jeden obiekt „tak” i jeden „nie”). Wówczas także podejmowana jest decyzja mniej kosztowna (w tym przypadku nastąpi błędne zaklasyfikowanie jednej osoby „tak” do klasy „nie”).

Widoczne w drzewie wartości parametrów p_{chance} opisane są w dalszej części.

4.7 Algorytm budowania drzewa ID3

W literaturze spotkać można algorytm budowania drzewa, nazwany ID3 [1][2]. Opisany powyżej sposób wykorzystujący zysk informacyjny IG jest analogiczny z tym algorytmem. Formalnie zapisuje się go następująco:

1. Obliczyć należy entropię dla każdego atrybutu.
2. Wybrać atrybut A z najniższą entropią.
3. Podzielić zbiór przykładów uczących ze względu na wartość atrybutu A na rozłączne podzbiory.
4. Dodać do drzewa krawędzie z warunkami:

JEŚLI $A=a_1$ **TO** ... (poddrzewo 1)

JEŚLI $A=a_2$ **TO**... (poddrzewo 2)

5. Dla każdego poddrzewa wykonać kroki od 1 do 4.
6. Powtarzać do wyczerpania atrybutów.

Algorytm ten ma jednak następujące wady:

- uzyskiwane drzewo ma duży rozmiar (testuje wszystkie atrybuty i rozgałęzia się na każdą występującą w zbiorze wartość atrybutu,
- zakłada, że wartości atrybutów są dyskretne a nie ciągłe,
- zakłada, że rekordy w zbiorze treningowym są kompletne, tzn. nie zadziała, jeżeli choć jeden rekord zawiera niepełne dane,
- brak odporności na przetrenowanie – pomaga upraszczanie drzewa.

Powyższe wady i problemy opisane są dokładniej w kolejnych rozdziałach, które wskazują możliwe rozwiązania i usprawnienia dla: zbyt dużych drzew, przetrenowania, atrybutów ciągłych lub niekompletnych.

4.8 Błąd treningowy i testowy

Skuteczność drzewa decyzyjnego sprawdzana jest w następujący sposób. Dla każdego rekordu w tablicy zawierającej atrybuty obiektów, testuje się zgodnie ze strukturą drzewa kolejne atrybuty i sprawdza czy wskazana w liściu decyzja jest zgodna z decyzją w tabeli. Procentowa ilość niezgodnych wartości to błąd drzewa.

Dla danych treningowych w powyższym przykładzie, dla 40 osób, tylko 1 została zaklasyfikowana nieprawidłowo (czerwona jedynka w liściu „tusza=norm.”) (rys. 4.5), wobec tego błąd wynosi 1/40·100% = 2,5%.

Celem działania drzewa decyzyjnego nie jest jednak klasyfikacja obiektów już występujących w tabeli, ani testowanie skuteczności klasyfikacji danych treningowych. Drzewa decyzyjne, tak jak wszystkie inne algorytmy decyzyjne, stosowane są do klasyfikacji obiektów nieznanych. Celem treningu jest stworzenie aparatu decyzyjnego, który w przyszłości podejmie właściwe decyzje dla nowych obiektów, nieobecnych na etapie treningu.

Nowe dane, tzw. **zbiór testowy**, pozyskiwane są przed treningiem drzewa. Zbiór wszystkich danych, które są dostępne dzielony jest na rozłączne podzbiory: **dane treningowe** i **dane testowe** w dowolnym stosunku, 1:1, 2:1, itd. Wykluczenie danych testowych z treningu to sposób zasymulowania tego, że w przyszłości drzewo będzie musiało rozpoznawać dane zupełnie nowe.

W omawianym przykładzie danych medycznych, drzewo trenowano na 40 przykładach, a testuje się je na 352 innych, uzyskując błąd w 74 przypadkach: 74/352·100%=21,02%. Błąd testowy jest ponad ośmiokrotnie większy od treningowego.

Należy rozważyć skąd wynika tak duża **różnica między błędem treningowym a testowym** i czy można poprawić trafność klasyfikacji? Ponadto w wielu zastosowaniach istotne może być zmniejszanie rozmiaru drzewa – czy jest to możliwe bez obniżenia trafności?

4.9 Przetrenowanie

Wskazany powyżej problem dużego błędu dla danych testowych a małego dla treningowych nazywany jest **przetrenowaniem**. Dochodzi do niego wówczas, gdy w trakcie budowy drzewa jego struktura tworzona jest tak, aby uwzględniać wszystkie zależności między wartościami atrybutów a decyzjami – także te, które są tylko dziełem przypadku lub wynikają z błędu pomiaru albo są wynikiem pominięcia jakiegoś nieznanego zmiennego czynnika.

Poniżej zaproponowane jest całkowicie syntetyczne zagadnienie, które pokaże przyczyny przetrenowania i sposób zapobiegania mu.

4.9.1 Przykład kontrolowanego przetrenowania

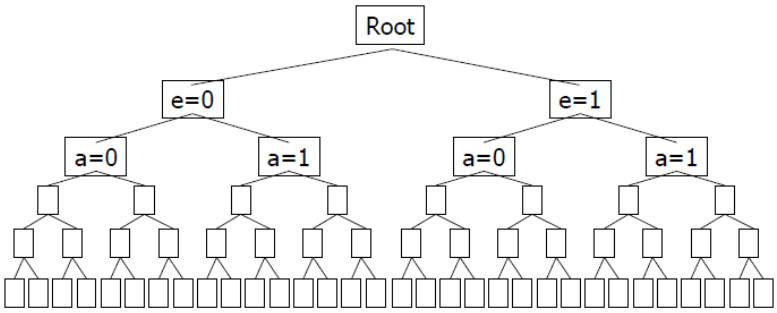
Niech zbiorem danych są abstrakcyjne obiekty opisane wszystkimi kombinacjami 5 bitów a,b,c,d i e i każdemu przypisana jest decyzja y, która:

- w 75% losowych przypadków jest kopią bitu „e”
- w pozostałych 25% przypadków jest negacją bitu „e” (rys. 4.6).

a	b	c	d	e	y
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	1
:	:	:	:	:	:
1	1	1	1	1	1

Rys. 4.6. Przykład danych syntetycznych: obiekty to wszystkie kombinacje 5 bitów, a decyzja „y” równa jest bitowi „e” lub negacji ¬e (kolor czerwony)

Intuicyjnie można dostrzec, że „y” zależy wyłącznie od „e”, gdyż to z niego zostało wyliczone. W procesie tworzenia drzewa opisaną metodą, bit „e” będzie miał rzeczywiście największą wartość IG i wykorzystany zostanie jako pierwsze rozgałęzienie drzewa. Inne rozgałęzienia sprawdzą pozostałe bity i drzewo dostosuje się do wszystkich 32 przypadków, uwzględniając je w indywidualnych liściach (rys. 4.7).



Rys. 4.7. Drzewo testujące wszystkie kombinacje pięciu bitów

Dla dowolnego obiektu opisanego tymi pięcioma bitami, zawsze wskazywana będzie właściwa, zgodna ze zbiorem treningowym odpowiedź „y”, gdyż każdy z obiektów ma swój własny liść i decyzję. **Błąd treningowy** wynosi 0%. W celu oceny drzewa konieczne jest obliczenie ponadto **błędu testowego**.

Zbiór testowy wykonany jest w identyczny sposób jak treningowy, z tym, że inne 25% wartości wyjściowych „y” jest negacjami „e”.

Dalej można się posłużyć analizą przypadków, które mogą zajść w trakcie testowania tymi danymi. Negacje „e” następują losowo w zbiorze treningowym, do którego w pełni dostosowuje się drzewo, te przypadki nazwane są decyzjami „uszkodzonymi”. Negacje „e” także losowo występują w zbiorze testowym – przypadki danych „uszkodzonych”. Dla każdego obiektu (każdej kombinacji pięciu bitów) zajść mogą następujące sytuacje (tab. 4.6):

Tab. 4.6. Klasyfikacja danych losowo zanegowanych (uszkodzonych) za pomocą reguł losowo zanegowanych (uszkodzonych)

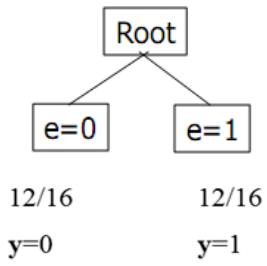
	¼ decyzji „uszkodzonych”	¾ decyzji dobrych
¼ danych „uszkodzonych”	1/16 zbioru testowego zostanie sklasyfikowana dobrze, ale przypadkowo	3/16 zbioru testowego zostanie **błędnie sklasyfikowanych**, ponieważ dane są „uszkodzone”
¾ danych dobrych	3/16 zbioru testowego zostanie **błędnie sklasyfikowanych**, ponieważ liście są „uszkodzone”	9/16 zbioru testowego zostanie sklasyfikowanych dobrze

Sumując wszystkie przypadki błędów, otrzyma się 3/16+3/16 = 3/8 = **37,5% błędu testowego**. Porównując z 0 procentowym błędem treningowym zauważyć należy różnicę, która dyskwalifikuje takie drzewo i nie pozwala używać go w praktyce. W dalszej części wyjaśnione zostanie skąd biorą się takie różnice w błędach i jak im zapobiegać. Zanim przedstawiona zostanie odpowiedź na to pytanie, rozpatrzony zostanie znacznie prostszy zbiór treningowy (rys. 4.8).

a	b	c	d	e	y
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	1
:	:	:	:	:	:
1	1	1	1	1	1

Rys. 4.8. Zbiór treningowy ograniczony tylko do atrybutu „e” i decyzji „y”

Bity a,b,c,d zostały usunięte, wyjście „y” równe jest w 75% przypadkach „e”, a w 25% przypadkach negacji „e”. Wytrenowane drzewo uwzględnić może tylko dostępne atrybuty, a więc testowana jest tylko wartość „e”. W większości przypadków y=e, wobec tego podejmowana przez drzewo decyzja jest następująca (rys. 4.9):



Rys. 4.9. Drzewo decyzyjne testujące tylko jeden dostępny atrybut „e”

12/16 przypadków zostaje zaklasyfikowanych do y=0, oraz 12/16 do y=1. Taka budowa drzewa nie pozwala na uwzględnianie przypadków z przekłamanym/zanegowanym „y”. Już w trakcie treningu drzewo obciążone jest **błędem treningowym** – **25%** przypadków klasyfikowanych jest niewłaściwie.

W zbiorze testowym inne 25% bitów jest zanegowanych, ale podejmowana decyzja to także y=e. Wobec tego **błąd testowy** wynosi tyle samo: **25%**.

Pomimo drastycznego uproszczenia drzewa do jednego tylko rozgałęzienia, udało się uzyskać spadek błędu z 37,5% do 25%.

4.9.2 Definicja przetrenowania

Jeżeli system decyzyjny analizuje dane nieistotne (szum, błędy pomiarowe, itd.), wówczas zachodzić może **przetrenowanie** (ang. *overfitting*). Przetrenowany system decyzyjny osiąga: *wysoką* trafność klasyfikacji **danych treningowych**, *niską* trafność klasyfikacji **danych testowych**. Przeciwieństwem przetrenowania jest zdolność generalizacji – nieuwzględnianie zależności, które są tylko dziełem przypadku lub wynikiem błędu pomiaru (błędu określenia wartości atrybutu lub decyzji).

Wskazane jest pomijanie danych nieistotnych w procesie treningu drzewa, aby uzyskać wysoką generalizację.

4.10 Upraszczanie drzewa

Zwykle brak jest informacji ujawniających, które atrybuty są nieistotne i doprowadzą do przetrenowania, a które mają faktyczny wpływ na decyzję. Uwzględnienie wyłącznie zysku informacyjnego IG prowadzi do zbudowania drzewa, które testuje wszystkie atrybuty. Na podstawie IG można uprościć drzewo, usuwając od dołu drzewa te rozgałęzienia, w których atrybuty miały bardzo małe IG i sprawdzając uzyskiwane po tej operacji wyniki. W opisanym powyżej syntetycznym przykładzie bity a,b,c,d miały bardzo niewielki i tylko przypadkowy wpływ na decyzję, ich IG było małe, wobec czego stosując dla IG odpowiednią wartość progową można drzewo uprościć do jednego tylko rozgałęzienia:

Jeżeli IG(*atrybut*) < $IG_{progowe}$ to usuń rozgałęzienie dla *atrybutu*

Jeżeli IG(*atrybut*) $\geq$ $IG_{progowe}$ to pozostaw rozgałęzienie dla *atrybutu*

Jest to tylko jedno z wielu podejść do zagadnienia upraszczania drzewa (ang. *pruning*).

4.10.1 Istotność danych – statystyka χ^2

Statystyka może dostarczyć informacji o tym, które atrybuty są nieistotne. Test χ^2 Pearsona jest nazywany testem istotności dla zmiennych jakościowych (skategoryzowanych). Miara oparta jest na

możliwości obliczenia **liczności oczekiwanych**, tj. liczności, jakie powinny wystąpić, gdyby **nie istniała zależność** między zmiennymi. Obserwowane liczności klas porównywane są z licznościami wyznaczanymi teoretycznie dla przypadku gdyby występowała całkowita losowość.

Przykład. Pytano 20 mężczyzn i 20 kobiet o upodobanie do jednej z dwóch gatunków wody mineralnej (gatunki A i B). Gdyby nie było **żadnej** zależności między upodobaniem odnośnie wody mineralnej a płcią, wówczas należałoby **oczekiwać** mniej więcej **jednakowych liczności** w preferencjach gatunku A i B dla obu płci, tzn. tyle samo kobiet i mężczyzn lubi wodę A (nie fakt która z nich jest lubiana bardziej, tylko zależność od płci). Test χ^2 staje się istotny w miarę **wzrostu odstępstwa** od tego oczekiwanego schematu (to znaczy w miarę jak odpowiedzi dla mężczyzn i kobiet zaczynają się różnić).

Testy statystyczne zakładają pewną wartość charakterystyczną (teoretyczną), wyliczaną na podstawie cech zbioru danych oraz określoną hipotezę zerową H_0, tzn. założenie specyficzne dla testu. W wyniku obliczeń zostaje określone, czy hipoteza jest odrzucana jako nieprawdziwa, czy nie ma podstaw do jej odrzucenia.

W teście χ^2 sprawdzana jest hipoteza zerowa o „**niezależności dwóch cech od siebie**”. Aby możliwe było wykonanie testu musi być spełniony warunek n>30, tzn. n-elementowa próba z populacji musi mieć liczność większą od 30. Próbki charakteryzowane są dwiema rozpatrywanymi w teście cechami, tymi, których (nie-)zależność jest sprawdzana. Cechy indeksowane są po i oraz po j. Cechy mają odpowiednio k i r różnych wartościach (np. ryzyko zawału ma k=2 {tak, nie}, pozom cholesterolu r=3, płcie są dwie, wody także dwie). Wartość χ^2 wyliczana jest dla zbioru danych w następujący sposób (4.5):

$$\chi^2 = \sum_{j=1}^{k}\sum_{i=1}^{r}\frac{(n_{ij}-\hat{n}_{ij})^2}{\hat{n}_{ij}} = \sum_{j=1}^{k}\sum_{i=1}^{r}\left(\frac{n_{ij}^2}{\hat{n}_{ij}}\right) - n \qquad (4.5)$$

gdzie:

n_{ij} – liczba istniejących w zbiorze danych obiektów opisanych konkretnymi wartościami i,j swoich atrybutów

$\hat{n}_{ij}$ – teoretyczna, oczekiwana liczność, wg. wzoru (4.6):

$$\hat{n}_{ij} = \frac{\sum_{j=1}^{k} n_{ij} \sum_{i=1}^{r} n_{ij}}{n} \qquad (4.6)$$

Ze wzoru (4.5) wywnioskować można, że χ^2 jest duże jeżeli różnice $(n_{ij} - \hat{n}_{ij})^2$ w liczniku także są duże dla wielu różnych i, j, tj. obserwowane liczności n_{ij} nie są takie jak oczekiwane liczności (4.6).

Odwrotnie: jeżeli obserwujemy liczności identyczne z oczekiwaniami, licznik się zeruje i $\chi^2 = 0$. Widoczne jest, że w miarę wzrostu odstępstwa od liczności oczekiwanych χ^2 stopniowo rośnie, konieczne jest więc ustalenie wartości progowej, powyżej której stwierdzone zostanie, że hipotezę H_0 się odrzuca i cechy są zależne od siebie. Wartość progową przyjmuje się z zadanym poziomem istotności, tj. można z dużym lub małym prawdopodobieństwem (istotnością) powiedzieć, że hipotezę H_0 się odrzuca. Ta wartość progowa zależy od *r* i *k* (ile różnych wartości mogą przyjmować cechy) oraz od zakładanego poziomu istotności testu (jaki procentowy poziom prawdopodobieństwa błędu jest akceptowany, tzn. wynik testu jest przyjmowany, gdy tymczasem z prawdopodobieństwem *p* może być on nieprawidłowy).

Wartość χ^2 wyliczoną z obserwowanego zbioru, porównać należy z $\chi^2_{p;\,(r-1)(k-1)}$ odczytaną z tablic statystycznych, gdzie w komórkach tabeli odnajduje się progowe χ^2 odpowiadające zadanemu w kolumnach tablic *p* - poziomowi istotności (np. 0,005; 0,01; 0,05), oraz wyliczonemu z cech iloczynowi odnajdowanemu w wierszach tablicy $(r-1)(k-1)$, który nazywany jest liczbą stopni swobody.

Jeżeli wyliczone χ^2 jest większe od $\chi^2_{p;\,(r-1)(k-1)}$ to odrzuca się hipotezę H_0 o niezależności cech (oznacza to, że **cechy są zależne** na poziomie istotności *p*). Jeżeli $\chi^2 < \chi^2_{p;\,(r-1)(k-1)}$ to nie ma podstaw do odrzucenia H_0 (**cechy są niezależne**, a obserwowane pozorne zależności to tylko „**dzieło przypadku**”).

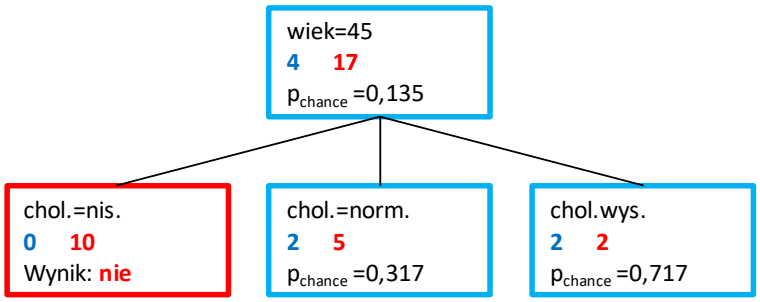
Dla konkretnych obserwowanych wartości χ^2 interesująca jest odpowiedź na pytanie – jaki musiałby być poziom istotności *p*, aby $\chi^2 < \chi^2_{p;\,(r-1)(k-1)}$, czyli aby cechy były niezależne? Dwie sprawdzane w teście χ^2 cechy to w przypadku drzewa zawsze atrybut testowany w danym rozgałęzieniu i decyzja końcowa. Jeżeli poziom istotności (czyli prawdopodobieństwo, że obserwacja jest dziełem przypadku) jest wysoki to decyzja podejmowana przez drzewo jest niewiarygodna i można z analizowanego rozgałęzienia zrezygnować.

4.10.2 Przypadkowość w zbiorze danych

Poniżej zostanie przeanalizowany fragment drzewa opisującego cechy osób i wynikowe ryzyko zawału. Dla każdego rozgałęzienia (rys. 4.10) wskazano wartość p_{chance}, która mówi z jakim prawdopodobieństwem zależności miedzy cechami a decyzjami, występujące poniżej rozgałęzienia, są tylko dziełem przypadku.

Roboczo można założyć, że w rzeczywistości wynikowe *ryzyko* jest całkowicie niezależne (nieskorelowane) z poziomem cholesterolu. Jakie jest wówczas prawdopodobieństwo zaobserwowania takich danych (danych, które są dziełem przypadku, a nie wynikają z zależności między atrybutami)? Otrzymuje się wartość $p_{\text{chance}} = 0{,}135$. Jednocześnie wyliczyć można dla testowanego atrybutu wartość zysku informacyjnego IG, która wynosi 0,224284 – atrybut wnosi

pewien wkład w podjęcie decyzji (rys. 4.10). Czy te przykładowe wartości p_{chance} i IG są duże, czy małe? Czy należy zrezygnować z testowania atrybutu *chol.* w tym rozgałęzieniu?



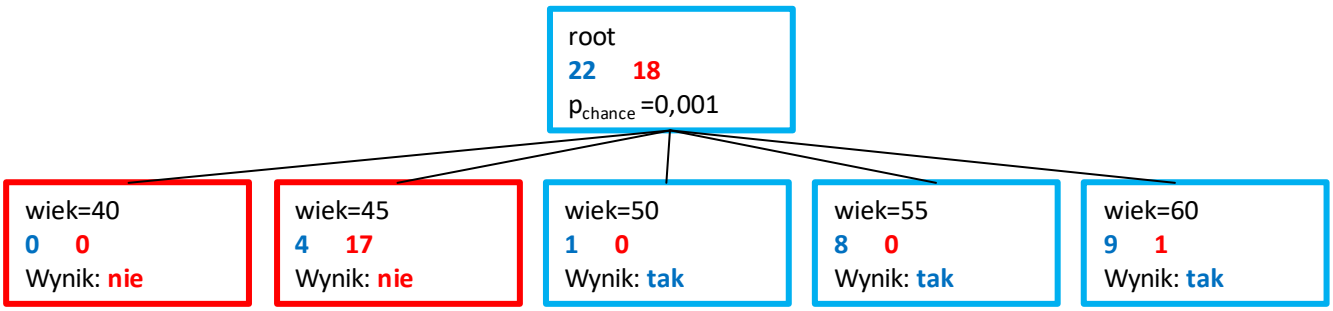
Rys. 4.10. Decyzje w oparciu o atrybut „chol.”: p_{chance} = 0,135 oznacza że obserwowane liczności mogą być dziełem przypadku z dość małym prawdopodobieństwem i decyzje są silnie zależne od atrybutu. Wykazuje to także wartość IG(ryzyko|chol.) równa 0,22

Upraszczanie drzewa realizowane może być zgodnie z różnymi strategiami, z których kilka wybrano i opisano poniżej.

4.10.3 Strategie upraszczania drzewa

Dla każdego z rozgałęzień drzewa obliczyć można z danych treningowych kilka parametrów, które dadzą wskazówkę dotyczącą ich istotności dla podjęcia decyzji.

Zwykle na podstawie wartości zysków informacyjnych IG buduje się kompletne drzewo decyzyjne, testujące wszystkie parametry, a następnie, przeglądając je od dołu, dokonuje się usuwania rozgałęzień (ang. *Pruning* – dosłownie: przycinanie). Kryterium może być oparte na wartości p_{chance}. Jeżeli p_{chance}>*MaxP* to **rozgałęzienie jest usuwane, zastępowane jest liściem**, w którym podejmowana jest decyzja najczęstsza wśród obiektów, które przechodziły przez to rozgałęzienie, lub decyzja najmniej ryzykowna. Parametr *MaxP* dobierany jest w zależności od chęci podejmowania ryzyka dopasowania drzewa do danych potencjalnie nieistotnych. Dla przykładu wykorzystywanego w tym rozdziale dla *MaxP* = 0,1 uzyskuje się drzewo (rys. 4.11):



Rys. 4.11. Decyzje po uproszczeniu drzewa dla *MaxP*=0,1

Usunięto z drzewa wszystkie rozgałęzienia oprócz pierwszego. Obserwacja błędu treningowego i testowego pokaże, czy było to słuszne postępowanie (tab. 4.7 i 4.8).

Tab. 4.7. Błędy testowe i treningowe dla drzewa pełnego

	Liczba błędów	Liczba obiektów	Procent błędnych decyzji
Zbiór treningowy	1	40	**2,5 %**
Zbiór testowy	74	352	**21,02 %**

Tab. 4.8. Błędy testowe i treningowe dla drzewa uproszczonego

	Liczba błędów	Liczba obiektów	Procent błędnych decyzji
Zbiór treningowy	5	40	**12,5 %**
Zbiór testowy	56	352	**15,91 %**

Zauważyć należy zwiększenie sześciokrotnie błędu treningowego w wyniku uproszczenia (drzewo uwzględnia poprawnie tylko 35 spośród 40 przypadków, dla 5 popełnia błąd) oraz **pożądane zmniejszenie błędu testowego** z 21% do ok. 16%.

Ustalając wartości *MaxP*, należy mieć na uwadze, że zbyt małe *MaxP* wprowadza duży błąd z powodu zbyt dużego **uogólnienia**. Z kolei zbyt duże *MaxP* to duży błąd z powodu **przetrenowania**. Nie ma jednej uniwersalnej wartości *MaxP*, jednak dla zbioru danych można automatyczne wyznaczyć najlepsze *MaxP*, stopniowo zmniejszając od *MaxP* =1, weryfikując jednocześnie **zmiany błędu testowego**.

4.11 Drzewa decyzyjne dla danych ciągłych

W wielu przypadkach praktycznych obiekty w zbiorze danych opisane mogą być atrybutami przyjmującymi wartości ciągłe, a nie dyskretne, o dowolnej lub zadanej dokładności (liczby całkowite lub nie, z dokładnością do kilku miejsc po przecinku, itd.). Również w wykorzystywanym przykładzie danych medycznych, atrybuty osób mogą mieć różną dokładność i wartości ciągłe, takie jak np. waga ciała (wyrażona z dokładnością do dziesiątej części kilograma), wyniki morfologii krwi z dokładnością do setnych części, a także wiele innych. W zagadnieniach medycznych, ekonomicznych, technicznych będą występowały atrybuty ciągłe: temperatura ciała, ciśnienie tętnicze, wyniki testów i pomiarów. Jak tworzyć test atrybutu w rozgałęzieniach drzewa, aby uwzględnić przypadki występujące w zbiorze treningowym? Czy w zbiorze testowym natrafi się na te same liczby, czy może różniące się na ostatnim miejscu po przecinku i nieuwzględnione do tej pory w drzewie?

Również atrybuty z bardzo wieloma całkowitymi wartościami, choć są dyskretne (nieciągłe), to nastręczają podobne problemy – np. czy wiek pacjenta z dokładnością do pojedynczego roku rozgałęzić trzeba na 100 różnych wartości? Przy wielokrotnym rozgałęzianiu uzyskiwane będą duże wartości p_{chance}, co doprowadzi na etapie upraszczania do usunięcia wielu poziomów drzewa.

Rozgałęzienie na każde możliwe wartości nie jest realne, gdyż doprowadzi do **przetrenowana** i bardzo słabej klasyfikacji obiektów testowych. Wobec tego stosuje się podział całej dziedziny na podprzedziały, a zamiast wartości używa się nazw tych podprzedziałów, etykiet zastępczych, co nazywane jest **dyskretyzacją**.

Gdyby zdecydować się na dyskretyzację na przedziały zadane przez eksperta, twórcę algorytmu, to czy lepiej wiek pacjenta dzielić na przedziały co 5 lat, może co 10? Jednakże nie ma powodów, aby przypuszczać, że na przełomie lat na przykład 59/60 regularnie następuje drastyczny spadek stanu zdrowia i osoby z tych przedziałów będą istotnie się różniły od siebie i taki test w drzewie będzie zasadny. W określeniu sposobu dyskretyzacji oraz właściwego podziału na podprzedziały dyskretyzacji pomoże zysk informacyjny.

4.11.1 Przedziały dyskretyzacji

Zamiast użycia zbyt wielu wartości atrybutu posłużyć można się nazwą/etykietą/symbolem przedziału. Przedziały nie muszą być określane ręcznie przez eksperta. Algorytm dyskretyzacji pozwoli tak określić tzw. cięcia, aby uzyskiwane w drzewie rozgałęzienia rzeczywiście testowały istotne wartości – aby znajomość podprzedziału, do którego należy atrybut istotnie pomagała w podjęciu decyzji.

Sformułowanie „atrybut istotnie wpływa na decyzję” powinno nasuwać skojarzenia z pojęciem zysku informacyjnego wprowadzonego w części dotyczącej budowy drzewa i wyboru kolejności atrybutów. Podobnie w procesie dyskretyzacji: dla określenia przydatności podziału dziedziny atrybutu wylicza się IG.

Niech IG(Y|X:t) oznacza zysk informacyjny dla wartości Y pod warunkiem, że wiadomo, czy X jest większe czy mniejsze od progu *t*. Ta wartość liczona jest w następujący sposób:

$$IG(Y|X:t) = H(Y) - H(Y|X:t) \tag{4.7}$$

gdzie:

$$H(Y|X:t) = H(Y|X<t)\cdot P(Y|X<t) + H(Y|X\geq t)\cdot P(Y|X\geq t) \tag{4.8}$$

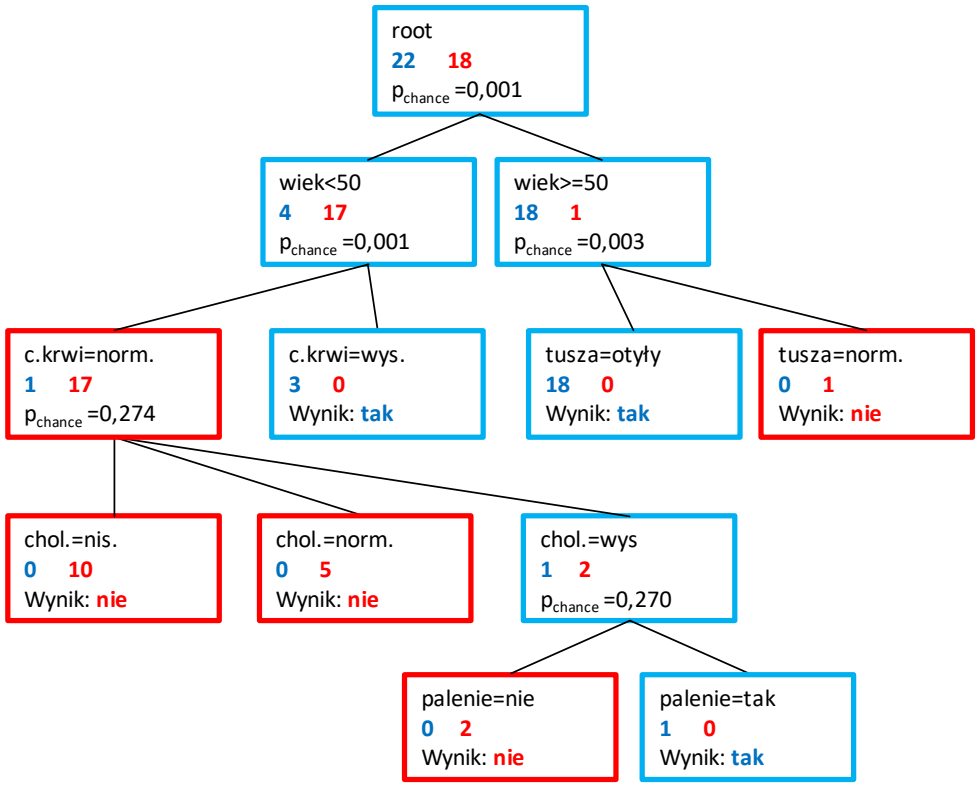
t - wartość dzieląca dziedzinę X na przedziały.

Ponadto:

$$IG^*(Y|X) = \max_t(IG(Y|X:t)) \tag{4.9}$$

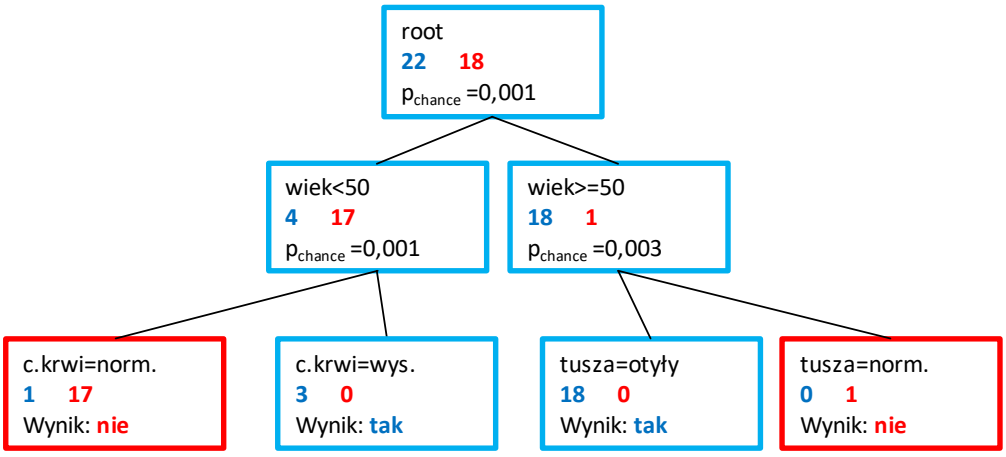
IG* jest największym możliwym do osiągnięcia zyskiem przy podziale dziedziny, a *t* to miejsce podziału generujące największy IG(Y|X:t). Test w rozgałęzieniu drzewa będzie zależny od wyznaczonego t.

Dla przykładu danych medycznych, stosując dyskretyzację atrybutów liczbowych i symbolicznych (wiek, ciśnienie krwi), uzyskuje się zmniejszone drzewo (rys. 4.12).



Rys. 4.12. Drzewo decyzyjne wykorzystujące dyskretyzację atrybutów

Poprzez ustalenie progowej wartości *MaxP* (czytelnik sam może określić jej wartość porównując drzewa z rys. 4.12 i 4.13) uzyskuje się uproszczone drzewo i dalszą poprawę skuteczności klasyfikacji (tab. 4.9).



Rys. 4.13. Drzewo decyzyjne wykorzystujące dyskretyzację i upraszczanie

Tab. 4.9. Błędy testowe i treningowe dla drzewa z dyskretyzacją i upraszczaniem

	Liczba błędów	Liczba obiektów	Procent błędnych decyzji
Zbiór treningowy	1	40	**2,5 %**
Zbiór testowy	53	352	**15,06 %**

Porównać należy samodzielnie wyniki i stopień złożoności pierwszego drzewa pełnego, bez dyskretyzacji, z przetrenowaniem i błędem testowym **21,05%** z ostatecznie uzyskanym drzewem z dwoma testami, błędem testowym **15,06%**.

4.12 Algorytm budowania drzewa C4

Przedstawione powyżej postępowanie, uwzględniające przycinanie drzewa i dyskretyzację parametrów, wchodzi w skład algorytmu budowania drzew decyzyjnych o nazwie C4 [2][3]. W uzupełnieniu należy wymienić cechy dodatkowe tego algorytmu:

- gdy dane są niekompletne, tj. rekordy mają **nieznaną** wartość dla pewnych atrybutów, to IG jest wyliczane tylko na podstawie rekordów, gdzie dana wartość jest zdefiniowana,
- atrybuty mogą mieć ciągłe wartości (stosuje się dyskretyzację w oparciu o IG*),
- przycinanie – wprowadzone jest w wersji algorytmu nazywanej „C4.5”. Zaczyna się od liści i działa w górę, porównując:
    - **przewidywany błąd** dla wybranego węzła i jego poddrzewa,
    - **przewidywany błąd** dla tego poddrzewa zastąpionego tylko jednym liściem z decyzją najpopularniejszą w tym poddrzewie.

4.13 Wybrane warianty drzew decyzyjnych

Omówione powyżej metody tworzenia struktury drzewa decyzyjnego są najczęściej stosowanymi, klasycznymi algorytmami. Budowanie drzew decyzyjnych na bazie zysku informacyjnego jest podstawą działania algorytmów drzew decyzyjnych C4.5 oraz CART. Oba te algorytmy mogą występować w wersji bez przycinania lub z przycinaniem gałęzi.

Algorytm C4.5 dopuszcza dowolną ilość wyników testu węzłowego, to znaczy, że z każdego węzła może wychodzić dowolna ilość gałęzi, uzasadniona zyskiem IG*.

Algorytm CART zezwala jedynie na podział binarny – węzeł może być zakończony maksymalnie dwoma gałęziami lub dwoma liśćmi.

Interesującą odmianą drzew decyzyjnych są algorytmy Random Tree (RT) i Random Forest (RF) [4]. Algorytm RT tworzy drzewo decyzyjne poprzez wybór *k* losowych atrybutów dla każdego z węzłów. Algorytm ten cechuje się niską skutecznością jest jednak bazą do działania algorytmu RF. RF wykorzystuje wiele klasyfikatorów RT do podjęcia decyzji. Każde drzewo RT działa niezależnie i daje własny wynik. Wybierana jest ta odpowiedź, która pojawiła się najczęściej. Zestawienie wielu „słabych” klasyfikatorów daje w konsekwencji precyzyjny wynik, przy zachowaniu dużej wydajności czasowej obliczeń. Zarówno RT jak i RF nie wykorzystują przycinania gałęzi.

Ponadto drzewa rozróżnia się według wyniku ich działania na typy:

Drzewo klasyfikacyjne – wynikiem jest przypisanie do klasy (jednej z określonej liczby wytrenowanych klas).

Drzewo regresyjne – wynikiem jest liczba rzeczywista z dziedziny ciągłej, np. przewidywana cena (dowolna wartość, nawet niewystępująca wcześniej w zbiorze treningowym).

Classification And Regression Tree (CART) – obie powyższe metody w hybrydowym systemie decyzyjnym.

4.14 Podsumowanie

Drzewo uproszczone i prawidłowo wykonana dyskretyzacja dostarczają istotnych praktycznych informacji:

– jakie atrybuty są ważne w procesie decyzji,

– jakie podprzedziały wartości są istotne.

Często dobrze utworzone drzewo decyzyjne pozwala dokładniej zrozumieć problem i dostrzec zależności między danymi. Jeżeli drzewo posiada zdolność generalizacji (dużą skuteczność klasyfikacji danych testowych), to nadaje się ono do praktycznego wykorzystania. Implementacja algorytmu gotowego drzewa jest bardzo prosta i polega na zastosowaniu kilku zagnieżdżonych testów logicznych IF – THEN – ELSE lub SWITCH – CASE, co jest łatwe w każdym języku programowania.

4.15 Literatura

[1] Quinlan, J. R. *Induction of Decision Trees*. Machine Learning, vol. 1, pp. 81-106, 1986

[2] Mitchell, T. M. *Machine Learning*. McGraw-Hill, pp. 55–58, 1997

[3] Quinlan, J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993

[4] Breiman, L. *Random Forests*. Machine Learning, vol. 45 (1), pp. 5–32, 2001

5 Zbiory przybliżone

5.1 Historia zbiorów przybliżonych

Teoria zbiorów przybliżonych (ang. *Rough Set*) stworzona została przez Polaka, prof. Zdzisława Pawlaka. Pierwsza praca na ten temat [1] wprowadziła postawy teorii, która bardzo szybko znalazła praktyczne zastosowanie w systemach informatycznych [2] i uzyskała dojrzałość – dostrzeżono jej liczne zalety i związki z innymi metodami wnioskowania [3].

5.2 System informacyjny i decyzyjny

Teoria zbiorów przybliżonych zajmuje się **klasyfikacją danych** zorganizowanych w postaci tabel. Dane uzyskane mogą być z pomiarów, testów lub od ekspertów. Głównym celem analizy danych wejściowych jest wyznaczenie aproksymacji (przybliżenia) badanej idei (koncepcji, np. pacjenta zdrowego, chorego, itd.) w celu dokładnej **analizy problemu**, związków i zależności między atrybutami i decyzjami oraz uzyskania narzędzia **klasyfikującego** nowe przypadki.
W przeciwieństwie do innych metod wnioskowania, w teorii zbiorów przybliżonych dopuszcza się: **nieprecyzyjne dane, sprzeczność danych, niekompletność danych.**

Wprowadza się następujący formalizm:

System informacyjny jest to zorganizowany zestaw danych, w postaci tabeli, w której wiersze reprezentują indywidualne **obiekty**, np. pomierzone w eksperymencie, obserwowane, historyczne, itd., a **atrybuty** obiektów zapisane są w osobnych kolumnach tej tabeli.

U – zbiór obiektów (od słowa „uniwersum”), np. $U=\{x_1, x_2, x_3, x_4\}$, gdzie x_i to obiekt i-ty,

A – zbiór odwzorowań, pomiarów, oznaczanych jako a, z obiektu na wartość jego cechy, tj. a: $U \rightarrow V_a$ dla każdego $a \in A$. Obiektowi x_n z U w wyniku pomiaru a_n, przypisywana jest wartość mierzonej cechy V_a, dla całej puli A metod pomiaru a_n.

Oznaczyć można system informacyjny jako:

$$SI = (U, A) \tag{5.1}$$

Przykład. Obiekty oraz metody ich pomiaru i wartości atrybutów, występujące w systemie informacyjnym różnić się będą w zależności od zastosowań. Przykładowo, czujnik światła w prostej lampie, która ma się sama włączyć, mierzy napięcie na fotodiodzie. Obiektem jest aktualna sytuacja w pobliżu lampy, atrybutem jest poziom oświetlenia zewnętrznego, metodą pomiaru jest pomiar

napięcia. Inteligentniejsza lampa, która włącza się przy słabym oświetleniu oraz po stwierdzeniu ruchu w jej pobliżu, musi mierzyć jeszcze jeden atrybut – obecność ruchu.

Medyczny system informacyjny, w którym obiektami są pacjenci z urazem kręgosłupa, może mieć następującą postać (patrz tab. 5.1).

Tab. 5.1. Przykład systemu informacyjnego

	Age	*LEMS*
x_1	16-30	50
x_2	16-30	0
x_3	31-45	1-25
x_4	31-45	1-25
x_5	46-60	26-49
x_6	16-30	26-49
x_7	46-60	26-49

Tab. 5.1 zawiera dane 7 pacjentów w różnym wieku (*Age*) z różnym wynikiem testu motoryki kończyn dolnych – *Lower-Extremity Motor Score* (LEMS).

Pytanie. W tabeli znajdują się dane pacjentów anonimowych, o których nie wiadomo nic, ponad te dwa atrybuty. Czy przy tym stanie wiedzy, są obiekty nierozróżnialne (tożsame ze sobą)?

Na temat obiektów systemu informacyjnego, w wyniku obserwacji, dedukcji, badań lub doświadczenia eksperta, można powiedzieć coś jeszcze – **określić wynikową decyzję**, dla każdego z obiektów. Powstaje w ten sposób **system decyzyjny**:

$$SD = (U, A, \{d\}) \tag{5.2}$$

gdzie *d* to zbiór możliwych decyzji, adekwatnych dla obiektów danego uniwersum *U*.

System decyzyjny dla powyższego przykładu medycznego przyjmuje postać:

Tab. 5.2. Przykład systemu decyzyjnego

	Age	*LEMS*	*Walk*
x_1	16-30	50	Yes
x_2	16-30	0	No
x_3	31-45	1-25	No
x_4	31-45	1-25	Yes
x_5	46-60	26-49	No
x_6	16-30	26-49	Yes
x_7	46-60	26-49	No

W ostatniej kolumnie tabeli 5.2 zapisano efekt terapii pacjenta – czy odzyskał on (*Yes*) czy nie (*No*) zdolność chodzenia (*Walk*).

Jak wskazano powyżej, występują obiekty tożsame, pod względem wartości atrybutów, tj. para x_3 i x_4 oraz para x_5 i x_7, jednakże dla tej pierwszej pary **decyzje nie są identyczne** – występuje **sprzeczność danych**.

Poniżej zostaną rozwinięte zagadnienia decyzji, tożsamości obiektów i modelowania zbiorów przybliżających określone koncepcje w oparciu o sprzeczne dane (w tym przypadku zbiór pacjentów chodzących i nie).

5.3 Reguły decyzyjne

Powyższy system decyzyjny odczytywać można jako zbiór siedmiu reguł logicznych, przykładowo dla pacjenta x_1:

JEŻELI Age=”16-30” **I** LEMS=”50” **TO** Walk=”Yes” (5.3)

Reguły takie wykorzystane mogą być **do klasyfikacji** przypadków nowych, dla których nie jest znana wartość decyzji i celem jest jej prognoza. Algorytm wnioskujący musi wówczas przeanalizować poprzedniki reguł (JEŻELI … I …) (5.3), znaleźć regułę o wartościach atrybutów odpowiadających nowemu przypadkowi i odczytać następnik reguły (TO …) i zwrócić go jako wynik. Ze względu na występujące sprzeczności w systemie decyzyjnym, nie zawsze możliwe jest podanie jednoznacznej odpowiedzi. Teoria zbiorów przybliżonych podaje rozwiązanie tego problemu niejednoznaczności i sprzeczności.

5.4 Tożsamość obiektów

Dla podanego przykładu pod względem wartości atrybutów pary x_3 i x_4 oraz x_5 i x_7 są nierozróżnialne. Należy podkreślić, że dodanie nowego atrybutu, np. płci, wagi lub historii choroby pacjenta, może spowodować, że obiekty te staną się rozróżnialne – tożsamość określa się względem wybranego podzbioru atrybutów $B \subseteq A$ (zawiera się, czyli może także być równy A – wszystkim atrybutom). Wybranie podzbioru B = {*Age*} sprawia, że tożsame stają się obiekty x_1, x_2 i x_6 (wszystkie mają tę samą wartość *Age*), gdyż przy takim jednoelementowym zbiorze atrybutów B nic innego ich od siebie nie rozróżnia.

Pytanie. Jakie obiekty są tożsame, jeżeli B kolejno równe jest {*Age*}, {*LEMS*}, {*Age, LEMS*}?

5.4.1 Relacja równoważności

Z każdym podzbiorem atrybutów: $B \subseteq A$ związana jest relacja IND (ang. *indiscernibility* – nierozróżnialność, tożsamość) (5.4):

$$IND(B) = \{(x, x') \in U^2 \mid \forall a \in B \;\; a(x)=a(x')\} \quad (5.4)$$

Zapis wzoru (5.4) można odczytać: zbiór takich par (x, x') z uniwersum, że dla każdego ich atrybutu a z podzbioru atrybutów B, wartości atrybutu dla obu obiektów są równe.

Jeżeli: $(x, x') \in IND(B)$ to x i x' są tożsame względem relacji $IND(B)$, tj. nierozróżnialne względem atrybutów B.

W teorii zbiorów (ogólnej teorii, nie chodzi o zbiory przybliżone), mówi się o relacjach obiektów. Między innymi wyróżnia się tzw. **relację równoważności** dwóch obiektów. Aby sprawdzić czy dana relacja R jest relacją równoważności należy sprawdzić, czy dla każdego x, y, z zachodzą trzy warunki:

(xRx), co można zapisać jako $(x,x) \in R$ – **zwrotność** relacji

Z faktu (xRy) wynika (yRx) – **symetryczność** relacji

Z faktu, że (yRx) i (yRz) wynika (xRz) – **przechodniość** relacji.

Okazuje się, że relacja $IND(B)$ jest taką relacją równoważności.

Przykład. Niech x, y, z będą liczbami naturalnymi a relacja R będzie równością „=”. Prosto wykazać można, że „=” jest zwrotna, bo $x=x$; symetryczna (jeśli $(x=y)$ to $(y=x)$); przechodnia (jeśli $x=y$ i $y=z$ to $x=z$).

Sprawdzić można, że relacja większości „>” nie jest relacją równoważności, ale jest przechodnia.

5.4.2 Klasa abstrakcji

Dla każdego dowolnego obiektu x odnalezione w uniwersum inne obiekty o tych samych wartościach atrybutów B tworzą zbiór nazywany **klasą abstrakcji**, oznaczany jako $[x]_B$. Zwrócić należy uwagę na indeks $_B$ w tym zapisie, który sugeruje, że dla innych $B \subseteq A$ te klasy mogą być inne.

Zauważamy, że wewnątrz klasy abstrakcji obiekty są tożsame ze sobą. Aby coś powiedzieć o klasie abstrakcji wystarczy zbadać (określić atrybuty) jeden z tych obiektów.

Przykład. Jeżeli B to kolor dominujący widzianego przedmiotu, to czerwony przedmiot x generuje klasę abstrakcji zawierającą wszystkie czerwone obiekty z uniwersum. Dla innych B, klasami abstrakcji mogą być np. obiekty o identycznym *rozmiarze i kolorze, wadze i płci,* itd.

5.4.3 Zbiory elementarne

Dla przykładu pacjentów w zbiorze atrybutów A występują trzy niepuste podzbiory:

$$B_1=\{Age\}$$

$$B_2=\{LEMS\}$$

$$B_3=\{Age,LEMS\}$$

Dla każdego z nich uzyskujemy inne relacje równoważności:

$$IND(\{Age\}) = \{\{x_1; x_2; x_6\}; \{x_3; x_4\}; \{x_5; x_7\}\}$$

$$IND(\{LEMS\}) = \{\{x_1\}; \{x_2\}; \{x_3; x_4\}; \{x_5; x_6; x_7\}\}$$

$$IND(\{Age,LEMS\}) = \{\{x_1\}; \{x_2\}; \{x_3; x_4\}; \{x_5; x_7\}; \{x_6\}\}$$

Każda taka relacja równoważności prowadzi do podziału uniwersum na tzw. **zbiory elementarne**. Można zwrócić uwagę, że uwzględnienie większej liczby atrybutów ({*Age*} w porównaniu do {*Age, LEMS*}) zwykle może prowadzić (nie musi) do uzyskania „drobniejszego” podziału uniwersum na zbiory elementarne.

5.5 Aproksymacja zbioru

Każda suma zbiorów elementarnych tworzy tzw. **zbiór definiowalny**. Rodzina zbiorów definiowalnych (wszystkie możliwe zbiory definiowalne, wszystkie możliwe sumy zbiorów elementarnych) oznaczana jest jako *Def*(*B*). Podziały uniwersum na zbiory elementarne za pomocą relacji równoważności służą do tworzenia podzbiorów uniwersum, które wykorzystuje się w zadaniach klasyfikacji i wnioskowania. Zwykle poszukiwane są podzbiory definiowalne charakteryzujące się taką samą wartością atrybutu decyzyjnego.

Tab. 5.3. Przykład obiektów tożsamych w systemie decyzyjnym o sprzecznych decyzjach

	Age	*LEMS*	*Walk*
x_1	16-30	50	Yes
x_2	16-30	0	No
x_3	31-45	1-25	No
x_4	31-45	1-25	Yes
x_5	46-60	26-49	No
x_6	16-30	26-49	Yes
x_7	46-60	26-49	No

W tabeli 5.3 zaznaczono obiekty, które są tożsame, czyli są w tym samym zbiorze elementarnym, jednakże posiadają różne atrybuty decyzyjne. Korzystając ze zbiorów elementarnych nie da się w tym przypadku stworzyć **zbioru definiowalnego** pacjentów o decyzji jednoznacznej *Yes* lub decyzji *No.* Poszukiwany zbiór pacjentów jest **niedefiniowalny**. Należy posłużyć się jego **aproksymacją**.

5.5.1 Dolna i górna aproksymacja zbioru

Pomimo wykazanej powyżej niejednoznaczności możliwe jest określenie, które obiekty **na pewno należą** do poszukiwanego zbioru, które na pewno do niego **nie należą**, a które leżą **częściowo** w tym zbiorze (na jego granicy). Jeżeli w opisywanym zbiorze jakiekolwiek obiekty leżą na granicy, mamy do czynienia ze zbiorem przybliżonym.

Możliwa jest aproksymacja rozpatrywanego zbioru *X* (w naszym przypadku zbioru z decyzją *Walk=Yes*) wyłącznie przez wykorzystanie atrybutów ze zbioru *B*, poprzez określenie *B-dolnej* (5.5) i *B-górnej* (5.6) aproksymacji zbioru *X*:

$$\underline{B}X = \{ x \mid [x]_B \subseteq X \} \tag{5.5}$$

Zapis ten odczytuje się: do *B-dolnej* aproksymacji należą te *x*, których klasy abstrakcji $[x]_B$ w całości zawierają się w rozpatrywanym zbiorze (ich elementy nie mogą mieć innych decyzji).

$$\overline{B}X = \{ x \mid [x]_B \cap X \neq \emptyset \} \tag{5.6}$$

Zapis ten odczytuje się: do *B-górnej* aproksymacji należą te *x*, których klasy abstrakcji $[x]_B$ mają część wspólną z rozpatrywanym zbiorem niepustą (co najmniej jeden ich element ma właściwą decyzję).

5.5.2 Przykład

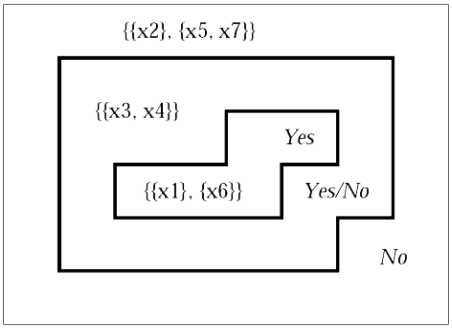
W tabeli 5.4 zaznaczono elementy na pewno należące do poszukiwanego zbioru *Walk=Yes* (niebieski), czyli takie, których zbiory elementarne i klasy abstrakcji mają decyzję *Walk=Yes* oraz elementy należące do granicy zbioru *Walk=Yes* (czerwony), czyli takie, których klasy abstrakcji mają obiekty o decyzji *Walk=Yes* ale także obiekty o decyzji *Walk=No.*

Tab. 5.4. Podzbiór obiektów wskazujących na decyzję *Yes* oraz ich obiekty tożsame (x_3 i x_4)

	Age	*LEMS*	*Walk*
x_1	16-30	50	Yes
x_2	16-30	0	No
x_3	31-45	1-25	No
x_4	31-45	1-25	Yes
x_5	46-60	26-49	No
x_6	16-30	26-49	Yes
x_7	46-60	26-49	No

B-dolną aproksymacją jest $\{x_1, x_6\}$ – niebieskie, na pewno należą, decyzja jednoznaczna *Walk=Yes*.
B-górną aproksymacją jest $\{x_1, x_3, x_4, x_6\}$ – czerwone i niebieskie, te które mają decyzję jednoznaczną, ale też i te, które mają decyzję jednocześnie *Yes* i *No*.

Wygodna graficzna reprezentacja tego zbioru przedstawiona jest poniżej (rys. 5.1).



Rys. 5.1. Graficzna interpretacja zależności między przybliżeniami rozpatrywanego zbioru

Należy zauważyć, że posługiwanie się tylko B_1={*Age*} lub B_2={*LEMS*} prowadzi do innego przybliżenia zbioru *Walk=Yes* niż w powyższym przypadku, gdzie B_3={*Age*, *LEMS*}. Z tego właśnie powodu mówi się „be-górna”, a nie po prostu „górna” aproksymacja.

5.5.3 Obszar graniczny i zewnętrzny

Przybliżenia $\bar{B}X$ i $\underline{B}X$ są zbiorami, na których elementach wykonywać można działania. Między innymi:

$$\mathrm{BND_B}(X) = \bar{B}X - \underline{B}X \tag{5.7}$$

to **obszar graniczny**, ang. *boundary*, zawiera te obiekty *x*, co do których nie można jednoznacznie zdecydować czy należą czy też nie do zbioru *X.* W rozpatrywanym przykładzie są to obiekty x_3, x_4.

$$\mathrm{EXT\ B}(X) = U - \bar{\mathrm{B}}\mathrm{X} \tag{5.8}$$

to **obszar zewnętrzny**, dopełnienie, ang. *exterior*, te *x*, które z całą pewnością nie należą do zbioru *X.* Obiekty x_2, x_5, x_7.

5.5.4 Dokładność przybliżenia

Aproksymację zbioru wyznacza się w oparciu o wiedzę zawartą w *B*. Zakłada się, że rozszerzanie liczby atrybutów, poprzez pomiar innych cech obiektów prowadzić będzie do uzyskania lepszej wiedzy o nich i do zmniejszenia mocy (liczby elementów) obszaru granicznego, czyli niejednoznacznych decyzji. Aby liczbowo wyrażać wpływ doboru atrybutów na dokładność przybliżenia rozpatrywanego zbioru obiektów, wylicza się miarę dokładności (5.9):

$$\alpha_{\mathrm{B}}(\mathrm{X}) = \frac{|\underline{\mathrm{B}}\mathrm{X}|}{|\bar{\mathrm{B}}\mathrm{X}|} \tag{5.9}$$

gdzie |…| oznacza moc zbioru, tj. liczbę jego elementów. Miara ta przyjmuje wartości $0 \le \alpha_B(X) \le 1$, gdzie $\alpha_B(X)$ =1 zachodzi dla zbioru tradycyjnego, który ma pusty obszar graniczny $\mathrm{BND_B}(X)$, a $\alpha_B(X)$ <1 dla zbiorów przybliżonych. Podstawiając ze wzoru (5.7) na obszar graniczny uzyskuje się (5.10):

$$\alpha_{\mathrm{B}}(\mathrm{X}) = \frac{|\underline{\mathrm{B}}\mathrm{X}|}{|\underline{\mathrm{B}}\mathrm{X}| + |\mathrm{BND_B(X)}|} \tag{5.10}$$

Z powyższego wyraźnie widać, że dla $|BND_B(X)|$ równego zero miara przyjmuje wartość 1, zbiór jest w pełni określony, tradycyjny, a im większe $|BND_B(X)|$, tym większy mianownik i mniejsza miara $\alpha_B(X)$. W końcu dla zbioru, który nie ma jednoznacznych obiektów, czyli jego $|\underline{B}X|$=0 miara ta przyjmuje wartość 0.

5.6 Własności zbiorów przybliżonych

Zachodzą następujące właściwości.

$$1)\ \underline{B}X \subseteq X \subseteq \overline{B}X$$
$$2)\ \underline{B}U = U = \overline{B}U$$
$$3)\ \underline{B}\varnothing = \varnothing = \overline{B}\varnothing$$
$$4)\ \underline{B}\underline{B}X = \overline{B}\underline{B}X = \underline{B}X$$
$$5)\ \overline{B}\overline{B}X = \underline{B}\overline{B}X = \overline{B}X$$
$$6)\ \overline{B}(-X) = -\underline{B}X$$
$$7)\ \underline{B}(-X) = -\overline{B}X$$

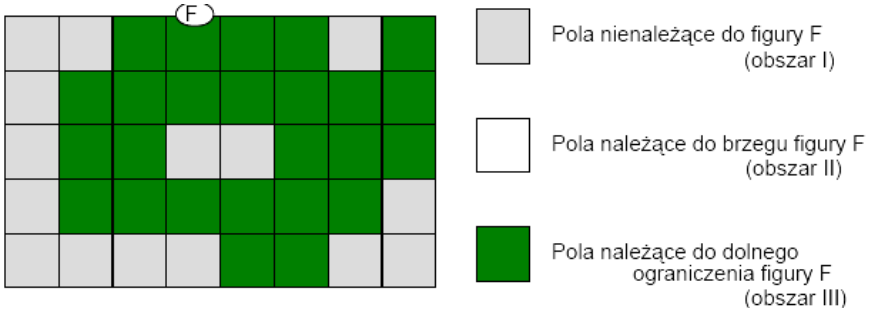
$-X$ oznacza $U-X$

$$8)\ \overline{B}(X \cup Y) = \overline{B}X \cup \overline{B}Y$$
$$9)\ \overline{B}(X \cap Y) = \underline{B}X \cap \underline{B}Y$$
$$10)\ \underline{B}(X \cup Y) \supseteq \underline{B}X \cup \underline{B}Y$$
$$11)\ \underline{B}(X \cap Y) \subseteq \underline{B}X \cap \underline{B}Y$$
$$12)\ BN(X \cup Y) \subseteq BNX \cup BNY$$
$$13)\ BNX = BN(-X)$$
$$14)\ \underline{B}(BNX) = \varnothing$$
$$15)\ \overline{B}(BNX) = BNX$$
$$16)\ BNX = \varnothing \Leftrightarrow \underline{B}X = X$$

5.7 Kategorie zbiorów przybliżonych

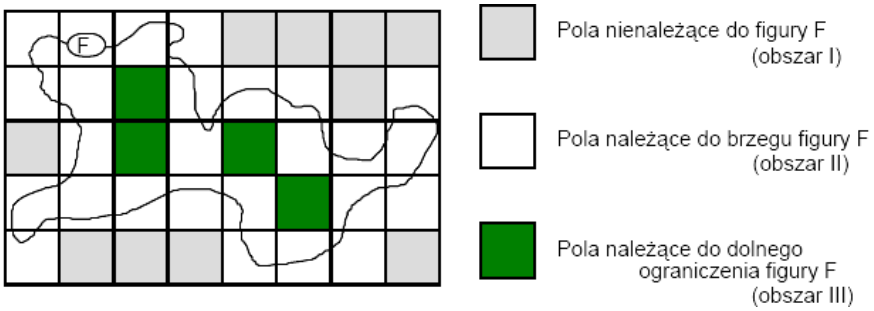
W zależności od liczności elementów w przybliżeniach górnych i dolnych określa się kilka typów zbiorów przybliżonych. Kategorie te można przedstawiać w sposób graficzny (rys. 5.2-5.6). Figura F oznacza hipotetyczny poszukiwany zbiór, małe kwadraty to zbiory elementarne, wewnątrz których znajdują się obiekty *x* (jeden kwadrat to cała klasa abstrakcji, nie określa się tu ile obiektów jest wewnątrz).

X jest zbiorem klasycznym, definiowalnym, gdy $\underline{B}(X) \neq \varnothing$ i $\overline{B}(X) = \underline{B}(X)$, $BND_B(X) = \varnothing$



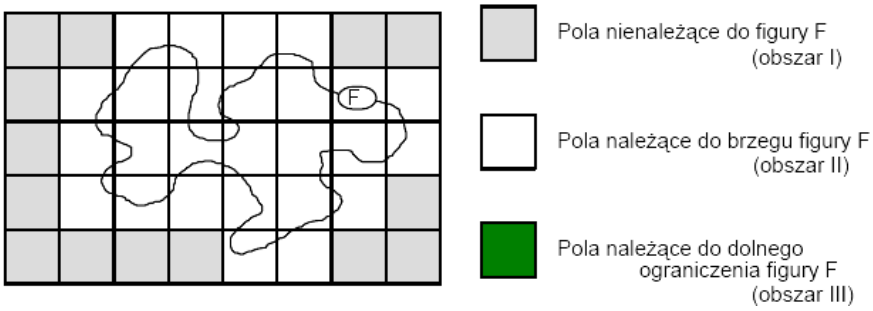
Rys. 5.2. Przykład zbioru klasycznego

X jest w przybliżeniu *B-definiowalny*, gdy: $\underline{B}(X) \neq \varnothing$ i $\overline{B}(X) \neq U$



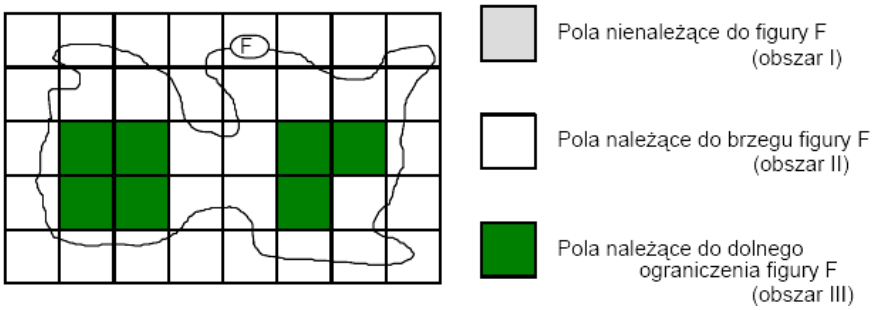
Rys. 5.3. Przykład zbioru *B*-definiowalnego

X jest wewnętrznie *B*-***nie****definiowalny*, gdy: $\underline{B}(X) = \emptyset$ i $\overline{B}(X) \neq U$



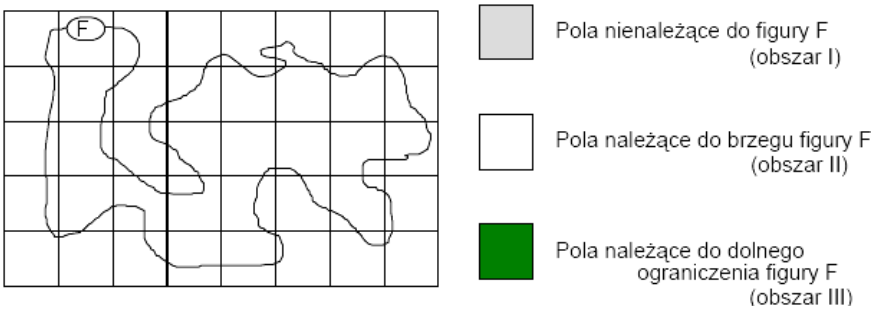
Rys. 5.4. Przykład zbioru wewnętrznie *B*-niedefiniowalnego

X jest zewnętrznie *B*-***nie****definiowalny*, gdy: $\underline{B}(X) \neq \emptyset$ i $\overline{B}(X) = U$



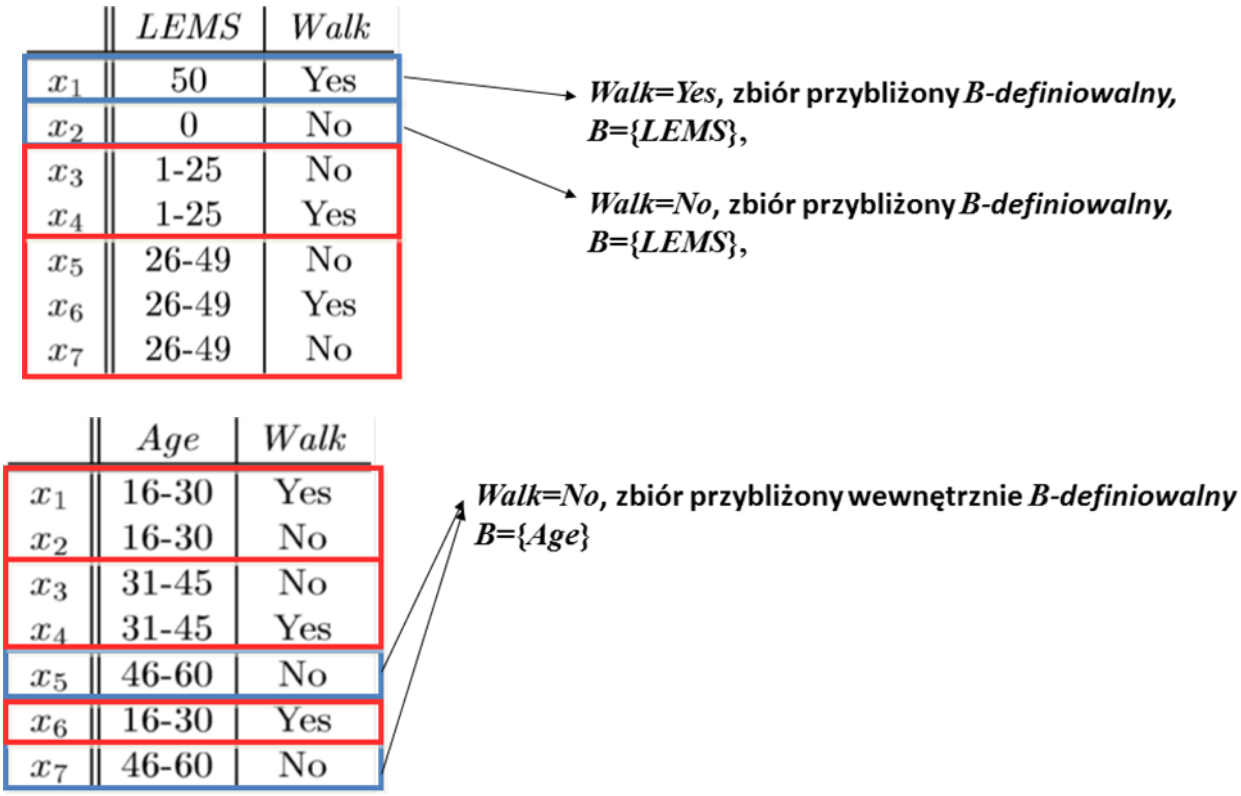
Rys. 5.5. Przykład zbioru zewnętrznie *B*-niedefiniowalnego

X jest całkowicie *B*-***nie****definiowalny*, gdy: $\underline{B}(X) = \emptyset$ i $\overline{B}(X) = U$



Rys. 5.6. Przykład zbioru całkowicie *B*-niedefiniowalnego

Dla przypadku medycznego można poszukać różnych typów zbiorów przybliżonych w podanym systemie decyzyjnym (rys. 5.7).



Rys. 5.7. Przykłady zbiorów przybliżonych

5.8 Redukty

Różne podzbiory atrybutów $B_1 \subseteq A$ i $B_2 \subseteq A$ mogą prowadzić do identycznych podziałów uniwersum, czyli takich, że $IND_{B1}(X)=IND_{B2}(X)$. Oznacza to, że zbiory elementarne uzyskane z tych podziałów są identyczne, a w konsekwencji także i aproksymacja jest równie precyzyjna.

Redukt B to taki podzbiór atrybutów, który ma minimalną ilość atrybutów, a ponadto $IND_B(X)= IND_A(X)$ (generuje taki sam podział jak cały zbiór atrybutów).

Wyznaczenie reduktu polega na pozostawieniu w podzbiorze tylko tych atrybutów, które zachowują relację równoważności, czyli nie zmieniają aproksymacji zbioru i na usunięciu pozostałych.

Zwykle dla danego systemu decyzyjnego istnieć może wiele reduktów. Czasami ze względów praktycznych nie korzysta się z reduktu najkrótszego (z najmniejszą liczbą atrybutów), tylko z takiego, którego implementacja jest np. najprostsza, a sposób pomiaru wartości atrybutu najmniej kosztowny, najszybszy, itd. Przykładowo w prostych przypadkach przeziębienia, pacjenta można pytać o samopoczucie (dobre, złe) i uzyskać te same wyniki, jak przy użyciu drogiego, czasochłonnego pomiaru temperatury, ciśnienia krwi, morfologii.

5.8.1 Wyznaczanie reduktów

Wyznaczanie reduktów jest problem NP-trudnym. Liczba możliwych reduktów wyraża się dwumianem $\binom{m}{\lfloor \frac{m}{2} \rfloor}$, gdzie m – liczba atrybutów, a $\lfloor \ \rfloor$ oznacza zaokrąglenie w dół do liczby całkowitej.

Proces wyznaczania reduktów uważany jest za „wąskie gardło” w systemach wnioskowania opartych na zbiorach przybliżonych. Często stosowane są metody genetyczne do wyznaczania reduktów dla dużych systemów decyzyjnych z dziesiątkami lub setkami atrybutów.

Z kolei w celu ręcznego wyznaczenia reduktu posłużyć się można macierzą rozróżnialności obiektów.

5.8.2 Macierz rozróżnialności

Macierz rozróżnialności w *i*-tym wierszu i *j*-tej kolumnie zawiera zbiór tych atrybutów, którymi różnią się obiekty x_i i x_j:

$$c_{ij}=\{a \in A \mid a(x_i) \neq a(x_j)\}, \qquad j,i=1,2...n \tag{5.11}$$

gdzie*: n* - liczba obiektów

Można zauważyć, że: $c_{ij}= c_{ji}$ (obiekt i-ty od j-tego odróżniają te same atrybuty co j-tego od i-tego).

5.8.3 Funkcja rozróżnialności

Analizując macierz rozróżnialności, tworzy się funkcję logiczną nazywaną funkcją rozróżnialności, będącą iloczynem logicznym wszystkich niepustych sum logicznych c_{ij}^*:

$$f_A(a_1^*, a_2^*,..., a_m^*)= \wedge\{\vee c_{ij}^* \mid c_{ij} \neq \varnothing\}, \qquad j,i=1,2...n \tag{5.12}$$

gdzie,

$$c_{ij}^*=\{a^* \mid a \in c_{ij}\} \tag{5.13}$$

oraz:

$$a_i^* = \{1 \text{ jeżeli } a_i \in c_{ij}; 0 \text{ w przeciwnym przypadku}\} \tag{5.14}$$

Wykorzystanie funkcji rozróżnialności (5.12) zaprezentowane zostanie na przykładzie danych ośmiu kandydatów do pracy (tab. 5.5). W systemie decyzyjnym opisani są oni 4 atrybutami (wykształcenie, doświadczenie, znajomość francuskiego, referencje) i przydzielono im decyzję o przyjęciu lub odrzuceniu kandydatury.

Tab. 5.5. Dane opisujące kandydatów do pracy

	Diploma	*Experience*	*French*	*Reference*	*Decision*
x_1	MBA	Medium	Yes	Excellent	Accept
x_2	MBA	Low	Yes	Neutral	Reject
x_3	MCE	Low	Yes	Good	Reject
x_4	MSc	High	Yes	Neutral	Accept
x_5	MSc	Medium	Yes	Neutral	Reject
x_6	MSc	High	Yes	Excellent	Accept
x_7	MBA	High	No	Good	Accept
x_8	MCE	Low	No	Excellent	Reject

Przykładowo c_{12}^* wg wzoru (5.13) jest listą atrybutów odróżniających obiekt x_1 od x_2, czyli c_{12}^* = {*experience*, *reference*}.

W celu wyznaczenia reduktu wylicza się dla jakich wartości atrybutów a_i^* wartość funkcji f_A wynosi 1. Realizowane jest to poprzez upraszczanie funkcji rozróżnialności. Przykładowo widoczne jest, że pary obiektów {x_1, x_2}, {x_1, x_3}, {x_1, x_4} i {x_1, x_5} rozróżniane są przez następujące atrybuty:

$$(e \vee r) \wedge (d \vee e \vee r) \wedge (d \vee r) \wedge (d \vee e \vee f) \tag{5.15}$$

Skoro poszukiwane są wartości czyniące funkcję f_A prawdziwą (wartość 1), to w powyższym iloczynie każdy z czynników (nawiasów) musi mieć wartość logiczną 1. Zakłada się na początek, że pierwszy z atrybutów *e* (w skrócie *experience*) ma wartość *e*=1. Powoduje to, że wśród powyższych sum prawdziwe będą pierwsza, druga i czwarta (dla ich prawdziwości nie będą już istotne wartości *r*, *d*, czy *f*). Z kolei, aby prawdziwa była suma trzecia wymagane jest aby np. *r*=1, co da wartość całego iloczynu f_A równego 1. W ten sposób uzyskano pierwszy redukt:

$$e \wedge r \tag{5.16}$$

Wyłącznie wiedza o doświadczeniu (*experience*) i referencjach kandydata wystarcza do określenia decyzji.

W postępowaniu powyższym można było dla prawdziwości trzeciej sumy w iloczynie (5.15) założyć, że *d*=1, wówczas reduktem będzie:

$$e \wedge d \tag{5.17}$$

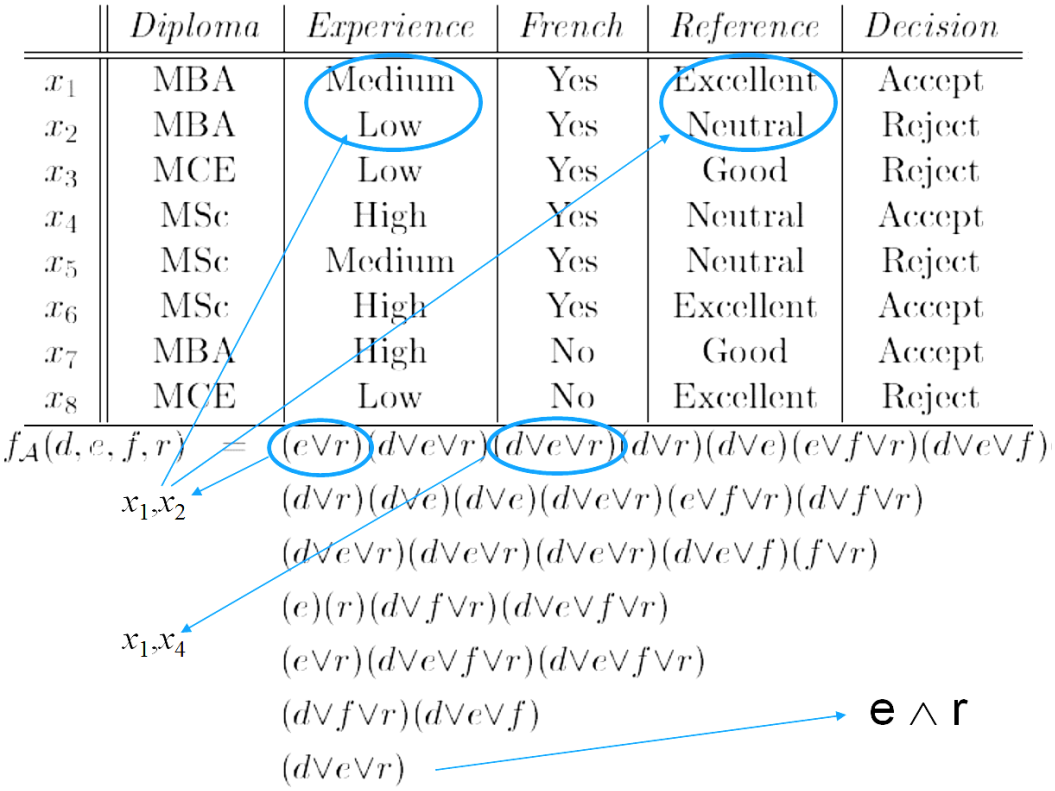
Jeżeli zaś założy się, ze w pierwszej sumie *e* jest nieprawdziwe (równe 0), lub nieznane, to *r* musi być równe 1 dla prawdziwości funkcji f_A = 1. Wówczas pierwsza, druga i trzecia suma będą równe 1, a dla prawdziwości czwartej zakładać trzeba będzie prawdziwość *d* lub *e* lub *f*. Takie założenia generują redukty:

$$r \wedge d$$

$$r \wedge e \text{ (identyczny z } e \wedge r) \tag{5.18}$$

$$r \wedge f$$

Aby wyznaczyć redukt dla całego systemu decyzyjnego z ośmioma kandydatami, należy uwzględnić atrybuty rozróżniające każdy z każdym (rys. 5.8).



Rys. 5.8. Funkcja rozróżnialności i wynik jej uproszczenia. Składniki funkcji zawierają atrybuty, które kolejno odróżniają każdy obiekt od każdego innego

Przedstawione postępowanie jest łatwo algorytmizowane i dostępne jest w oprogramowaniu wspierającym podejmowanie decyzji. Ponieważ wymaga ono sprawdzenia wszystkich możliwych par obiektów, nazywane jest **algorytmem wyczerpującym**.

Inne podejścia do wyznaczania reduktów mogą opierać się na losowym wyborze atrybutów i sprawdzaniu jakości rozróżniania obiektów, uzyskiwanej z ich użyciem. Ten początkowy, losowy wybór poprawiany może być w sposób deterministyczny (np. kolejno dodawać/podmieniać atrybuty na inne, aby znaleźć lepszy redukt), lub losowy (np. metodą genetyczną losowania wielu reduktów, krzyżowania ich ze sobą i mutacji). Opis tych podejść dostępny jest w literaturze uzupełniającej [5].

5.9 Wykorzystanie reguł decyzyjnych w klasyfikacji

Po wyznaczeniu reduktu (reduktów) możliwe jest wygenerowanie z danych dostępnych w systemie decyzyjnym reguł logicznych o postaci **JEŻELI** … **I** … **TO** …. Przykładowa uproszczona funkcja logiczna dla zagadnienia rekrutacji do pracy posiada dwa redukty: $f_A(d,e,f,r)=(d\wedge e)\vee(e\wedge r)$.

Dla każdego z nich należy odczytywać z danych z systemu decyzyjnego (z tabeli) występujące tam wartości atrybutów i odpowiadające im decyzje, tworząc w ten sposób zestaw reguł, opisujących dostępne przypadki, np. **JEŻELI** *Diploma*=MBA **I** *Experience*=Medium **TO** *Decision*=Accept.

Reguł utworzyć można tyle, ile wynosi liczba obiektów rozróżnialnych (nie tożsamych) pomnożona przez liczbę reduktów. Nie każdą z reguł trzeba jednak wykorzystywać czy implementować w procesie decyzyjnym.

5.9.1 Klasyfikacja

Proces klasyfikacji – nadawania decyzji – nowemu obiektowi przebiega w kilku typowych krokach:

- Obliczenie atrybutów nowego obiektu, czyli pomiar jego cech, tych, które są istotne dla decyzji, powiązanych z treścią reduktów,
- Poszukiwanie reguł pasujących do wartości atrybutów,
- Jeżeli **brak pasujących reguł**, wynikiem jest **najczęstsza** decyzja w systemie decyzyjnym, lub decyzja **najmniej kosztowna**, np. czasem mniejsze może być ryzyko odrzucenia dobrego kandydata do pracy, lub mniejsze może być ryzyko przyjęcia niepewnego kandydata na okres próbny,
- Jeżeli **pasuje wiele reguł** mogą one wskazywać na różne decyzje, wówczas przeprowadzane jest głosowanie – wybierana jest odpowiedź pojawiająca się najczęściej.

W procesie klasyfikacji napotkać można kilka przypadków:

- nowy obiekt pasuje dokładnie do jednej **deterministycznej** reguły - sytuacja najbardziej pożądana - uzyskuje się wiadomość, iż obiekt należy do zadanej klasy, do **dolnego przybliżenia zbioru**,
- nowy obiekt pasuje dokładnie do jednej, **niedeterministycznej** reguły - sytuacja ta jest nadal pozytywna, choć tym razem uzyskuje się jedynie wiadomość, iż obiekt prawdopodobnie należy do zbioru - a wiec, że należy do jego **górnego przybliżenia**,
- nowy obiekt pasuje do **więcej niż jednej** reguły - kilka potencjalnych przynależności obiektu, a więc decyzja nie jest jednoznaczna; zazwyczaj w takim przypadku stosuje się dodatkowe kryteria dla oceny, do której z klas z największym prawdopodobieństwem należy obiekt. Należy zauważyć, że problem ten nie występowałby, gdyby wszystkie klasy obiektów były parami rozłączne, lecz jest to warunek trudny do spełnienia i jest rzadko stosowany.

5.9.2 Aktualizacja systemu wnioskującego

W wyniku napotykania nowych przypadków i konfrontowania decyzji systemu z decyzją eksperta, kiedy to „życie weryfikuje” poprawność wcześniejszych decyzji, system decyzyjny i bazę reguł można aktualizować. Dodanie nowego przypadku jest szczególnie wskazane, jeżeli:

- jest on opisany wartościami atrybutów, które wcześniej w systemie nie występowały,
- był niewłaściwie sklasyfikowany a jego dodanie wygeneruje nową regułę, poprawiającą decyzję,
- dodanie obiektu i powtórne generowanie reduktu ujawni nowe, przydatne atrybuty.

5.9.3 Jakość decyzji

Zwykle dysponuje się tzw. danymi testowymi, dla których znana jest decyzja a sprawdzane jest, czy odpowiedź zwracana przez system jest z nią identyczna. Wówczas można wyznaczyć jakość decyzji i dokonać oceny systemu. W tym celu wykorzystuje się pojęcie obszaru B-pozytywnego.

5.9.4 Klasa decyzyjna i obszar B-pozytywny

Wprowadzony zostaje następujący formalizm.

r – liczba decyzji w systemie decyzyjnym,

v^i_d – wartość decyzji dla i-tego obiektu, np. *Yes*, *No, Accept, Reject,...* $i=1,...,r$

$V_d=\{v^1_d, ... v^{r(d)}_d\}$ – zbiór wszystkich wartości decyzji w systemie decyzyjnym, np. {*Yes*,*No*}

$X^k_A = \{x \in U \mid d(x)= v^k_d\}$ – k-ta klasa decyzyjna, $k=1,...,r$, czyli zbiór tych wszystkich obiektów x, dla których decyzja $d(x)$ ma wartość k-tą v^k_d.

Wówczas analizując wszystkie r klas otrzymuje się zbiór klas decyzyjnych $CLASS(d)=\{X^1_A,...X^r_A\}$. Decyzja d determinuje podział uniwersum na r zbiorów, np. $U = X^{Yes} \cup X^{No}$, oznacza podział na obiekty o decyzji *Yes* i decyzji *No*. W idealnym przypadku są to decyzje jednoznaczne, a w przypadku zbioru przybliżonego o decyzjach niedeterministycznych taka suma może być różna od U.

W jednoznaczny sposób jakość decyzji określa się poprzez wyliczenie i posumowanie przybliżeń dolnych wszystkich klas decyzyjnych X^i:

$$POS_B(d) = \underline{B}X^1 \cup \underline{B}X^2 \cup ... \cup \underline{B}X^r \tag{5.19}$$

co nazywane jest obszarem B-pozytywnym. System decyzyjny jest **deterministyczny** (zgodny), jeżeli $POS_B(d) = U$ (do dolnych przybliżeń należą wszystkie obiekty uniwersum, inaczej: dla każdego obiektu uniwersum istnieje klasa decyzyjna, w której dolnym przybliżeniu jest ten obiekt) w przeciwnym wypadku jest **niedeterministyczny**.

5.10 Zbiory przybliżone o zmiennej precyzji

W typowym podejściu do zbiorów przybliżonych wyróżniane są obszary: B-dolny (obiekty z pewnością należące do zbioru) i graniczny (obiekty o decyzji niepewnej). B-górny obszar to suma tych dwóch. Jednak w praktyce wystąpić może przypadek, kiedy niewielka niepewność jest akceptowana. Obiekty, których klasa abstrakcji jest liczna, a tylko jeden z przypadków jest klasyfikowany niewłaściwie, można zaliczyć wówczas do przybliżenia B-dolnego a nie do obszaru granicznego, łagodząc w ten sposób kryterium przynależności do zbioru. Takie podejście nazywane jest zmianą precyzji. W tym celu wykorzystuje się przybliżoną przynależność obiektu do zbioru.

5.10.1 Przybliżona przynależność do zbioru

Niech μ^X_B: $U \rightarrow$ [0;1], funkcja, która obiektom z uniwersum przypisuje wartości od zera do jeden, w następujący sposób:

$$\mu_B^X(x) = \frac{|[x]_B \cap X|}{|[x]_B|} \tag{5.20}$$

W mianowniku (5.20) wyznaczana jest moc klasy abstrakcji – ile elementów jest identycznych z *x*. W liczniku sprawdzane jest ile spośród tych elementów ma decyzję zgodną z poszukiwanym zbiorem *X*. Jeżeli cała klasa abstrakcji jest w B-dolnym przybliżeniu *X*, to μ^X_B=1. Taka funkcja zwracająca wartości z przedziału [0;1] nazywana jest funkcją przynależności, w tym przypadku **przybliżonej przynależności**.

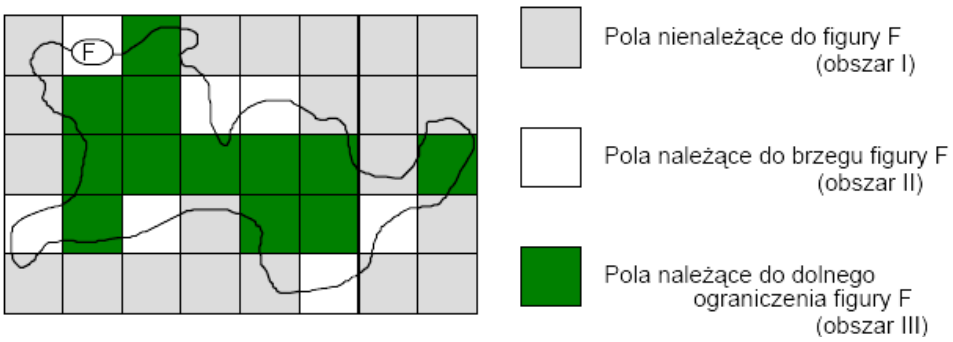
5.10.2 Zbiór przybliżony o zmiennej precyzji

Korzystając z wartości funkcji μ^X_B dla obiektów *x*, wprowadza się pojecie przybliżenia o **zmiennej precyzji**, zależnej od parametru π (dla π=1, przypadek klasyczny):

$$\underline{B}_\pi X=\{x \mid \mu^X_B(x) \geq \pi\} \quad \text{oraz} \quad \overline{B}_\pi X=\{x \mid \mu^X_B(x) \geq 1-\pi\} \tag{5.21}$$

Graficzna interpretacja przedstawiona jest poniżej (rys. 5.9). Jeżeli klasa abstrakcji (jeden z kwadratów siatki) ma pewną istotną **część powierzchni** wewnątrz figury F to zostaje przydzielony do B-dolnego przybliżenia. Wartość precyzji π kojarzyć należy z pomiarem tego pola powierzchni.

Niech przykładowo π=0,66, wówczas jeżeli $\mu^X_B(x) \geq$ 0,66 to *x* i jego klasa abstrakcji są w B-dolnym przybliżeniu; jeżeli $\mu^X_B(x) \geq$ 0,34 to *x* i jego klasa abstrakcji są w B-górnym przybliżeniu, $\mu^X_B(x) <$ 0,34, to *x* jest poza zbiorem. Oczywiście gdy 0,66 > $\mu^X_B(x) \geq$ 0,34 to *x* jest w obszarze granicznym (kolor biały).



Rys. 5.9. Graficzna reprezentacja zbioru o zmniejszonej precyzji przybliżenia

Takie postępowanie pozwala na złagodzenie kryterium przydziału do B-dolnego przybliżenia. Dla rzeczywistych danych wystąpić mogą czasem błędne klasyfikacje identycznych obiektów (klasa abstrakcji nie jest w pełni w zbiorze o danej decyzji), ale jeżeli jest to dość rzadki przypadek, to odpowiednia wartość precyzji π pozwoli stworzyć system o regułach deterministycznych (obszar B-pozytywny równy całemu uniwersum).

5.11 Dyskretyzacja parametrów

Obiekty w systemie decyzyjnym mogą w ogólności być opisane atrybutami liczbowymi o dowolnej dokładności z ciągłej dziedziny, co przedstawiono na poniższym przykładzie (rys. 5.10). Nie ma pewności, że nieznane przypadki w przyszłości będą miały podobne wartości – czy cecha *a* może mieć wartość mniejszą od 0,8 lub większą od 1,6?; czy dokładność do jednego miejsca po przecinku jest tu wystarczająca?; jakie reguły należy wówczas utworzyć dla brakujących wartości?; czy cecha *b* dla drugiego obiektu tylko w wyniku błędu pomiaru jest wyrażona ułamkiem, może wskazane jest zaokrąglenie do liczy całkowitej? Te i podobne wątpliwości można zaniedbać jeżeli dokona się prawidłowej dyskretyzacji parametrów – zamiany wartości ciągłych na całkowite (lub na etykiety słowne, nazwy przedziałów). W poniższym przykładzie zauważyć można, że u_3 i u_5 stały się tożsame po dyskretyzacji, jednak mają tę samą decyzję *d*, podobnie jak u_4 i u_7. W wyniku dyskretyzacji utracono możliwość rozróżniania obiektów między sobą, jednak bez obniżenia miary jakości aproksymacji zbiorów.

$\mathbf{A}$	a	b	d
u_1	0.8	2	1
u_2	1	0.5	0
u_3	1.3	3	0
u_4	1.4	1	1
u_5	1.4	2	0
u_6	1.6	3	1
u_7	1.3	1	1

$\mathbf{A^P}$	a^P	b^P	d
u_1	0	2	1
u_2	1	0	0
u_3	1	2	0
u_4	1	1	1
u_5	1	2	0
u_6	2	2	1
u_7	1	1	1

Rys. 5.10. System decyzyjny przed i po dyskretyzacji

Zasadę dyskretyzacji (użyteczną m.in. na etapie wstępnego przetwarzania danych wejściowych przez algorytm) zapisuje się w postaci zbioru cięć (**P** oznacza partycjonowanie), np. **P** = {(*a*; 0,9); (*a*; 1,5); (*b*; 0,75); (*b*; 1,5)}.

Taki zbiór cięć odczytuje się następująco: dziedzinę atrybutu *a*, należy podzielić w punkcie 0,9 i w punkcie 1,5, uzyskując trzy przedziały (–∞; 0,9), <0,9; 1,5), <1,5; ∞). W powyższym przypadku przedziały te zostały nazwane liczbami całkowitymi 0, 1, 2 (rys. 5.10), jednak można także użyć etykiet słownych, lub innej numeracji.

Powyższe cięcia uzyskano w następujący sposób:

Krok 1. dla atrybutu poszeregowano wartości i utworzono z nich przedziały:

<0.8;1); <1; 1.3); <1.3; 1.4); <1.4; 1.6) dla *a*

<0.5; 1); <1; 2); <2; 3) dla *b*

Krok 2. środki przedziałów przyjmowane są za miejsca cięć:

(*a*; 0.9); (*a*; 1.15); (*a*; 1.35); (*a*; 1.5);

(*b*; 0.75); (*b*; 1.5); (*b*; 2.5)

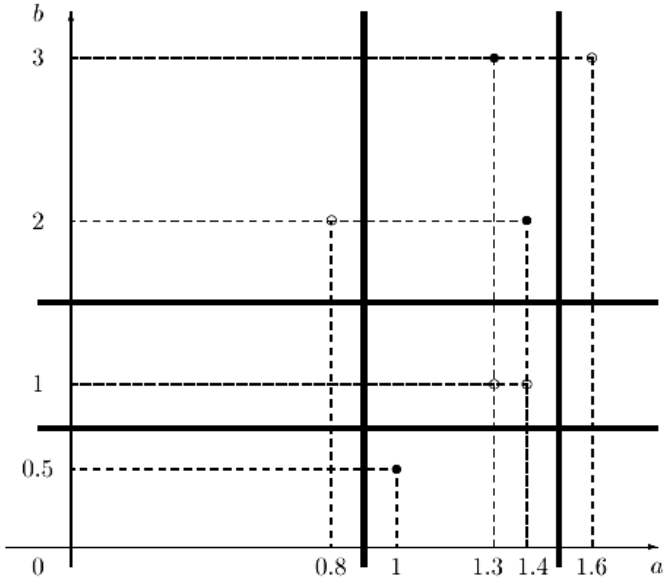
Krok 3. usunięto te cięcia, które nie prowadzą do rozróżnienia choć jednej pary obiektów:

(*a*; 0.9); ~~(*a*; 1.15);~~ ~~(*a*; 1.35);~~ (*a*; 1.5);

(*b*; 0.75); (*b*; 1.5); ~~(*b*; 2.5);~~

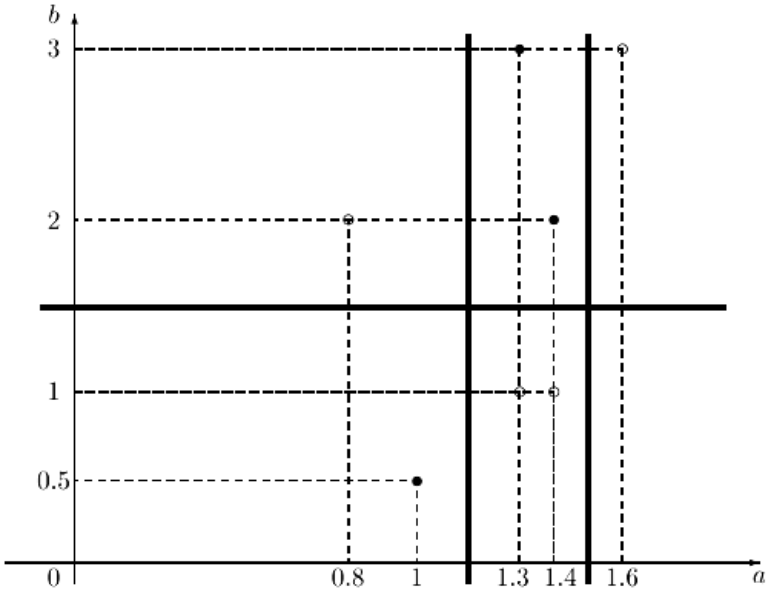
Uzyskane cięcia tworzą zbiór **P** = {(*a*; 0.9); (*a*; 1.5); (*b*; 0.75); (*b*; 1.5)}.

Graficzna interpretacja przedstawiona jest poniżej (rys. 5.11), punkty czarne (koła) to obiekty o decyzji „1”, punkty białe (okręgi) – o decyzji „0”. Grube linie to cięcia. Usunięte cięcie (*b*; 2.5) nie prowadziło do rozróżniania kół od okręgów, nie było potrzebne.



Rys. 5.11. Przykładowa dyskretyzacja dziedzin dwóch atrybutów

Należy zwrócić uwagę, że w wyniku takiego podziału uzyskuje się 9 obszarów, co dla innych cięć wyliczane jest jako (*n*+1)(*k*+1), gdzie *n* i *k* to liczba cięć dla atrybutów *a* i *b*. Niestety taki podział nie jest optymalny – występują **obszary bez decyzji** (w tym przykładzie cztery obszary bez obiektów). Nowy obiekt o wartościach atrybutów, lokujących go w jednym z tych przedziałów nie zostanie sklasyfikowany. Zamiast powyższego podejścia stosuje się często algorytm MD-Heuristics [4], wyznaczający dyskretyzację o mniejszej liczbie cięć i obszarów niepewnych (rys. 5.12).

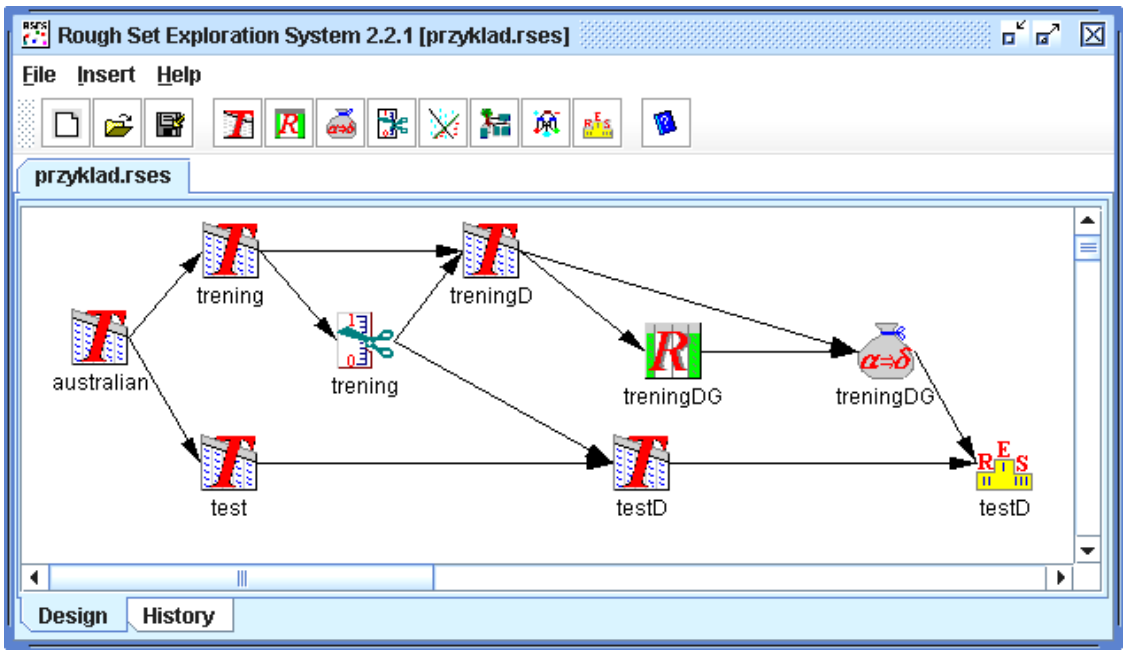


Rys. 5.12. Dyskretyzacja o mniejszej liczbie cięć i mniejszej liczbie obszarów bez decyzji

Dla tych samych danych jest mniej obszarów i cięć (łatwiejsza implementacja) oraz liczba obszarów pustych (bez decyzji), zmalała do jednego.

5.12 System decyzyjny – RSES

Rough Set Exploration System to aplikacja autorstwa naukowców z Uniwersytetu Warszawskiego (http://logic.mimuw.edu.pl/~rses/). Udostępnia wszystkie najważniejsze operacje na danych, dotyczące teorii zbiorów przybliżonych i jej zastosowań. Praca z programem polega na budowaniu grafu przetwarzania danych (rys. 5.13).



Rys. 5.13. Graf przetwarzania danych: T to tabele, zbiór cięć oznacza ikona z nożyczkami, R to redukty, reguły zapisane są w strukturze dostępnej pod ikoną worka, wyniki pod ikoną podium

Standardowy sposób budowania klasyfikatora zrealizować można następująco:

- wczytać tabelę z danymi
- dokonać jej podziału na część trenującą i testującą (z menu kontekstowego – prawy przycisk myszy na ikonie tabeli danych)
- na części treningowej wykonać generowanie cięć (Generate Cuts z menu kontekstowego)
- na części treningowej wykonać dyskretyzację (Discretize z menu kontekstowego), wskazując w oknie dialogowym źródło danych i źródło cięć (w grafie symbolizowane jest to początkami strzałek)
- na danych dyskretnych przeprowadzić wyliczanie reduktu
- na redukcie wykonać polecenie generowania reguł, jako źródło podając zbiór danych treningowych dyskretyzowanych
- na danych testujących wykonać dyskretyzację tym samym zbiorem cięć
- na danych testujących dyskretyzowanych wykonać klasyfikację z użyciem wygenerowanych reguł.

Aplikacja RSES dostarcza wygodnych graficznych narzędzi do przeglądania danych, cięć, reguł i wyników (rys. 5.14-5.19).

Table: trening

A8	A9	A10	A11	A12	A13	A14	CLASS
0	0	0	1	2	100	1213	0
0	0	0	0	2	160	1	0
0	0	0	1	2	280	1	0
1	1	11	1	2	0	1	1
1	1	14	0	2	60	159	1
1	1	6	0	2	43	561	1
0	0	0	0	2	176	538	0
1	1	3	1	2	100	51	0
1	1	4	1	2	253	858	1
1	1	6	1	2	470	1	1
1	1	6	1	2	0	1001	1

Rys. 5.14. Przykładowa tabela z danymi trenującymi. Stosować można dane binarne, liczby naturalne, rzeczywiste oraz etykiety słowne

Cut set: trening

(1-14)	Attribute	Size	Description
1	A1	0	*
2	A2	8	21.04; 21.21; 23.04; 24.0; 27.915; 31.125; 37.25; 45.58
3	A3	5	0.555; 2.23; 4.48; 6.02; 8.54
4	A4	0	*
5	A5	0	*
6	A6	0	*
7	A7	3	0.1875; 1.02; 3.1675
8	A8	0	*
9	A9	0	*
10	A10	2	0.5; 2.5
11	A11	0	*
12	A12	0	*
13	A13	4	75.5; 111.0; 172.0; 296.0
14	A14	2	13.5; 309.0

Rys. 5.15. Przykładowy zbiór cięć: gwiazdka oznacza, że dany atrybut nie będzie dyskretyzowany, gdyż obecnie jego wartości mają odpowiednią postać, np. są binarne, lub odpowiednio mało różnych wartości

Table: treningD

A8	A9	A10	A11	A12	A13	A14	CLASS
0	0	"(-Inf,0.5)"	1	2	"(75.5,111.0...	"(309.0,Inf)"	0
0	0	"(-Inf,0.5)"	0	2	"(111.0,172....	"(-Inf,13.5)"	0
0	0	"(-Inf,0.5)"	1	2	"(172.0,296....	"(-Inf,13.5)"	0
1	1	"(2.5,Inf)"	1	2	"(-Inf,75.5)"	"(-Inf,13.5)"	1
1	1	"(2.5,Inf)"	0	2	"(-Inf,75.5)"	"(13.5,309.0...	1
1	1	"(2.5,Inf)"	0	2	"(-Inf,75.5)"	"(309.0,Inf)"	1
0	0	"(-Inf,0.5)"	0	2	"(172.0,296....	"(309.0,Inf)"	0
1	1	"(2.5,Inf)"	1	2	"(75.5,111.0...	"(13.5,309.0...	0
1	1	"(2.5,Inf)"	1	2	"(172.0,296....	"(309.0,Inf)"	1
1	1	"(2.5,Inf)"	1	2	"(296.0,Inf)"	"(-Inf,13.5)"	1
1	1	"(2.5,Inf)"	1	2	"(-Inf,75.5)"	"(309.0,Inf)"	1
0	0	"(-Inf,0.5)"	0	2	"(-Inf,75.5)"	"(309.0,Inf)"	1
0	0	"(-Inf,0.5)"	0	2	"(111.0,172....	"(-Inf,13.5)"	0

Rys. 5.16. Tabela z danymi po dyskretyzacji. Zauważyć można zastąpienie liczb naturalnych lub rzeczywistych nazwami podprzedziałów, np. „(-Inf, 0.5)” to liczby od $-\infty$ do 0,5

Reduct set: treningDG

(1-10)	Size	Pos.Reg.	SC	Reducts
1	6	1	1	{A2, A3, A7, A10, A13, A14}
2	7	1	1	{A2, A3, A5, A7, A8, A10, A13}
3	7	1	1	{A1, A2, A3, A4, A5, A7, A13}
4	7	1	1	{A1, A2, A3, A5, A7, A8, A13}
5	7	1	1	{A1, A2, A3, A5, A7, A9, A13}
6	7	1	1	{A1, A2, A3, A5, A7, A10, A13}
7	7	1	1	{A2, A3, A4, A5, A7, A10, A13}
8	7	1	1	{A2, A3, A5, A10, A12, A13, A14}
9	7	1	1	{A2, A3, A5, A8, A10, A13, A14}
10	7	1	1	{A2, A3, A5, A8, A9, A13, A14}

Rys. 5.17. Lista 10 przykładowych reduktów. *Size* oznacza liczbę atrybutów, *Pos.Reg*. to region B-pozytywny

Rule set: treningDG

...	Match	Decision rules
1	1	(A2="(21.21,23.04)")&(A3="(8.54,Inf)")&(A7="(1.02,3.1675)")&(A10="(-Inf,0.5)")&(A13="(75.5,111.0)")&(A14="(309.0,Inf)")=>(CLASS={0[1]})
2	1	(A2="(21.21,23.04)")&(A3="(6.02,8.54)")&(A7="(-Inf,0.1875)")&(A10="(-Inf,0.5)")&(A13="(111.0,172.0)")&(A14="(-Inf,13.5)")=>(CLASS={0[1]})
3	2	(A2="(27.915,31.125)")&(A3="(0.555,2.23)")&(A7="(1.02,3.1675)")&(A10="(-Inf,0.5)")&(A13="(172.0,296.0)")&(A14="(-Inf,13.5)")=>(CLASS={0[2]})
4	1	(A2="(21.21,23.04)")&(A3="(8.54,Inf)")&(A7="(-Inf,0.1875)")&(A10="(2.5,Inf)")&(A13="(-Inf,75.5)")&(A14="(-Inf,13.5)")=>(CLASS={1[1]})
5	1	(A2="(-Inf,21.04)")&(A3="(6.02,8.54)")&(A7="(1.02,3.1675)")&(A10="(2.5,Inf)")&(A13="(-Inf,75.5)")&(A14="(13.5,309.0)")=>(CLASS={1[1]})
6	1	(A2="(45.58,Inf)")&(A3="(2.23,4.48)")&(A7="(1.02,3.1675)")&(A10="(2.5,Inf)")&(A13="(-Inf,75.5)")&(A14="(309.0,Inf)")=>(CLASS={1[1]})
7	1	(A2="(24.0,27.915)")&(A3="(0.555,2.23)")&(A7="(1.02,3.1675)")&(A10="(-Inf,0.5)")&(A13="(172.0,296.0)")&(A14="(309.0,Inf)")=>(CLASS={0[1]})
8	1	(A2="(45.58,Inf)")&(A3="(6.02,8.54)")&(A7="(3.1675,Inf)")&(A10="(2.5,Inf)")&(A13="(75.5,111.0)")&(A14="(13.5,309.0)")=>(CLASS={0[1]})
9	1	(A2="(31.125,37.25)")&(A3="(0.555,2.23)")&(A7="(3.1675,Inf)")&(A10="(2.5,Inf)")&(A13="(172.0,296.0)")&(A14="(309.0,Inf)")=>(CLASS={1[1]})
10	2	(A2="(37.25,45.58)")&(A3="(4.48,6.02)")&(A7="(3.1675,Inf)")&(A10="(2.5,Inf)")&(A13="(296.0,Inf)")&(A14="(-Inf,13.5)")=>(CLASS={1[2]})
11	1	(A2="(31.125,37.25)")&(A3="(4.48,6.02)")&(A7="(3.1675,Inf)")&(A10="(2.5,Inf)")&(A13="(-Inf,75.5)")&(A14="(309.0,Inf)")=>(CLASS={1[1]})
12	1	(A2="(45.58,Inf)")&(A3="(6.02,8.54)")&(A7="(-Inf,0.1875)")&(A10="(-Inf,0.5)")&(A13="(-Inf,75.5)")&(A14="(309.0,Inf)")=>(CLASS={1[1]})
13	1	(A2="(-Inf,21.04)")&(A3="(0.555,2.23)")&(A7="(-Inf,0.1875)")&(A10="(-Inf,0.5)")&(A13="(111.0,172.0)")&(A14="(-Inf,13.5)")=>(CLASS={0[1]})
14	1	(A2="(27.915,31.125)")&(A3="(0.555,2.23)")&(A7="(-Inf,0.1875)")&(A10="(-Inf,0.5)")&(A13="(172.0,296.0)")&(A14="(309.0,Inf)")=>(CLASS={0[1]})
15	1	(A2="(-Inf,21.04)")&(A3="(0.555,2.23)")&(A7="(0.1875,1.02)")&(A10="(-Inf,0.5)")&(A13="(111.0,172.0)")&(A14="(-Inf,13.5)")=>(CLASS={0[1]})
16	4	(A2="(37.25,45.58)")&(A3="(0.555,2.23)")&(A7="(-Inf,0.1875)")&(A10="(-Inf,0.5)")&(A13="(111.0,172.0)")&(A14="(-Inf,13.5)")=>(CLASS={0[4]})
17	1	(A2="(37.25,45.58)")&(A3="(0.555,2.23)")&(A7="(0.1875,1.02)")&(A10="(-Inf,0.5)")&(A13="(111.0,172.0)")&(A14="(-Inf,13.5)")=>(CLASS={0[1]})
18	1	(A2="(-Inf,21.04)")&(A3="(8.54,Inf)")&(A7="(0.1875,1.02)")&(A10="(-Inf,0.5)")&(A13="(75.5,111.0)")&(A14="(309.0,Inf)")=>(CLASS={0[1]})
19	1	(A2="(31.125,37.25)")&(A3="(0.555,2.23)")&(A7="(3.1675,Inf)")&(A10="(2.5,Inf)")&(A13="(-Inf,75.5)")&(A14="(-Inf,13.5)")=>(CLASS={1[1]})
20	2	(A2="(24.24,23.04)")&(A3="(-Inf,0.555)")&(A7="(-Inf,0.1875)")&(A10="(-Inf,0.5)")&(A13="(172.0,296.0)")&(A14="(13.5,309.0)")=>(CLASS={0[2]})

Rys. 5.18. Treść przykładowych reguł. *Match* oznacza ile obiektów ze zbioru treningowego pasuje do danej reguły

Results of experiments by train&test method: testD

		Predicted				
		0	1	No. of obj.	Accuracy	Coverage
Actual	0	23	10	99	0.697	0.333
	1	4	18	74	0.818	0.297
	True positive rate	0.85	0.64			

Total number of tested objects: 173
Total accuracy: 0.745
Total coverage: 0.318

Rys. 5.19. Wynik klasyfikacji przedstawiony w postaci tzw. macierzy pomyłek. *Actual* to wartości właściwe i oczekiwane, które były dostępne w tabeli testowej, *Predicted* to odpowiedź klasyfikatora

5.13 Zbiory przybliżone w obliczeniach granularnych

W końcowej części niniejszego rozdziału warto by jeszcze wspomnieć o rozwijanych od wielu lat metodach obliczeń granularnych [6-17], które bardzo istotnie zaznaczyły się w tematyce zbiorów

przybliżonych. Intuicyjnie można określić, że granulacja przestrzeni badanych obiektów (uniwersum) może być wynikiem albo celowych zabiegów (np. parametryzacja, dyskretyzacja, cyfrowa analiza sygnałów, itd., albo skutkiem naturalnych ograniczeń w zakresie percepcji, dokładności pomiarów i gromadzenia danych o obiektach. W ujęciu Zadeha granula informacyjna oznacza skupienie obiektów zebranych ze względu na nierozróżnialność, podobieństwo lub funkcjonowanie (funkcjonalność). W ujęciu zaproponowanym przez Z. Pawlaka granulacja przestrzeni generowana jest przez nierozróżnialność obiektów z uwagi na rozważane atrybuty i może być modelowania jako pewna relacja równoważności.

Do realizacji obliczeń granularnych wykorzystuje się metody analizy przedziałowej, analizy skupień, zbiorów przybliżonych, zbiorów rozmytych i innych, a więc w ogólności dotyczy to wnioskowania w oparciu o dane z niekompletnym opisem obiektów i rozwiązywania problemów w warunkach niedoskonałej informacji.

5.14 Literatura

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

[2] Marek W., Pawlak Z., *Rough Sets and Information Systems*. Fundamenta Informaticae 17, pp. 105–115, 1984

[3] Pawlak Z., *Rough Sets – Theoretical Aspects of Reasoning About Data*. Kluwer Academic Publishers, Dordrecht, 1991

[4] Komorowski J., Polkowski L., Skowron A., *Rough Set: A Tutorial*. Rough fuzzy hybridization: A new trend in decision-making, pp. 3-98, Springer, 1999

[5] RSES 2.1. Rough Set Exploration System. Podręcznik Użytkownika. Publikacja elektroniczna http://logic.mimuw.edu.pl/~rses/RSES_doc.pdf. Warszawa, 2004

[6] Bargiela A., Pedrycz W. (eds.), *Human-Centric Information Processing Through Granular Modelling*, Springer -Verlag, Heidelberg, 2009

[7] Gacek A, Pedrycz W., *A characterization of electrocardiogram signals through optimal allocation of information granularity*, Journal Artificial Intelligence in Medicine, 54, 2, pp. 125-134, 2012

[8] Gomolińska A,, *Zbiory przybliżone w obliczeniach granularnych*, seminarium, Poznań 2011, http://idss.cs.put.poznan.pl/site/fileadmin/seminaria/2011/poznan11.pdf

[9] Lin T.Y., Yao Y.Y., Zadeh L.A. (eds.), *Data Mining, Rough Sets and Granular Computing*, Physica-Verlag, Heidelberg, 2002

[10] Pawlak Z., *Granularity of knowledge, indiscernibility and rough sets*, Fuzzy Systems Proceedings, 1998. IEEE World Congress on Computational Intelligence., Anchorage, Alaska, USA, 4-9 May 1998, pp. 106 – 110, vol. 1, 1998

[11] Pedrycz W. (ed.), *Granular Computing: An Emerging Paradigm*, Physica-Verlag, Heidelberg, 2001

[12] Pedrycz W., Gacek A., *Temporal granulation and its application to signal analysis*, Information Sciences, 143, 1-4, 2002, pp. 47-71; Application of information granules to description and processing of temporal, 2002

[13] Pedrycz W., Skowron A., Kreinovich V. (eds), *Handbook of Granular Computing*, Wiley&Sons, Chichester, West Sussex, England, 2008

[14] Polkowski L., Skowron A., *Towards Adaptive Calculus of Granules*, Proc. 1998 IEEE International Conference on Fuzzy Systems, pp. 111-116, 1998

[15] Wang Y., *The Theoretical Framework of Cognitive Informatics*, International Journal of Cognitive Informatics and Natural Intelligence, IGI Publishing, USA, 1(1), Jan., pp. 1-27, 2007

[16] Wang Y., Zadeh L.A., Yao Y.Y., *On the System Algebra*, Foundations for Granular Computing, Int. J. of Software Science and Computational Intelligence, 1(1), pp. 64-86, January-March 2009

[17] Zadeh L.A., *Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic*, Fuzzy Sets and Systems, Volume 90, Issue 2, pp. 111–127, 1997

6 Algorytmy genetyczne

6.1 Wprowadzenie

Teoria ewolucji i doboru naturalnego Darwina opisuje i systematyzuje zjawiska występujące w naturze. Wyjaśnia sposób adaptacji organizmów do otaczającego je środowiska i maksymalizacji czasu przeżywania jednostek, **optymalizacji kryteriów** korzystnych dla całego gatunku. Mechanizmy te zainspirowały informatyków i matematyków do stworzenia algorytmów opartych na tych samych zjawiskach i implementowania ich do zadań optymalizacji.

6.1.1 Optymalizacja genetyczna

W praktyce inżynierskiej często zachodzi potrzeba znalezienia parametrów, dla których system/urządzenie będzie działać w sposób optymalny. Uwzględniane jest np. zużycie energii, zużycie materiałów, czas wykonania zadania. Klasyczne podejście do optymalizacji jest następujące:

• Sformułowanie **funkcji celu** (zależnej od *n* zmiennych, stanu urządzenia, parametrów jego pracy, parametrów otoczenia, parametrów zadania, które należy wykonać, itd.), której wartość zależna od podanych argumentów będzie wyrażała stopień spełnienia zadanych kryteriów (np. sumę zużytej energii lub wynikową odległość do celu, do którego ma się dostać robot wędrujący w trudnym terenie).

a następnie:

• Poszukiwanie minimum lub maksimum funkcji celu (np. od losowego punktu startowego zmiana parametrów pracy tak, aby poruszać się zgodnie z kierunkiem gradientu malejącego lub rosnącego; optymalizacja metodą symulowanego wyżarzania; optymalizacja cząsteczkowa; lub inne metody poza zakresem tematyki tego skryptu).

W ten sposób formułować i rozwiązywać można rozmaite zagadnienia – ważne jest, aby zawsze uwzględnić warunki otoczenia, warunki pracy i zależność celu od nich.

Klasyczne ujęcie charakteryzuje się wieloma problemami praktycznymi i implementacyjnymi:

• Model taki zwykle jest bardzo skomplikowany, zależności niedookreślone, trudne do opisania,

• Potrzeba jest wykonania dużej liczby obliczeń,

• Zawsze występuje „pułapka” minimum lokalnego, tzn. nie ma pewności, że uzyskane rozwiązanie rzeczywiście jest najlepsze (patrz także: dyskusja w rozdziale poświęconym sieciom neuronowym).

W przeciwieństwie do klasycznego poszukiwania optimum, metoda genetyczna zakłada:

• jednoczesne sprawdzanie wielu hipotez poprzez ocenianie wielu osobników (osobnik w metodzie genetycznej reprezentuje jedno z rozwiązań, jedną z wersji systemu lub urządzenia, która realizować ma postawione zadanie),

• wybór najlepszych osobników – tj. symulacja przetrwania,

• tworzenie nowego pokolenia w oparciu o najlepsze osobniki – tj. symulacja doboru naturalnego,

• możliwość zachodzenia mutacji – tj. losowej zmiany cech.

Szczegóły i warianty powyżej podanych etapów genetycznych opisano poniżej.

6.1.2 Terminologia teorii algorytmów genetycznych

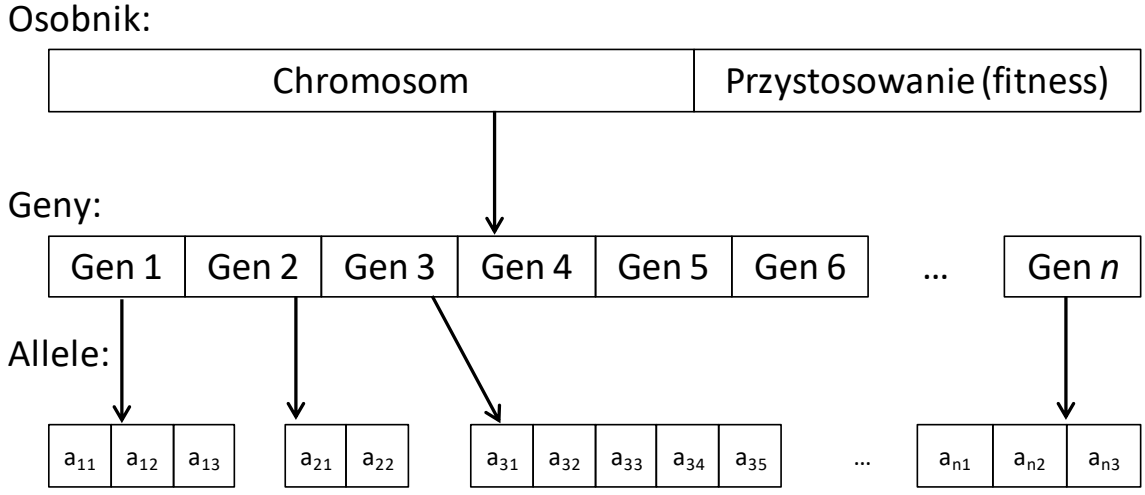
W algorytmach genetycznych stosuje się szereg pojęć zapożyczonych z biologii (rys. 6.1):

• Osobnik – pojedyncza propozycja rozwiązania problemu, charakteryzująca się indywidualnymi cechami, parametrami, które wpływają na to, jak dobrze osobnik jest dostosowany do zadania (do swojego środowiska). Osobnik o *n* parametrach to punkt w *n*-wymiarowej przestrzeni rozwiązań.

• Populacja – zbiór osobników, na których operuje algorytm; odwzorowanie pewnych punktów z przestrzeni potencjalnych rozwiązań.

• Chromosom – reprezentacja potencjalnego rozwiązania (np. wektor o długości n, drzewo, inna struktura) podlegająca ocenie poprzez działania algorytmu genetycznego. Chromosom może mieć różną postać w zależności od natury rozwiązywanego problemu.

• Gen – najmniejszy element niosący informację genetyczną. Możliwymi wartościami genu są allele (np. gen może oznaczać/kodować liczbę chwytaków robota, wartości/allele to 0, 1, 2, itd.)



Rys. 6.1 Związki między pojęciami genetycznymi

6.1.3 Przykłady osobników i chromosomów

W zależności od określonego zagadnienia i pożądanego celu osobniki i ich chromosomy mogą mieć bardzo różne postaci.

Dla zadania poszukiwania **ekstremum funkcji** – osobnikiem jest argument (argumenty) funkcji. Jeżeli jest ona funkcją jednej zmiennej, to poszukiwana jest wartość rzeczywista x dla której y=f(x) będzie minimalne lub maksymalne. Rzeczywistą wartość może reprezentować:

- chromosom binarny, np. 11000101, 01001111;
- osobno kodowane cyfry dla liczby rzeczywistej z założoną dokładnością, np. 3 cyfry przed przecinkiem i 5 po przecinku, razem 8 cyfr o wartościach 0…9.

Dla zadania poszukiwania **ciągu znaków**, np. jakiegoś hasła, bądź doboru kolejności elementów lub czynności w procesie – chromosomy typu hiasgfdo, qpom82ja, 7ama9g;1.

Dla zadań permutacyjnych, takich jak problem komiwojażera odwiedzającego tylko raz wszystkie miasta: chromosom np. w postaci ciągu niepowtarzających się liter BDAFCE, ABCDEF, FECDBA.

6.2 Algorytm genetyczny

Metody genetyczne proponują alternatywne podejście do problemu optymalizacji. Dzięki ich zastosowaniu możliwe jest szybkie przeszukiwanie przestrzeni rozwiązań z uniknięciem pułapek minimum lokalnego. Należy jednak pamiętać, że ich wynik jest jedynie przybliżeniem najlepszego rozwiązania. Jeżeli problem jest prosty i możliwe jest analityczne wyznaczenie optimum, to użycie podejścia genetycznego wykaże, że nie otrzymuje się rozwiązania identycznego z analitycznym. Może mieć to miejsce z powodu dobranych wartości genów, czy sposobów ich dekowania na docelowe cechy obiektu. W prostych przypadkach nie jest wskazane wykorzystywanie **czasochłonnych obliczeń genetycznych**, gdyż wówczas podejście analityczne (np. wyznaczanie drugiej pochodnej funkcji i jej miejsc zerowych) jest szybsze i dokładniejsze.

6.2.1 Definicja

Algorytm genetyczny (ang. GA - *genetic algorithm*) jest jedną z ewolucyjnych metod optymalizacji. Zalicza się go do klasy algorytmów heurystycznych. Przeszukiwanie możliwych rozwiązań w celu znalezienia rozwiązania najlepszego lub potencjalnie najlepszego odbywa się za pomocą mechanizmów ewolucji oraz doboru naturalnego.

6.2.2 Zasada działania algorytmu genetycznego

Algorytm genetyczny naśladuje zjawiska doboru naturalnego i zasadę przetrwania najsilniejszych i najlepiej przystosowanych do środowiska. Można zapisać go w kilku krokach.

1. Inicjacja (najczęściej w sposób losowy) początkowej populacji osobników.

Osobników istniejących jednocześnie powinno być tyle, aby w jednej generacji sprawdzany był pewien założony wycinek z wszystkich możliwych rozwiązań. Jeżeli z kombinatoryki można policzyć, że rozwiązań jest np. 10^6 to w jednym kroku sprawdzanie tylko 10^1 oznacza ryzyko, że konieczne może być wykonanie maksymalnie 10^5 kroków. Jeżeli sprawdzi się 10^2, to kroków może być 10^4. To oczywiście bardzo pesymistyczne założenie, że konieczne jest sprawdzanie wszystkich kombinacji, czyli, że nie korzysta się z założeń doboru naturalnego i krzyżowania osobników.

2. Poddaje się każdego z osobników ocenie.

Ocena może być realizowana poprzez symulowanie zachowania urządzenia/systemu, którego wewnętrzne zmienne parametry przyjmują wartości z genotypu danego osobnika. Taka ocena przystosowania powinna dawać w wyniku wartość skalarną, która pozwala m.in. uszeregować osobniki od najlepszego do najgorszego. Dlatego też mówi się w tym przypadku o **funkcji oceny**.

3. Z populacji wybiera się osobniki najlepiej przystosowane do realizacji zadania.

Wybrane osobniki zostaną genetycznymi rodzicami kolejnego pokolenia obiektów. Strategii selekcji jest kilka, są one opisane poniżej. Ważne jest, że nie zawsze wybór wyłącznie najlepszych osobników na rodziców jest dobrym rozwiązaniem. Inny istotny aspekt to decyzja o liczbie rodziców.

4. Za pomocą operacji genetycznych (krzyżowanie oraz mutacja) tworzy się nowe pokolenie.

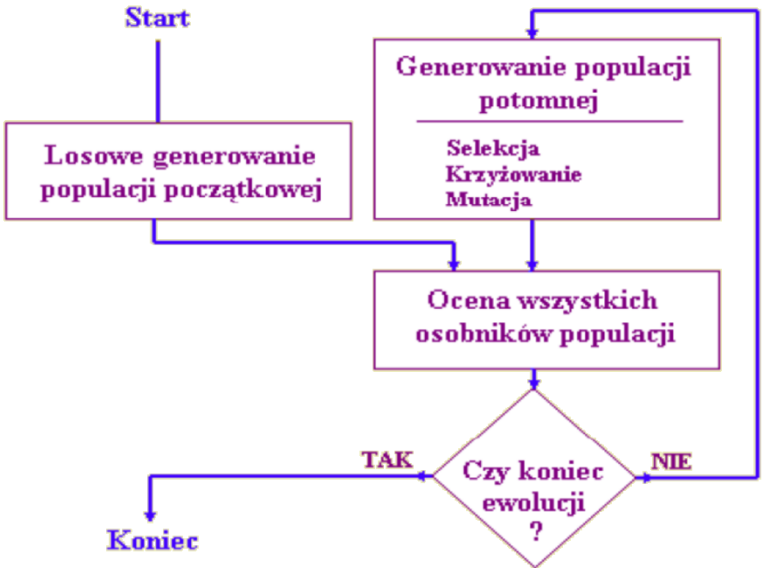
Różne możliwe strategie krzyżowania i mutacji opisane są poniżej. W wyniku kopiowania informacji genetycznej od dwóch osobników tworzony jest nowy (ich potomek), który może, ale nie musi, realizować zadanie równie dobrze lub lepiej niż jego rodzice. W niektórych algorytmach do nowego pokolenia przechodzą także wierne kopie rodziców – czyli osobniki do tej pory najlepsze nie są usuwane z populacji.

5. Sprawdzenie kryteriów zakończenia i powrót do 2.

Algorytm wykonywany może być określoną liczbę razy (np. wynikającą z podzielenia liczby możliwych rozwiązań przez liczbę osobników testowanych w jednej populacji) lub do osiągnięcia wartości funkcji oceny powyżej/poniżej założonego progu (czyli uzyskaniu osobnika dostatecznie dobrze realizującego zadanie).

Potencjalne rozwiązania traktowane są jako osobniki populacji. Algorytm symuluje biologiczny proces naturalnej selekcji poprzez ocenę przystosowania poszczególnych osobników, eliminację osobników słabszych i krzyżowanie ze sobą osobników najsilniejszych (rys. 6.2). Wynikiem działania

algorytmu genetycznego jest populacja najlepiej przystosowanych osobników, wśród których może znajdować się najlepsze rozwiązanie.



Rys. 6.2. *Schemat blokowy typowego algorytmu genetycznego*

Najlepiej przystosowane osobniki nie muszą leżeć blisko siebie w przestrzeni rozwiązań, mogą np. różnić się wartościami parametrów, które nie mają dużego wpływu na wartość funkcji oceny. Mogą także realizować stawiane im zadanie w różny sposób, ale z równie dobrym skutkiem, wówczas wskazane jest kontynuowanie procesu genetycznego, aby uzyskać takie krzyżowanie obiektów, które wykorzysta najlepsze cechy rodziców.

6.3 Kodowanie

Kodowanie to proces tworzenia fenotypu z genotypu; tj. odwzorowanie rzeczywistych parametrów problemu za pomocą reprezentacji liczbowej.

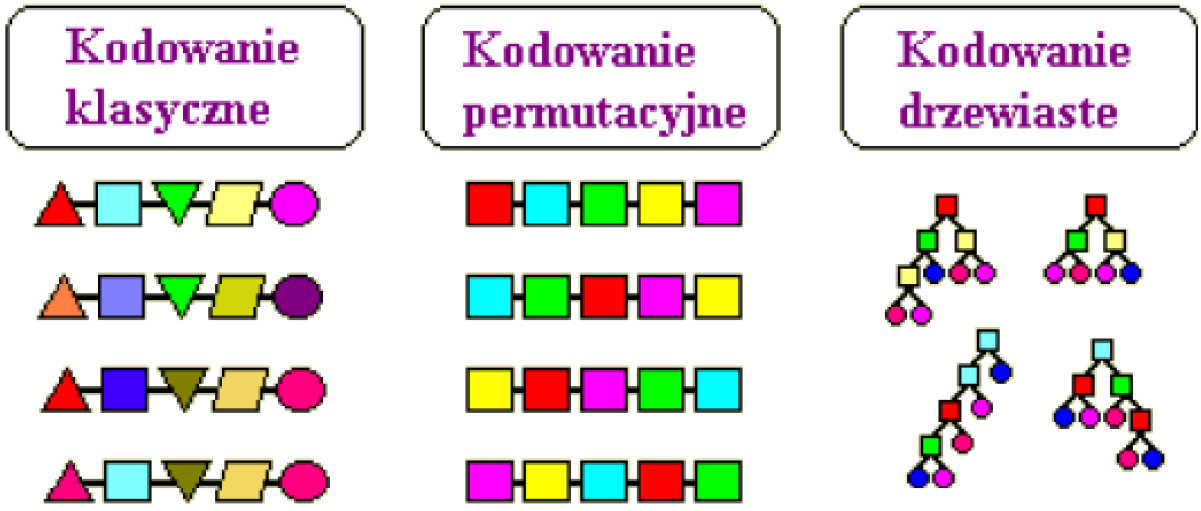
6.3.1 Warianty ułożenia genów

W zależności od rozwiązywanego problemu, optymalizowanego systemu/urządzenia, wartości genów oznaczać mogą różne cechy a ponadto występować mogą w kilku różnych wariantach (rys. 6.3).

Klasyczny - geny na różnych pozycjach przechowują różne informacje. W wyniku krzyżowania geny nie zmieniają pozycji (np. pierwsza pozycja zawsze oznacza to samo, np. liczbę chwytaków robota), lecz geny zmieniają swoje wartości. Wariant ten wykorzystywany jest w problemach, gdzie należy dobrać optymalne cechy osobnika.

Permutacyjny - geny przechowują podobne informacje, np. każdy gen to miasto na trasie komiwojażera. W wyniku krzyżowania nie zmieniają się wartości, lecz miejsca genów w chromosomie. Wariant wykorzystywany jest w problemach kombinatorycznych.

Drzewiasty - chromosom tworzy złożoną strukturę drzewiastą. W czasie krzyżowania przesunięciom ulegają całe gałęzie genów. Często geny mogą zmieniać wartości. Wykorzystywany jest w programowaniu genetycznym oraz tam, gdzie ewolucji podlegają reguły matematyczne. Na rys. 6.3 geny zaznaczone kwadratami symbolizują działania matematyczne, a okręgi to dane wejściowe – struktura koduje więc algorytm postępowania z danymi.



Rys. 6.3. Graficzna interpretacja sposobów kodowania. Różny kształt figury to różne znaczenie genu (np. w permutacyjnym są to miasta, w klasycznym 5 różnych cech obiektu), kolor symbolizuje wartość

6.4 Selekcja

W procesie selekcji wybierane są osobniki najlepiej przystosowane, które zostaną włączone do grupy rozrodczej i ich genotyp przetrwa do następnego pokolenia, tj. do następnej iteracji algorytmu. Stopień przystosowania określany jest poprzez wyliczenie wartości $y=f(x)$, gdzie f to funkcja oceny przystosowania a x to reprezentacja chromosomu ocenianego obiektu.

Istotne jest, aby właściwie dobrać stosunek wielkości tworzonej grupy do rozmiaru populacji. Zbyt małe stosunki (np. 1/1000) mogą doprowadzić do zaniku różnorodności genetycznej i defektów fenotypów, kiedy to jeden obiekt, który jest najlepszy w danej populacji, ale daleki od rozwiązania idealnego szybko propaguje swoje suboptymalne cechy na całą populację. Wówczas brakuje obiektów różnych od niego, które mogłyby w kolejnych iteracjach stać się źródłami innych informacji genetycznych. Natomiast zbyt duże (np. 1/2) powodują wprowadzenie do rozrodu zbyt dużej liczby słabych genów, co również obniża jakość najlepszych osobników.

6.4.1 Metoda koła ruletki

Tworzone jest symboliczne koło ruletki, którego tarcza podzielona jest na liczbę wycinków równą liczbie ocenianych obiektów, o różnych rozmiarach. Każdy osobnik otrzymuje obszar o rozmiarze wprost proporcjonalnym do jego oceny przystosowania. Koło ruletki puszczane jest w ruch, a po jego zatrzymaniu wskaźnik zatrzyma się na osobniku, który wchodzi do grupy rozrodczej. Im większy obszar dostał dany osobnik, tym większe prawdopodobieństwo jego wylosowania, dlatego też wielkość obszaru przydziela się przeważnie funkcją prawdopodobieństwa. Jeśli przez F(i) oznaczy się wartość funkcji przystosowania osobnika *i*, to prawdopodobieństwo jego przetrwania wynosi:

$$p(i) = \frac{F(i)}{\sum_{i=1}^{n} F(i)}$$

Gdzie w mianowniku sumuje się wszystkie wartości przystosowania dla *n* osobników.

6.4.2 Selekcja rankingowa

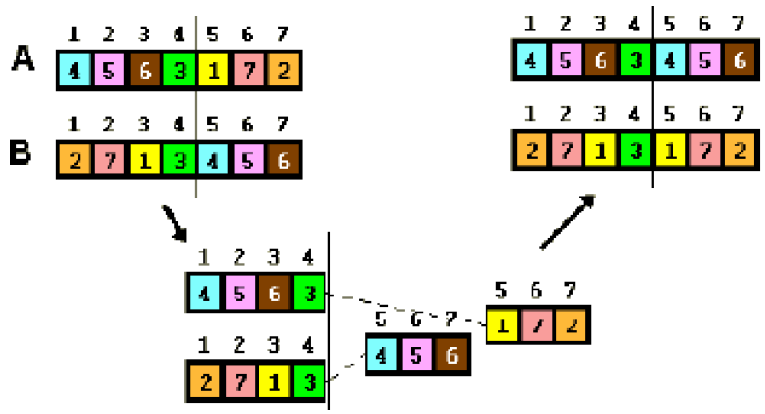
W tej metodzie osobniki populacji są sortowane według ich jakości: od najlepszego do najgorszego. Do dalszego rozrodu przechodzi tylko *k* najlepiej przystosowanych osobników. Jeżeli *k*=1, to podejście nazywane jest elitarnym, kiedy pierwszy najlepszy osobnik przekazuje swoje geny (część informacji) do kolejnego pokolenia. Metoda często daje lepsze wyniki niż koło ruletki.

6.4.3 Selekcja turniejowa

Populację dzieli się na szereg dowolnie licznych grup. Następnie z każdej z nich wybierany jest osobnik o najlepszym przystosowaniu. Ten rodzaj selekcji także sprawdza się lepiej niż metoda ruletki.

6.5 Krzyżowanie

Krzyżowanie (ang. *cross-over*) to proces generowania nowej populacji z osobników, które przeszły etap selekcji i weszły do grupy rozrodczej. Najprostszy wariant krzyżowania polega na rozcięciu chromosomów rodziców w dowolnym punkcie. Następnie, część chromosomu przypada jednemu potomkowi, a część drugiemu.



Rys. 6.4 Krzyżowanie proste: rodzice A i B wymieniają się wartościami 5,6,7, generując dwa nowe osobniki

Warianty tej metody mogą wykorzystywać przecięcia w więcej niż jednym punkcie, wymianę genów z pewnym założonym prawdopodobieństwem, itd. Najważniejsze jest, aby nie tracono informacji genetycznej obiektów-rodziców, gdyż aktualne wartości genów sprawiły, że osobniki te były lepsze od innych, jednak nie jest wiadome, które z genów miały na to wpływ. Dlatego też każdy z genów musi być przekazany do jakiegoś osobnika potomnego.

6.6 Mutacja

Mutacja to losowa zmiana w genotypie dowolnego osobnika. W przyrodzie powstaje w wyniku błędów reprodukcji lub uszkodzeń fizycznych i zwiększa różnorodność materiału genetycznego. Algorytm genetyczny symuluje zachodzenie mutacji. Prawdopodobieństwo mutacji regulowane jest przez użytkownika. Jego wartość powinna być bardzo niska, gdyż inaczej w informacji genetycznej osobników pojawi się za dużo losowości, a ewentualne korzystne geny nie będą miały szans na utrwalenie się w populacji.

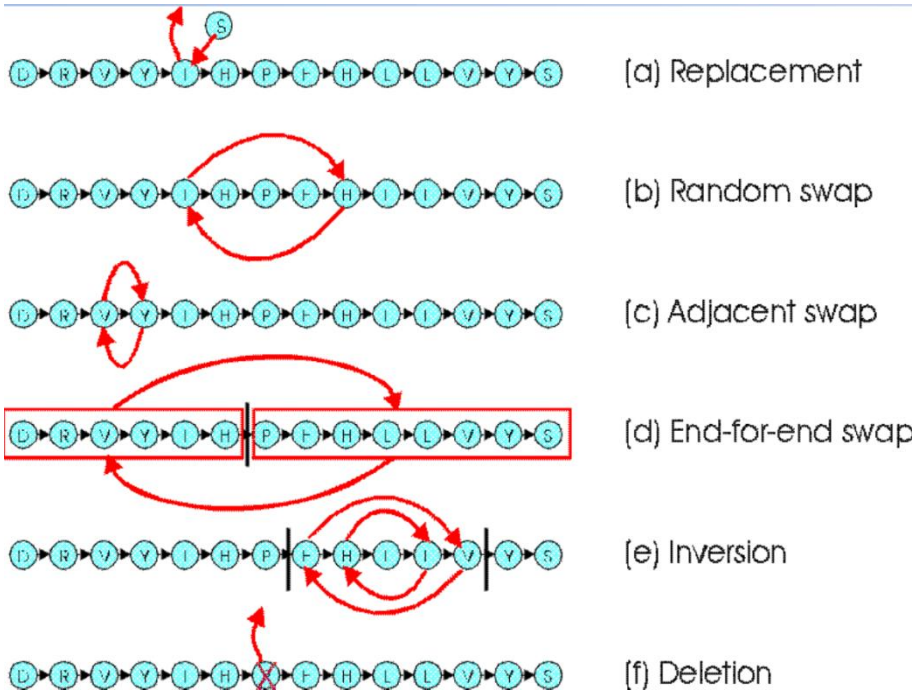
6.6.1 Metody mutacji

W zależności od rozpatrywanego problemu można stosować jedną z wielu metod mutacji.

W problemach permutacyjnych ważne jest dopilnowanie, aby po mutacji wartości genów w całym chromosomie się nie powtarzały, czyli stosować można metody **zmiany kolejności** b, c, d, e z poniższej listy.

Metoda **zastępowania** w miejsce wylosowanego genu wprowadza inną losową wartość. Ważne jest, aby zakres zmienności tej liczby losowej odpowiadał cesze kodowanej na tym miejscu ciągu genów.

Metoda usuwania genu z losowo wybranej pozycji może być stosowana łącznie z metodą wstawiania nowego genu w inne losowo wybrane miejsce. Najczęściej mają one zastosowanie w programowaniu genetycznym i kodowaniu drzewiastym.



Rys. 6.5. Graficzna reprezentacja wariantów mutacji: a) podstawienie, b) losowa zamiana, c) zamiana sąsiednich, d) zamiana końca z początkiem, e) inwersja segmentu, f) usunięcie

6.7 Literatura

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

[2] Michalewicz Z., *Algorytmy genetyczne + struktury danych = programy ewolucyjne*. WNT, Warszawa 2003

[3] Poli R., Langdon W.B., McPhee N.F., *A Field Guide to Genetic Programming*, 2008

[4] Rutkowski L., *Metody i techniki sztucznej inteligencji*, PWN, Warszawa, 2005

[5] Wierzchoń S., *Issues in Intelligent Systems*. Paradigms, Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2005

7 Przykłady zastosowania metod sztucznej inteligencji w medycynie

Tematyka niniejszego rozdziału dotyczy zagadnień z dziedziny analizy danych biomedycznych, a szczególności problemów związanych z oceną stanu pacjenta z chorobą Parkinsona (ang. PD - *Parkinson Disease*) oraz parametryzacji zaburzeń mowy. W niniejszym rozdziale wykorzystano materiał zawarty w artykule: „Wykorzystanie sieci neuronowych i metody wektorów nośnych SVM w procesie rozpoznawania aktywności ruchowej pacjentów dotkniętych chorobą Parkinsona” [37].

7.1 Proces rozpoznawania aktywności ruchowej pacjentów dotkniętych chorobą Parkinsona

Choroba Parkinsona zaliczana jest do grupy chorób neurodegeneracyjnych. Jest to powoli postępująca choroba zwyrodnieniowa ośrodkowego układu nerwowego. Obecnie około 1% osób powyżej 60 roku życia cierpi na chorobę Parkinsona (PD). Rozwój tej choroby jest monitorowany przez klinicystów, jednakże większość metod stosowanych w praktyce klinicznej nie dostarcza w pełni obiektywnej informacji zwrotnej o przebiegu PD. Leczenie osób dotkniętych PD oparte jest głównie na minimalizowaniu wpływu symptomów choroby. Jej powstawanie związane jest z zaburzeniem produkcji dopaminy przez komórki nerwowe mózgu. Choroba manifestuje się zaburzeniami ruchowymi. Przyczyna występowania tego typu zaburzeń nie została do końca wyjaśniona. Leczenie osób dotkniętych PD oparte jest głównie na minimalizowaniu wpływu symptomów choroby.

Proces rozwoju PD jest powolny i może trwać wiele lat. We wczesnym stadium choroba jest trudna do wykrycia. U osób ze zdiagnozowanym schorzeniem istotne jest powstrzymanie szybkiego rozwoju choroby. Główne objawy PD związane są z ograniczeniami pojawiającymi się w motoryce ciała pacjenta, takimi jak: mimowolne spowolnienie ruchowe, zmniejszenie szybkości i zmniejszenie amplitudy ruchu (ang. *Bradykinesia*), problemy z chodem m.in. zastyganie chodu (ang. FOG – *Freezing of Gait*), zaburzenia równowagi skutkujące upadkami, problemy z koordynacją ruchów, drżeniem kończyn (spoczynkowe i wysiłkowe) (ang. *Tremor*) [10, 11], trudności w połykaniu, spowolnienie lub brak mimiki twarzy, itd. Poza objawami związanymi z motoryką ciała PD może wpływać na trudności z koncentracją uwagi oraz planowaniem codziennych zajęć.

Jedną z metod oceny stanu pacjenta ze zdiagnozowaną chorobą PD jest wykonanie przez lekarza specjalistę serii znormalizowanych testów klinicznych, znanych jako Ujednolicona Skala Oceny Choroby Parkinsona (ang. UPDRS - *Unified Parkinson’s Disease Rating Scale*) [33]. Jednak wyniki tych testów są obarczone błędem wynikającym z subiektywnego charakteru tych testów. Oceny stanu

pacjenta uzyskane w serii testów UPDRS, przeprowadzonych w odstępach trzymiesięcznych czy półrocznych, pozwalają na ocenę postępu choroby. Przeprowadzanie badań wymaga jednak regularnych wizyt u lekarza prowadzącego, co nie zawsze jest możliwe m.in. z powodów organizacyjnych czy trudności z dostępem do specjalisty. Dodatkową przeszkodą w tym przypadku może być nie w pełni możliwe odtworzenie godziny przyjęcia lekarstw oraz efekty uboczne przyjmowanych środków, które objawiają się w postaci dyskinez (mimowolne ruchy pląsawicze). Niektóre z późnych objawów choroby Parkinsona wiążą się z wieloletnim leczeniem lewodopą lub antagonistami receptora dopaminowego [25]. Początek terapii lewodopą powoduje z reguły dużą poprawę sprawności chorych, która utrzymuje się przez kilka lat. Zazwyczaj jednak po 3-5 latach leczenia aż u 50% chorych pojawiają się specyficzne zaburzenia ruchowe zwane fluktuacjami i dyskinezami. Fluktuacje polegają na występowaniu wyraźnych zmian sprawności ruchowej chorych w ciągu doby. Z upływem czasu skuteczność działania leku zaczyna się stopniowo skracać i występują naprzemienne stany dobrej sprawności ruchowej, zwane stanami „on” i znacznie gorszej sprawności ruchowej, zwane stanami „off”. Zmiany sprawności są wyraźnie związane z rytmem przyjmowania leku i dają się przewidzieć w przypadku stanu „on”. U niektórych chorych występują jednak nagłe stany „off” bez wyraźnego związku z lekiem, a dodatkowo może pojawić się zjawisko „on-off” polegające na wielokrotnym, szybkim przejściu z jednego stanu w drugi [25]. Dlatego zbyt rzadkie lub nieregularne wizyty kontrolne nie dają informacji o ewentualnym pogorszeniu się stanu pacjenta, nie zapewniają też możliwości stwierdzenia czy dany lek (i dawka) jest właściwy, a przede wszystkim nie dają odpowiedzi na pytanie czy pacjent zgłaszający się do specjalisty jest w tzw. stanie „on”, czy „off” [36]. Brak możliwości stwierdzenia tego faktu jest przeszkodą w pełnej ocenie stanu pacjenta z PD.

Większość proponowanych obiektywnych rozwiązań diagnostyki choroby Parkinsona bazuje obecnie na technologiach wymagających montowania na palcach lub na dłoni pacjenta różnego rodzaju czujników, np. akcelerometrów lub czujników zbliżeniowych. Podejście tego typu może być stosowane w celu wspomagania procesu oceny stanu pacjenta dotkniętego PD. Tworzone są też systemy monitorujące w sposób ciągły jego stan. Przykładem takiego typu monitoringu może być system opracowany w ramach projektu PERFORM (*A soPhisticatEd multi-paRametric system FOR the continuous effective assessment and Monitoring of motor status in Parkinson's disease (PD) and other neurodegenerative diseases progression and optimizing patients' quality of life*). Jest to wieloczujnikowy system, służący do ciągłego monitorowania i oceny funkcji motorycznych osób z zaburzeniami neurodegeneracyjnymi [2,7,12,14,21]. Zadaniem systemu jest 24-godzinny monitoring stanu pacjenta na podstawie analizy sygnałów biomedycznych rejestrowanych przez specjalnie zaprojektowane czujniki umieszczone na jego ciele oraz przez serię testów przeprowadzanych z wykorzystaniem urządzeń diagnostycznych. Pacjent monitorowany jest w swoim domu, a informacje

uzyskane po wstępnym przetworzeniu zebranych sygnałów, przesyłane są do jednostki centralnej znajdującej się w szpitalu. Jednostka centralna wykonuje szczegółową analizę odebranych danych. W wyniku analizy lekarz nadzorujący ma możliwość obserwacji bieżącego stanu pacjenta, zgodnie ze skalą UPDRS na podstawie analizy motoryki pacjenta i testów wykonanych z wykorzystaniem urządzeń diagnostycznych. Może to zapewnić zdalne obserwowanie bieżącego stanu pacjentów oraz ocenę poprawności i efektywności indywidualnie dobranego schematu leczenia, a także ewentualnej jego korekty. Podczas monitorowania osób z PD wykorzystuje się głównie informacje pochodzące z czujników przyspieszenia (akcelerometrów), żyroskopów, elektrookulogramu, spirometru, czujników nacisku oraz wykonuje się analizę obrazu wideo rejestrowanego podczas wykonywania wymienionych wcześniej testów.

7.2 Rejestracja sygnałów biomedycznych

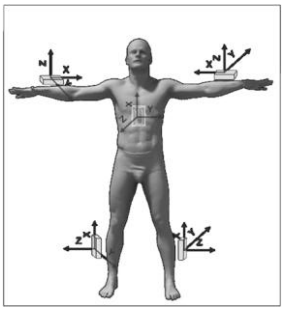
W niniejszym rozdziale przedstawiono algorytmy pozwalające na klasyfikację wybranych kategorii ruchowych u osób z PD. Zostały one zaprojektowane w celu wspomagania oceny stopnia zaawansowania choroby u osób chorych na Parkinsona. Przedstawione algorytmy pozwalają na rozpoznanie następujących kategorii ruchu: chód/brak chodu, ruch rąk/ brak ruchu. Klasyfikacja wykonywana jest na podstawie analizy sygnałów przyspieszenia pochodzących z trójosiowych akcelerometrów umieszczonych na ciele pacjenta. Problem rozpoznawania kategorii ruchowych na podstawie analizy sygnałów przyspieszenia był tematem wielu badań [3,6,17-19,24, 30]. Jednak klasyfikacja kategorii ruchowych u osób z PD stanowi zasadniczo osobny problem, ponieważ wymaga ona uwzględnienia podczas analizy zakłóceń związanych z upośledzeniem ruchowym osób chorych [31, 32, 35]. Rozpoznawanie rodzaju ruchu utrudnione jest m.in. przez drżenie ciała osoby chorej czy wspomniane wcześniej dyskinezy. Zależnie od natężenia oraz częstotliwości drżenia algorytmy, odpowiadające za analizę np. ruchu rąk, mogą błędnie rozpoznawać intensywne drżenie jako zamierzony ruch kończyny. Natomiast rozpoznawanie chodu ograniczone jest przez fakt, iż osoby z PD często podczas spoczynku wykonują nogami ruchy podobne do chodu.

Do testowania oraz trenowania klasyfikatorów rozpoznających kategorie aktywności ruchowych osób z PD, wykorzystywano sygnały przyspieszenia zarejestrowane podczas badań przeprowadzonych z udziałem pacjentów i lekarzy w Oddziale Neurologii, Szpitala Specjalistycznego św. Wojciecha w Gdańsku oraz w szpitalu w Joaninie (Grecja). W nagraniach wzięło udział 33 pacjentów (średnia wieku pacjentów 68,2 lat, zaś odchylenie standardowe 9,8 lat, wśród badanych były zarówno kobiety, jak i mężczyźni, grupy były w przybliżeniu równoliczne). Zadaniem pacjenta było wykonanie serii czynności symulujących aktywności z życia codziennego. Sekwencje wykonywanych ruchów oraz ich rejestracja

odbywały się w warunkach kontrolowanych, aby możliwe było w procesie przetwarzania wstępnego przypisanie zarejestrowanym sygnałom etykiet opisujących aktywność ruchową.

Każdy z pacjentów wykonywał następujące czynności: chodzenie po linii prostej z obrotem; podnoszenie obiektu odpowiednio ręką lewą, prawą oraz obiema rękami; podnoszenie lewej, prawej oraz obu rąk; siadanie i wstawanie z krzesła; kładzenie się i wstawanie z łóżka; stanie; siedzenie; leżenie. Każda aktywność wykonywana była w sekwencji łączącej aktywność dynamiczną z czynnością statyczną. Dla przykładu sekwencję chodu rejestrowano w sekwencji: stanie w miejscu, chodzenie z obrotem oraz zatrzymanie się. Każda sekwencja ruchu była powtarzana trzykrotnie. Podczas rejestracji sygnałów przyspieszenia rejestrowano także obraz wideo zsynchronizowany z sygnałami przyspieszenia. Nagrania wideo pozwalały na precyzyjne wyznaczanie początku i końca każdej z kategorii ruchowych.

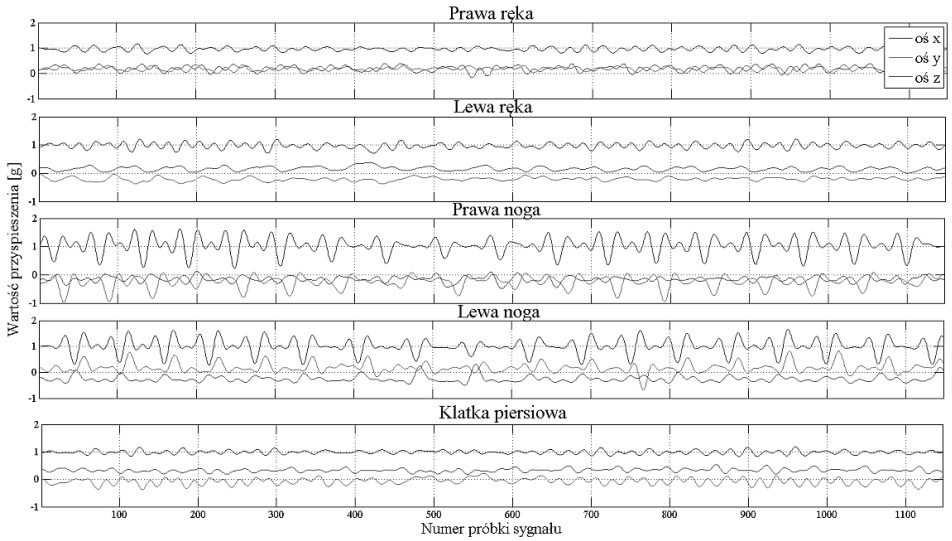
Sygnały przyspieszenia rejestrowano z wykorzystaniem dwóch rodzajów urządzeń wyposażonych w trójosiowe akcelerometry. Zbierany sygnał zapisywano na kartach *microSD* zintegrowanych z urządzeniami zapisu. Podczas testów wykorzystywano urządzenia wyprodukowane przez firmę *Shimmer* [28] oraz urządzanie zaprojektowane w ramach projektu *PERFORM* [2,7] Akcelerometry rozmieszczano na nadgarstkach, kostkach oraz klatce piersiowej osoby biorącej udział w testach. Rozmieszczenie urządzeń zostało dobrane w taki sposób, by pozwalało na detekcję jak największej liczby aktywności ruchowych oraz ocenę stanu pacjenta dokonywaną automatycznie przez algorytmy rozpoznające symptomy choroby. Jednocześnie liczba czujników została ograniczona, aby pacjent nie czuł dyskomfortu podczas użytkowania urządzeń. Zakres przyspieszeń rejestrowanych przez akcelerometry zawierał się w przedziale ±6 g, co w pełni pokrywa zakres przyspieszeń uzyskiwanych podczas typowych aktywności ruchowych ciała ludzkiego. Częstotliwość próbkowania sygnału wynosiła odpowiednio 51.2 Hz dla urządzenia *Shimmer* oraz 62.5 Hz dla systemu *PERFORM*. Sposób rozmieszczenia urządzeń na ciele osoby badanej przedstawiono na rys. 7.1.



Rys. 7.1. Sposób rozmieszczenia czujników przyspieszenia na ciele osoby badanej [14]

Podczas rejestracji sygnałów uzyskiwano 15 niezależnych przebiegów sygnału przyspieszenia. Przykładowe przebiegi czasowe sygnałów dla aktywności chodu zamieszczono na rys. 7.2. Na wykresach sygnału odpowiadającego osi *x* można zaobserwować składową stałą równą przyspieszeniu

ziemskiemu (1 g). Występowanie składowej stałej dla tej osi związane jest z orientacją osi akcelerometrów, które były ustawione w następujący sposób: oś *x* - wertykalnie, oś *y* - horyzontalnie i prostopadle do klatki piersiowej, oś *z* - wertykalnie i równolegle do klatki piersiowej.



Rys. 7.2. Przebiegi sygnałów przyspieszenia rejestrowanych podczas testów

7.3 Parametryzacja sygnałów przyspieszenia

Analizę sygnałów przyspieszenia wykonuje się zazwyczaj po wcześniejszej parametryzacji sygnału [1,9,30]. W przypadku klasyfikacji sygnałów zarejestrowanych z udziałem osób z PD konieczne jest dodatkowe przetworzenie sygnałów w celu eliminacji zakłóceń wprowadzanych przez symptomy choroby. Wcześniejsze badania [13,20] pozwoliły na stwierdzenie faktu, iż dolnoprzepustowa filtracja sygnału przyspieszenia z częstotliwością odcięcia równą 3 Hz daje najlepsze wyniki w eliminacji zakłóceń, jednocześnie nie usuwając istotnych w procesie klasyfikacji informacji związanych z częstotliwością ruchu. Dlatego przed analizą właściwą, wszystkie sygnały zostały poddane filtracji dolnoprzepustowej za pomocą filtra IIR o częstotliwości odcięcia równej 3 Hz.

Parametryzacja sygnału wykonywana była w ramkach czasowych zależnych od rodzaju aktywności. W przypadku rozpoznawania chodu zastosowano ramkę o długości 1250 ms przesuwaną z krokiem 625 ms, co odpowiadało 64 i 32 próbkom sygnału dla urządzeń *Shimmer* oraz 78 i 39 próbkom dla urządzeń opracowanych w ramach projektu *PERFORM*. Podczas parametryzacji sygnału wykorzystanej do klasyfikacji ruchu rąk zastosowano krótszą ramkę o długości 625 ms i kroku 320 ms. Zastosowanie krótszej ramki związane było z koniecznością wykrywania aktywności szybkozmiennych. Wszystkie parametry przed podaniem na wejście klasyfikatora były normalizowane do zakresu wartości <-1,1>.

Wykorzystane parametry opisujące każdą ramkę sygnału można podzielić na parametry wyznaczane w dziedzinie czasu oraz w dziedzinie widma sygnału, zostaną one przedstawione odpowiednio w podrozdziale 7.3.1 i 7.3.2.

7.3.1 Parametry w dziedzinie czasu

W procesie parametryzacji wykorzystano w pierwszej kolejności typowe parametry opisujące właściwości statystyczne parametryzowanych sygnałów. Jako pierwszy parametr przyjęto wartość średnią sygnału – parametr opisujący poziom przyspieszenia odpowiadający analizowanej ramce sygnału. Można zauważyć, że parametr ten przyjmuje wysokie wartości dla aktywności dynamicznych (np. chód, ruch rąk) i niskie w przypadku aktywności statycznych (np. siedzenia). Wartość średnią sygnału przyspieszenia można wyznaczyć w sposób następujący:

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x(n) \tag{7.1}$$

gdzie *n* - oznacza numer próbki sygnału przyspieszenia, a *N* - długość ramki sygnału wyrażoną w próbkach.

Odchylenie standardowe – przedstawia z kolei zakres zmienności sygnału:

$$std\,(x(n)) = \sqrt{\left(\frac{1}{N-1}\sum_{n=1}^{N}\left(x(n)-\bar{x}\right)^2\right)} \tag{7.2}$$

Kolejny parametr - kurtoza została wyznaczona w celu określenia dynamiki zmian sygnału przyspieszenia:

$$krt(x(n)) = \frac{m_4(x(n))}{std\,(x(n))^2} - 3 \tag{7.3}$$

gdzie $m_4(x(n))$ reprezentuje czwarty moment centralny.

Współczynnik szczytu (ang. *crest factor*) oznacza stosunek maksymalnej wartości sygnału do wartości RMS. Parametr ten opisuje charakter impulsowy sygnału:

$$k_{sz}(x(n)) = \frac{\max(x(n))}{\sqrt{\frac{1}{N}\sum_{n=1}^{N} x(n)^2}} \tag{7.4}$$

W celu opisania zależności pomiędzy ruchem poszczególnych kończyn oraz korpusem ciała wyznaczono współczynnik korelacji pomiędzy tymi samymi osiami akcelerometrów umieszczonych na różnych częściach ciała. W efekcie dla 5 czujników otrzymano 30 współczynników. Dodatkowo wyznaczono współczynnik korelacji pomiędzy różnymi osiami tego samego akcelerometru, co uzupełniło opis ruchu o zależność chwilowego położenia czujnika.

Współczynnik korelacji wyznaczony został z wykorzystaniem następującego wzoru:

$$corr(x(n)_i^l, x(n)_j^m) = \frac{\overline{x(n)_i^l x(n)_j^m} - \overline{x_i^l}\,\overline{x_j^m}}{std(x(n)_i^l)\,std(x(n)_j^m)} \tag{7.5}$$

gdzie *i*=1…5, *j* =1…5 przedstawiają numery czujników przyspieszenia, podczas gdy *l*={*x, y, z*}, *m*={*x, y, z*} oznaczają osie akcelerometrów. Współczynnik korelacji pomiędzy czujnikami wyznaczono dla wartości *i≠j* oraz *l=m,* współczynniki korelacji pomiędzy osiami akcelerometrów dla wartości *i=j* oraz *l≠m*.

7.3.2 Parametry w dziedzinie widma sygnału

Złożoność ruchu opisano za pomocą energii widmowej sygnału wyrażonej za pomocą wzoru:

$$E(A(k)) = \frac{\sum_{k=1}^{K} A(k)^2}{K} \tag{7.6}$$

gdzie *A*(*k*) jest *k-tym* prążkiem widma amplitudowego sygnału przyspieszenia, *K* oznacza całkowitą liczbę prążków widma.

Ocena okresowości ruchu opisywana jest przez entropię:

$$Ent = -\sum_{k=1}^{K} p(k)\log_2\big(p(k)\big) \tag{7.7}$$

gdzie *p*(*k*) to prawdopodobieństwo wystąpienia wartości *A*(*k*) w widmie amplitudowym sygnału przyspieszenia. Mała wartość entropii wskazuje na okresowość analizowanego sygnału.

7.4 Klasyfikacja

Klasyfikację sygnałów przyspieszenia wykonywano z wykorzystaniem dwóch zależnych od siebie klasyfikatorów, służących do rozpoznawania kategorii chód oraz do rozpoznawania ruchu rąk. Klasyfikatory zostały przygotowane w oparciu o algorytm sztucznych sieci neuronowych [29] oraz maszynę wektorów nośnych (ang. SVM - *Support Vector Machine*) [34]. Ze względu na fakt, iż klasyfikatory oparte na sztucznych sieciach neuronowych zostały przedstawione w sposób obszerny w Dodatku do Encyklopedii, dlatego w niniejszym rozdziale opis stosowanych algorytmów ograniczy się do podania najważniejszych informacji dotyczących struktury oraz parametrów klasyfikatorów oraz założeń działania algorytmu SVM. W celu zbadania optymalnego rozmieszczenia akcelerometrów, czyli takiego, które pozwala na klasyfikację danej kategorii ruchu z możliwie najwyższą skutecznością, przygotowano klasyfikatory pozwalające na rozpoznawanie aktywności ruchowych z wykorzystaniem różnej liczby czujników.

7.4.1 Rozpoznawanie chodu

Klasyfikator chodu ma za zadanie odróżnić fragmenty sygnału, w których osoba badana chodzi od sytuacji, w których wykonuje wszystkie inne aktywności ruchowe (np. kładzie się na łóżku, siada na krześle). Decyzje podejmowane przez ten klasyfikator mogą być wykorzystywane przez algorytmy dokonujące oceny np. zastygania chodu. W związku z koniecznością uwzględnienia możliwości rozpoznawania chodu z wykorzystaniem różnej liczby czujników na wejście projektowanych klasyfikatorów podawano różną liczbę parametrów. W tabeli 7.1 przedstawiono zależność liczby parametrów od liczby czujników użytych podczas analizy.

Tab. 7.1. Zależność pomiędzy liczbą czujników a liczbą parametrów

Liczba czujników	1	2	3	5
Liczba parametrów	21	45	72	135

Wykorzystywana sieć neuronowa posiadała jedną warstwę ukrytą, liczba neuronów w warstwie wejściowej zależna była od liczby akcelerometrów użytych do klasyfikacji (patrz tab. 7.1). Liczbę neuronów w warstwie ukrytej, wyznaczano zgodnie ze wzorem:

$$n_{ukr} = \frac{n_{wej}}{2} + n_{wyj} \tag{7.8}$$

gdzie n_{wej} oznacza liczbę neuronów w warstwie wejściowej, a n_{wyj} liczbę wyjść sieci.

Opracowana sieć posiadała dwa wyjścia. Kodowanie odpowiedzi było następujące, jeżeli dane należały do klasy ‘chód’, to na wyjściu oczekiwano wartości [1, 0], w przeciwnym wypadku oczekiwano wartości [0, 1]. Podczas klasyfikacji wartość ‘1’ przypisywano do wyjścia, które zwróciło największą wartość. Do treningu sieci wykorzystywano algorytm wstecznej propagacji błędu. Neurony w obydwu warstwach sieci wykorzystywały sigmoidalną funkcję aktywacji zdefiniowaną za pomocą formuły:

$$f(x) = \beta \frac{(1 - e^{\alpha x})}{(1 + e^{\alpha x})} \tag{7.9}$$

gdzie α i β przyjmowały wartości równe jedności.

W klasyfikacji chodu i ruchu rąk wykorzystano prostą strukturę sieci: warstwa wejściowa zależna była od liczby zastosowanych czujników, pojedyncza warstwa ukryta oraz warstwa wyjściowa składająca się z dwóch neuronów (wartość: chód/brak chodu, t.j. [1,0]/[0,1]).

Klasyfikator oparty na maszynie wektorów nośnych stworzono, bazując na algorytmie C_SVC (*C-Support Vector Classification*) [4,5,8]. Zastosowano gaussowską funkcję jądra wyrażoną za pomocą formuły:

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|}{2\gamma^2}} \tag{7.10}$$

Podczas trenowania klasyfikatora opartego na metodzie SVM użyto dwa różne podejścia związane z doborem parametrów kosztu (*C*) oraz *γ*. Początkowo ustawiono stałe wartości parametrów *C* = 62,5 oraz *γ* =0,5, niezależnie od liczby wykorzystywanych czujników przyspieszenia. Wartości parametrów dobrano w sposób eksperymentalny. W drugim podejściu, w celu znalezienia parametrów dających najwyższą skuteczność klasyfikacji, wykorzystano metodę *grid-search* dostępną w bibliotece OpenCV (*Open Computer Vision*), co można tłumaczyć, jako „wyszukiwanie na węzłach siatki”. Metoda ta polega na podziale obszaru wyszukiwania umowną siatką, a następnie wykorzystania do obliczeń danych dwóch parametrów, które podlegają zmianie w każdej iteracji. Ostatecznie wybiera się taką parę parametrów *C* i γ, dla których uzyskuje się największą dokładność rozpoznawania [8]. Należy zaznaczyć, że niezwykle ważne jest znalezienie takiej pary parametrów *C* i γ, aby klasyfikator najlepiej rozpoznawał nieznane dane. W drugim podejściu uzyskano wartości parametrów *C* i *γ* w zależności od liczby wykorzystywanych akcelerometrów.

7.4.2 Rozpoznawanie ruchu rąk

Klasyfikator ruchu rąk został zaprojektowany w celu rozpoznawania aktywności polegającej na poruszaniu się jednej lub obu rąk. Informacje na temat aktualnie wykonywanej czynności mogą zostać wykorzystane np. procesie oceny spowolnienia ruchowego (*bradykinezja*). Klasyfikator ten analizował jedynie te fragmenty sygnału, które nie zostały oznaczone przez detektor chodu, jako chód.

Wykorzystana sieć neuronowa posiadała strukturę niemal identyczną, jak w przypadku klasyfikacji chodu. Różnica pomiędzy strukturami występowała w liczbie wyjść sieci. Zastosowano cztery wyjścia odpowiadające możliwym klasom rozpoznawanego ruchu rąk: lewa, prawa, obie ręce, brak ruchu. Kodowanie wyjść odbywało się w sposób analogiczny, jak w przypadku klasyfikacji chodu. Użyto, także tego samego algorytm treningu oraz funkcji aktywacji neuronów, co w przypadku detekcji chodu.

Ponieważ klasyfikator oparty o maszynę wektorów nośnych, z definicji jest klasyfikatorem dwuklasowym (*k*=2), w przypadku detekcji aktywności rąk wymagał on uzupełnienia o możliwość klasyfikacji wielu klas ruchu. Zastosowano w tym przypadku metodę *one-versus-all*. Metoda ta wymaga stworzenia wielu (klasyfikatorów dwuklasowych (liczba klasyfikatorów jest równa liczbie klas), z których każdy ma za zadanie odróżnić jedną z klas od wszystkich pozostałych. Podczas rozpoznawania ruchu wektor parametrów podawany jest na wejście wielu klasyfikatorów, a końcowa decyzja, dotycząca przynależności do danej klasy, podejmowana jest na podstawie informacji dotyczącej np. pewności decyzji podjętej przez klasyfikator. Jeżeli kilka klasyfikatorów podejmie decyzję, iż wektor należy do klasy przez niego rozpoznawanej, ostatecznie wybierana jest ta klasa, dla której podjęta została decyzja z najwyższym estymowanym prawdopodobieństwem.

Prawdopodobieństwo podjęcia decyzji wyznaczane jest na podstawie modelu SVM, a implementacja pozwalająca na jej wyznaczenie dostępna jest m.in. w bibliotece libSVM [5].

7.4.3 Skuteczność rozpoznawania chodu i motoryki dłoni

Poniżej przedstawiono wyniki demonstrujące skuteczność rozpoznawania aktywności ruchowych w zależności od zastosowanego algorytmu klasyfikacji oraz metody testowania. Każdy algorytm został przetestowany z wykorzystaniem algorytmu walidacji krzyżowej oraz metody *leave-one-out* (w tym przypadku *N*-elementowa próba jest dzielona na *N* podzbiorów, zawierających po jednym elemencie). W metodzie *leave-one-out* klasyfikatory trenowano w następujące sposób: w zbiorze treningowym znajdowały się parametry odpowiadające aktywnościom ruchowym wykonywanym przez *n*-1 osób, gdzie *n* odpowiada liczbie wszystkich pacjentów, natomiast w zbiorze testującym znajdowały się parametry odpowiadające jednej osobie. Wartości skuteczności umieszczone odpowiednio w tab. 7.2 i 7.3 są wartościami uśrednionej skuteczności dla wszystkich 33 pacjentów. Metoda *leave-one-out* pozwala na ocenę zdolności generalizujących algorytmów rozpoznawania aktywności ruchowych, ponieważ testowanie odbywa się z wykorzystaniem parametrów wyznaczonych dla osób, dla których wyekstrahowane nie były wykorzystywane do trenowania klasyfikatora.

Rozpoznawanie chodu

W tab. 7.2 i 7.3 umieszczono skuteczności klasyfikacji chodu w zależności od liczby użytych akcelerometrów oraz od rodzaju klasyfikatora. Poza skutecznością wyznaczono także odchylenie standardowe w celu pokazania różnic wyników uzyskiwanych w przypadku poszczególnych osób (test *leave-one-out*) oraz kolejnych walidacji (walidacja krzyżowa).

Wynik z najwyższą skutecznością oraz najmniejszym odchyleniem standardowym dla danej konfiguracji akcelerometrów zaznaczono w tabelach poprzez pogrubienie czcionki. Najlepsze wyniki klasyfikacji uzyskane z wykorzystaniem danej metody testowania zaznaczono poprzez podkreślenie wyniku. Analizując wyniki testowania z wykorzystaniem metody *leave-one-out* (tab. 7.2), można zaobserwować, iż największą skuteczność detekcji uzyskano w przypadku klasyfikatora SVM. Najwyższą skuteczność w przypadku sieci neuronowych otrzymano tylko w dwóch sytuacjach (akcelerometr na lewej nodze - rozpoznawanie chodu; akcelerometry na obu nogach, kategoria - brak chodu). Globalną najwyższą skuteczność rozpoznawania dał klasyfikator SVM. Najdokładniejsze rozpoznawanie chodu otrzymano w przypadku zastosowania 3 akcelerometrów (nogi, klatka piersiowa), a klasa przeciwstawna została najlepiej rozpoznawana przy użyciu wszystkich dostępnych

czujników. Wyniki otrzymane podczas testowania algorytmów z wykorzystaniem metody walidacji-krzyżowej (tab. 7.3) w przypadku detekcji chodu pokrywają się z wynikami umieszczonymi w tabeli 7.2 (najwyższą skuteczność dał klasyfikator SVM w konfiguracji akcelerometrów nogi i klatka piersiowa). Ta sama konfiguracja akcelerometrów pozwoliła także na najskuteczniejszą klasyfikację klasy przeciwstawnej.

Tab. 7.2. Wyniki rozpoznawania chodu – testowanie metodą *leave-one-out*

Konfiguracja akcelerometrów	Rodzaj aktywności	SVM (C = 62.5, γ=0.5)		SVM grid-search		Sieć neuronowa	
		Skuteczność	Odchylenie standardowe	Skuteczność	Odchylenie standardowe	Skuteczność	Odchylenie standardowe
1 – lewa noga	Chód	95,63	7,34	96,34	**5,89**	**97,27**	17,82
	Brak	97,66	7,76	**97,86**	**7,57**	97,00	7,90
1 – prawa noga	Chód	**95,06**	**17,65**	94,93	18,69	94,86	17,97
	Brak	96,96	8,82	**97,18**	8,87	96,04	**8,06**
2 - nogi	Chód	96,71	13,27	97,51	10,91	**98,00**	**5,73**
	Brak	**98,86**	**3,16**	98,62	4,23	96,82	7,28
2 – prawa noga, klatka piersiowa	Chód	97,15	9,71	**98,24**	**6,97**	96,35	14,00
	Brak	96,67	9,48	**97,75**	**7,08**	95,88	9,73
3 – nogi, klatka piersiowa	Chód	98,82	2,03	**99,19**	**1,66**	85,49	20,82
	Brak	**98,12**	6,08	98,01	**6,04**	77,10	16,77
3 – lewa ręka, prawa noga, klatka piersiowa	Chód	97,02	6,74	**98,01**	**4,96**	91,02	13,63
	Brak	**97,33**	**6,51**	97,31	6,95	86,28	17,81
5 – nogi, ręce, klatka piersiowa	Chód	96,94	6,86	**97,16**	**6,65**	34,35	28,99
	Brak	**99,24**	**1,91**	98,36	3,71	83,89	19,60

Tab. 7.3. Wyniki rozpoznawania chodu – testowanie metodą walidacji krzyżowej

Konfiguracja akcelerometrów	Rodzaj aktywności	SVM (C = 62.5, γ=0.5)		SVM grid-search		Sieć neuronowa	
		Skuteczność	Odchylenie Standardowe	Skuteczność	Odchylenie standardowe	Skuteczność	Odchylenie standardowe
1 – lewa noga	Chód	**97,47**	1,14	97,03	**0,97**	97,42	1,39
	Brak	**98,39**	0,88	98,31	0,86	98,30	**0,76**
1 – prawa noga	Chód	98,37	0,89	**98,65**	**0,51**	97,50	1,30
	Brak	**99,09**	**0,54**	99,01	0,71	98,58	0,76
2 - nogi	Chód	**98,89**	**0,44**	98,99	0,72	98,68	0,90
	Brak	**99,53**	**0,41**	99,50	0,46	98,55	0,79
2 – prawa noga, klatka piersiowa	Chód	98,62	1,07	**99,14**	**0,46**	98,66	0,67
	Brak	**99,26**	**0,42**	99,16	0,53	97,77	1,41

3 – nogi, klatka piersiowa	Chód	99,37	**0,37**	**99,69**	0,42	90,54	8,01
	Brak	**99,78**	**0,26**	99,69	0,29	89,10	12,95
3 – lewa ręka, prawa noga, klatka piersiowa	Chód	98,49	1,65	**98,91**	**1,19**	92,64	8,13
	Brak	**99,36**	0,34	99,35	**0,31**	85,04	19,11
5 – nogi, ręce, klatka piersiowa	Chód	98,38	2,35	**98,99**	**1,29**	38,51	16,23
	Brak	99,65	0,37	**99,69**	**0,27**	94,04	7,48

Analiza wyników skuteczności rozpoznawania kategorii chodu w zależności od metody testowania, konfiguracji akcelerometrów oraz rodzaju klasyfikatora pokazała, że najwyższą skuteczność przy możliwie niewielkiej liczbie czujników można uzyskać stosując klasyfikator SVM i analizując sygnały rejestrowane przez trzy akcelerometry (nogi, klatka piersiowa).

Rozpoznawanie ruchu rąk

W tab. 7.4 i 7.5 znajdują się wyniki uzyskane podczas testowania algorytmów rozpoznawania ruchu rąk. Oprócz skuteczności klasyfikacji oraz odchylenia standardowego dodatkowo zamieszczono wartość błędu drugiego rzędu (ang. *false-negative*). Wartość taka określa liczbę (procent) błędnie sklasyfikowanych przykładów z danej klasy. W tabelach użyto tych samych oznaczeń dotyczących najlepszej globalnej skuteczności oraz najlepszej skuteczności uzyskanej w przypadku danej konfiguracji akcelerometrów.

Porównując wyniki, można zaobserwować, iż w przypadku walidacji krzyżowej (tab. 7.5) najwyższe skuteczności rozpoznawania kategorii ruchu dawał klasyfikator SVM. Jednak z testu wykonanego metodą *leave-one-out* widać, iż większe możliwości uogólniania daje sieć neuronowa (prawie zawsze pozwala osiągnąć najwyższą skuteczność klasyfikacji).

Tab. 7.4. Wyniki rozpoznawania ruchu rąk – testowanie metodą *leave-one-out*

Konfiguracja akcelerometrów	Rodzaj aktywności	SVM (*C*=60 *γ*=0.5)			Sieć neuronowa		
		Skuteczność	Odchylenie standardowe	Błąd drugiego rzędu	Skuteczność	Odchylenie standardowe	Błąd drugiego rzędu
2 czujniki - nadgarstki rąk	Lewa	76,33	35,90	**1,43**	**81,56**	**33,55**	4,46
	Prawa	74,88	**35,31**	**0,44**	**77,85**	38,02	1,54
	Obie	**93,15**	**13,15**	**4,30**	90,89	18,69	5,36
	Brak	99,41	0,91	50,06	**99,63**	**0,84**	**38,70**
3 czujniki - nadgarstki rąk, klatka piersiowa	Lewa	70,41	35,74	**1,76**	**77,22**	**33,03**	6,68
	Prawa	71,50	**37,75**	**0,36**	**74,95**	39,96	2,37
	Obie	**90,40**	**15,96**	**4,64**	86,21	23,15	7,30
	Brak	**99,69**	**0,60**	61,23	98,92	3,15	**46,35**

Tab. 7.5. Wyniki rozpoznawania ruchu rąk – testowanie metodą walidacji krzyżowej

Konfiguracja akcelerometrów	Rodzaj aktywności	SVM (*C*=60 *γ*=0.5)			Sieć neuronowa		
		Skuteczność	Odchylenie standardowe	Błąd drugiego rzędu	Skuteczność	Odchylenie standardowe	Błąd drugiego rzędu
2 czujniki - nadgarstki rąk	Lewa	**92,12**	**3,34**	3,94	81,81	4,10	**3,47**
	Prawa	**89,93**	**2,88**	2,64	70,62	5,19	**1,04**
	Obie	**93,60**	**2,30**	**0,95**	81,00	4,44	3,80
	Brak	**99,89**	**0,05**	**16,94**	99,69	0,13	58,57
3 czujniki - nadgarstki rąk, klatka piersiowa	Lewa	**88,52**	**3,22**	**2,94**	82,12	4,92	3,46
	Prawa	**90,12**	3,56	3,27	75,70	**2,21**	**1,35**
	Obie	**96,51**	**1,64**	**0,99**	87,53	2,49	8,01
	Brak	**99,96**	**0,04**	**17,72**	99,74	0,11	42,11

Jak pokazują wyniki przedstawione w tab. 7.4 i 7.5, najlepsze wyniki klasyfikacji ruchu rąk uzyskano stosując sieć neuronową i analizując wyłącznie sygnały rejestrowane przez czujniki umieszczone na nadgarstkach rąk.

7.5 Prace rozwojowe

W tej części rozdziału przedstawiono możliwe kierunki dalszych prac prowadzących do obiektywizacji badań pacjentów z chorobą PD. Dotyczą one możliwości analizy motoryki dłoni z wykorzystaniem przetwarzania obrazu przechwytywanego przez kamerę internetową zamocowaną na statywie. Testy motoryki dłoni należą do grupy testów UPDRS i są najczęściej wykonywaną grupą testów UPDRS. Dotyczy to w szczególności testów: UPDRS 23, 24, 25. Ocenie poddawane są takie aktywności dłoni, jak: stykanie palców, otwieranie i zamykanie dłoni oraz obrót dłoni. Automatyczna klasyfikacja tego typu testów jest alternatywnym sposobem testowania motoryki dłoni do testów z wykorzystaniem rękawiczek zawierających np. czujniki ścisku lub akcelerometry [15, 17].

Podejście to bazuje na inteligentnej analizie obrazu dłoni rejestrowanej przez kamerę podczas wykonywania testów. Warstwa sprzętowa rozwiązania złożona jest z kamery internetowej mocowanej na statywie oraz komputera. Kamera usytuowana jest w taki sposób, aby rejestrowała obraz dłoni umieszczonej równolegle do powierzchni stołu/biurka, itp. Zarejestrowany sygnał wizyjny poddany jest przetwarzaniu i analizie obrazu, które pozwalają na detekcję gestów wykonywanych przez pacjenta. Warstwa przetwarzania obrazu odpowiedzialna jest za przechwytywanie obrazu z kamery, rozpoznawanie gestów dłoni oraz tworzenie wyników opisujących ruch. Przetwarzanie obrazu składa się z dwóch głównych kroków: detekcji dłoni w obrazie, tworzenia modelu 3D dłoni oraz rozpoznawania gestów. Dodatkowym zadaniem konstruowanego systemu jest klasyfikacja

wykonanego gestu oraz w przypadku kolejnych wizyt pacjenta, automatyczna ocena postępu choroby. Klasyfikacja gestów odbywa się z wykorzystaniem metody sztucznych sieci neuronowych, maszyny wektorów nośnych SVM oraz ukrytych modelach Markowa (HMM) [KK, BK].

Klasyfikacja gestów dynamicznych i przypisanie rozpoznanemu gestowi odpowiedniej wartości skali UPDRS wymaga zastosowania metody, która będzie wspomagać w sposób automatyczny specjalistę, ale będzie jednocześnie czytelna do interpretacji przez specjalistę, dlatego proponowany system wymaga zastosowania metody wykorzystującej reguły decyzyjne.

7.6 Analiza i parametryzacja sygnału mowy

Narząd głosu składa się z trzech połączonych systemów – systemu oddechowego, generatora krtaniowego i traktu głosowego. Systemy te są ściśle uzależnione od siebie, lecz każdy z nich pełni oddzielną funkcję. Dodatkowo w skład elementów uczestniczących w wytwarzaniu i kształtowaniu mowy wchodzi również część centralna. Współdziałanie wszystkich struktur traktu głosowego jest konieczne w procesie fonacji, a osłabienie któregokolwiek z elementów prowadzi do zaburzeń wytwarzania głosu, dlatego proces parametryzacji powinien pozwolić na różnicowanie wartości parametrów w zależności od elementu struktury i jego prawidłowego/bądź nieprawidłowego działania.

Trakt głosowy obejmuje następujące elementy:

a) klatkę piersiową i płuca; ich rolą jest tłoczenie strumienia powietrza przez krtań;
b) krtań, która jest generatorem dźwięku i podstawowym narządem głosu;
c) przedsionek krtani, gardło, jamę ustną i nosową, zatoki przynosowe, których funkcją jest wzmocnienie i kształtowanie dźwięku.

Zgodnie z analizą procesów biomechanicznych w narządzie głosu do wytworzenia mowy potrzebne jest współistnienie następujących elementów:

- mechanizmu drgającego powodującego rytmiczne otwieranie i zamykanie się głośni,
- podgłośnieniowego ciśnienia wydechowego wytwarzającego podmuch powietrza,
- przestrzeni rezonansowej klatki piersiowej oraz gardła, które nadają dźwiękowi barwę.

O barwie dźwięku decydują jego cechy widmowe, a ściślej rozkład i zmiany harmonicznych w czasie. Ton krtaniowy stanowi generator sygnału mowy, dlatego widmo tonu krtaniowego jest jednym z ważniejszych czynników kształtujących barwę dźwięku. Jednym z parametrów określających właściwości tonu krtaniowego jest szybkość spadku amplitud harmonicznych ze wzrostem częstotliwości. Widmo tonu krtaniowego nie jest jednak jedynym czynnikiem kształtującym barwę głosu. Trakt głosowy oprócz funkcji artykulacyjnych posiada funkcję kształtowania barwy, jest to

szczególnie zauważalne w kształtowaniu barwy śpiewu. Dźwięk wytworzony w wyniku drgania strun głosowych jest słaby i wymaga wzmocnienia. Możliwe jest ono dzięki rezonatorom traktu głosowego. Należą do nich: przedsionek krtani, gardło, jama ustna i nosowa, zatoki przynosowe. Każde z nich ma charakterystyczny zakres częstotliwości, które wzmacnia, i które tłumi. W ten sposób modelują one wytwarzany dźwięk. Kształt i pojemność wymienionych przestrzeni rezonacyjnych tworzy pośrednio barwę i bezpośrednio siłę powstającego głosu. W procesie artykulacji mowy można wpływać na wynikowy głos poprzez ruch warg, języka, żuchwy oraz podniebienia miękkiego, powoduje to zmiany wnęk rezonacyjnych, co z kolei skutkuje wzmacnianiem innych składowych sygnału i powstaniem odpowiedniego dźwięku.

Analiza sygnału mowy w celach klasyfikacji oraz terapeutycznych wykorzystuje w dużym stopniu elementy systemów decyzyjnych, a w szczególności – w pierwszej kolejności – bloki akwizycji i parametryzacji sygnału mowy. W przypadku procesu klasyfikacji czy rozpoznawania zaburzeń mowy akwizycja sygnału mowy będzie rozumiana jako przygotowanie sygnału do automatycznej parametryzacji (np. jednolity format sygnału cyfrowego, itp.), a nie klasycznie rozumiane pozyskiwanie sygnału cyfrowego poprzez próbkowanie (twierdzenie o próbkowaniu i wynikające z niego kryterium Nyquista) i kwantyzację sygnału analogowego.

Zaprojektowany właściwie system parametryzacji pozwala na wstępną analizę, a nawet klasyfikację przypadków medycznych. Należy bowiem pamiętać, że patologie wytwarzania i artykulacji głosu wpływają na parametry akustyczne sygnału mowy. Do typowych analiz sygnału mowy należą: czasowa, widmowa, czasowo-częstotliwościowa, cepstralna, itd. Najczęściej stosowane parametry sygnału mowy odnoszą się właśnie do wymienionych wcześniej metod analizy. Do podstawowych parametrów sygnału mowy można zaliczyć:

- czasowe, np. gęstość przejść przez 0, energia, środek ciężkości i obwiednia sygnału,
- widmowe, np. momenty widmowe zwykłe i znormalizowane, płaskość widma. Wśród parametrów widmowych można wyróżnić parametry:
  - formantowe (częstotliwość i poziom formantu),
  - cepstralne, obliczane na podstawie cepstrum sygnału, czyli transformacji Fouriera logarytmu widma. Wektor takich parametrów jest wektorem wybranych współczynników cepstrum. Niskie współczynniki niosą informacje o trakcie głosowym, a wysokie o tonie krtaniowym [41].
  - LPC (współczynniki filtra analizującego sygnał mowy wykorzystywanego w tej technice).
- inne parametry sygnału mowy, pozyskiwane w analizach sygnału mowy.

Pierwszym z obliczanych parametrów w przypadku sygnału mowy jest częstotliwość podstawowa. Wyznacza się też jej wartość średnią oraz odchylenie standardowe.

Kolejnymi, istotnymi parametrami są częstotliwości pierwszych trzech formantów, tj. F_1, F_2, F_3 (również oblicza się wartości średnie oraz odchylenie standardowe tych parametrów). Formanty stanowią obszar szczególnie dużego wzmocnienia niektórych składowych tonu krtaniowego [38, 39, 41]. Ich powstawanie jest skutkiem działania górnych rezonatorów traktu głosowego. Miejsce powstawania formantów kolejnych formantów:

- jama gardłowa dla F_1,
- jama ustna dla F_2,
- przedsionek krtani dla F_3.

W ujęciu matematycznym, formanty odpowiadają maksimom obwiedni widma sygnału. Poprzez proces artykulacji mowy, a więc świadome ruchy narządów mowy, zmieniają się rozmiary i właściwości wnęk rezonansowych kształtujących dźwięk. Zmiany są widoczne na wykresie charakterystyki traktu głosowego. Dla poszczególnych głosek i wyrazów zmieniają się maksima charakterystyki częstotliwościowej – formanty mają inne położenie, różna jest ich liczba, wielkość. Ma to ogromne znaczenie badawcze: pozwala rozróżnić np. zgłoski pomiędzy sobą, a także cechy osobnicze właściwości traktu.

Kolejnym badanym parametrem może być energia sygnału. Oblicza się ją ze wzoru:

$$E = \sum_n x^2(n), \tag{7.11}$$

Parametr ten należy do parametrów czasowych. Jednostka to $Pa^2 \cdot s$.

Z kolei harmoniczność określana jest jako współczynnik stosunku energii sygnału zawartego w składowych harmonicznych do energii szumu. Określa ona stopień okresowości sygnału (inaczej *HNR*, *Harmonics-to-Noise Ratio*). Wyraża się ją w decybelach.

Parametry opisujące kształt widma sygnału to między innymi momenty widmowe. Wyznaczany środek ciężkości widma odnosi się w interpretacji do momentu unormowanego pierwszego rzędu. Przy widmie zawierającym wiele składowych, wartość tę utożsamia się ze średnią ważoną częstotliwości współczynników widma.

Moment widmowy *m*-tego rzędu określa się jako:

$$M(m) = \sum_{k=0}^{\infty} |G(k)| \cdot [f_k]^m, \tag{7.12}$$

gdzie f_k to częstotliwość środkowa *k*-tego pasma wyróżnionego w analizie częstotliwościowej, a $G(k)$ to widmo gęstości mocy.

Z kolei moment unormowany *m*-tego rzędu jest definiowany jako:

$$M_u(m) = \frac{M(m)}{M(0)}, \tag{7.13}$$

gdzie: M(0) jest momentem zerowym.

Momenty unormowane centralne obliczane są właśnie względem środka ciężkości widma, zgodnie ze wzorem:

$$M_{uc}(m) = \sum_{k=0}^{\infty} \frac{|G(k)| \cdot [f_k - M_u(1)]^m}{M(0)}, \tag{7.14}$$

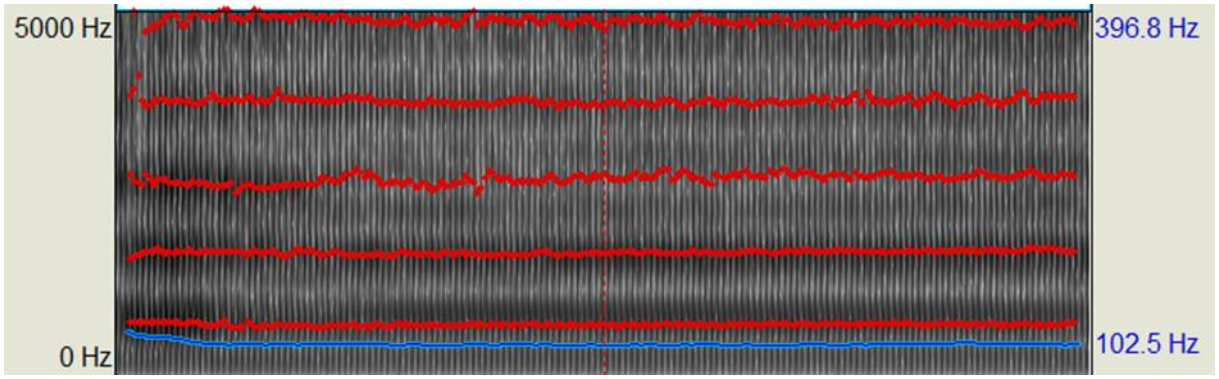
7.6.1 Zaburzenia głosu

Definicyjnie, aby głos mógł być określony jako prawidłowy, powinien być dźwięczny, czysty, tworzony swobodnie, bez napięcia krtani i szyi, być bogaty rezonansowo, bez komponentu szumowego, niemęczliwy, z miękkim nastawieniem głosowym. W przypadkach, w których niemożliwe jest wytworzenie prawidłowego dźwięku, mówi się o zaburzeniu głosu. Podłoża zaburzeń głosu (dysfonii) są bardzo zróżnicowane i ich właściwe rozpoznanie jest czasem trudne

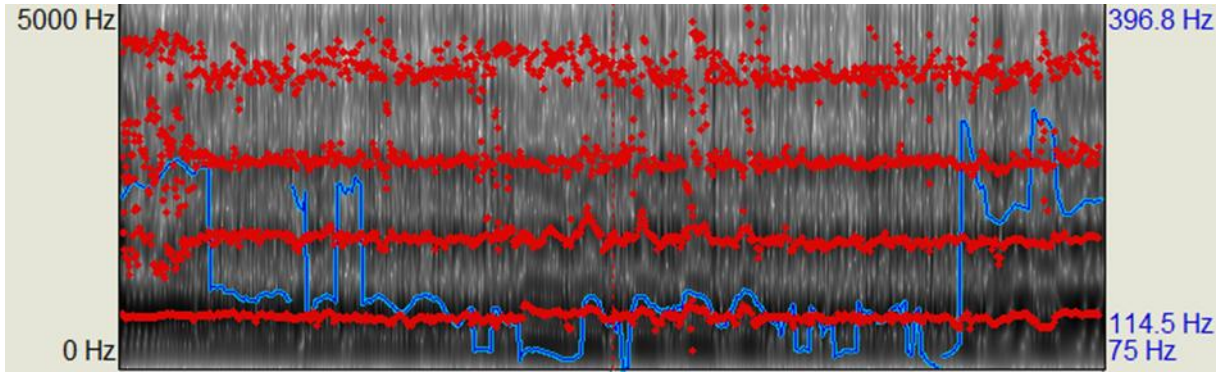
Najczęstszymi przykładami zaburzeń są: chrypka, szorstkość głosu, drżenie głosu, dysfonia hipofunkcyjna, afonia, dwugłos czy jąkanie. Chrypka to najczęstszy objaw nieprawidłowości powstawania głosu. W warunkach fizjologicznych fałdy głosowe drgają symetrycznie, regularnie i równocześnie, a amplituda, częstotliwość i faza ich drgań jest jednakowa. W wypadku chrypki drgania strun znacznie odbiegają od tego opisu. Najczęściej jest to spowodowane zmianami chorobowymi w tkankach, np. obrzękiem zapalnym, nowotworem, zmniejszonym napięciem wywołanym porażeniem nerwu. Chrypka może być też początkiem zaburzeń czynnościowych u osób nadużywających głosu, pomimo braku zmian patologicznych w obrębie krtani. Odbieganie od przedstawionych wyżej warunków fizjologicznych skutkuje różnymi częstotliwościami drgań fałdów i ich krawędzi. Powstają interferencyjne fale, przejawiające się jako zmiany barwy głosu.

Na rys. 7.3 przedstawiono przykład spektrogramu wypowiedzi osoby zdrowej oraz osoby z chrypką.

a.



b.



Rys. 7.3. Spektrogram wypowiedzi: (a) głos zdrowy, (b) głos z chrypką (przebieg formantów - kolor czerowony, częstotliwość podstawowa F_0 - kolor niebieski – widać zmiany w przebiegu F_0

Również szczególnie szorstki głos może być oceniany jako ochrypły. Szorstkość określana jest jako nieprzyjemna dla ucha cecha barwy dźwięku spowodowana zjawiskiem dudnienia tonów składowych dźwięków lub szumem wąskopasmowym. Stopień odczuwanej szorstkości zależy od częstotliwości podstawowej F_0. Im F_0 jest niższa, tym łatwiej odczuwalna jest szorstkość.

Drżenie głosu jest z kolei cechą łatwo dostrzeganą na wykresie spektrogramu. Częstotliwość podstawowa F_0 i wysokość są wtedy niestabilne i widać ich zmiany. Powodem takiej sytuacji jest zbyt małe napięcie mięśnia głosowego. Może to wynikać ze zmęczenia i przepracowania głosu.

Dysfonia hipofunkcyjna, rodzaj dysfonii czynnościowej, objawia się osłabieniem siły głosu, niemożnością długiego mówienia, skróceniem czasu fonacji oraz koniecznością wykonywania częstych wdechów. Głos traci na swojej dźwięczności, jest cichy, często pojawia się chrypka. Charakterystyczne jest zmniejszone napięcie mięśni krtani w trakcie fonacji, co powoduje niepełne zwarcie głośni. Dysfonia hipofunkcyjna zazwyczaj nie występuje u osób młodych i często wiąże się z zaburzeniami ogólnymi, np. zatruciem, awitaminozą, wyniszczeniem. Afonia może być skrajnym stanem zaawansowanej dysfonii. Przejawia się zupełną niemożnością wydania głosu. Do bezgłosu może także doprowadzić skrajnie silny stan emocjonalny albo porażenie nerwów krtani.

Dwugłos jest często skutkiem nieprawidłowego mechanizmu powstawania tonu krtaniowego. W tym przypadku w jego wytwarzaniu udział biorą, oprócz strun głosowych, również fałdy przedsionkowe. Łatwo zobaczyć tę dysfunkcję na wykresie zmian częstotliwości podstawowej, gdzie widać jej „skoki” o oktawę (właściwie obecność dwóch parametrów F_0). Dwugłos może być spowodowany m. in. opóźnioną mutacją, guzkami śpiewaczymi, polipami więzadeł oraz porażeniem nerwów krtani.

Jąkanie traktowane jest ogólnie jako brak ciągłości wypowiedzi, jedna z definicji precyzuje: „Jąkanie jest niepłynnością mówienia, spowodowaną nadmiernymi skurczami mięśni oddechowych, fonacyjnych i artykulacyjnych, której to niepłynności towarzyszą różnorodne reakcje indywidualne i społeczne, zakłócające komunikacje międzyludzką". Do typów jąkania należą: kloniczne, cechujące się powtarzaniem, toniczne, polegające na zablokowaniu powietrza, toniczno – kloniczne. Z punktu widzenia analizy sygnałowej i w tym przypadku stosuje się parametryzację jako wstępny element procesu przetwarzania i klasyfikacji zaburzonego jąkaniem sygnału mowy.

7.6.2 Analiza sygnału mowy osób z rozszczepem podniebienia

W niniejszym rozdziale przedstawiona zostanie przykład parametryzacji sygnału, która pozwala na rozróżnienie głosu patologicznego od prawidłowego. Dla celów analizy zostały zrealizowane nagrania mowy osoby z rozszczepem podniebienia (rozszczep podniebienia, podobnie jak i wargi jest wada, która powstaje u dziecka w embrionalnej fazie życia; rozwarstwienie to może utrudniać prawidłową wymowę) oraz dla porównania nagrania mowy osoby z wymową poprawną. Były to liczby od jeden do dwadzieścia, wybrane litery alfabetu greckiego, alfabet polski oraz kilka zdań służących do ćwiczeń logopedycznych. Nagrania zostały zrealizowane cyfrowo: w formacie PCM, 16 bitów, próbkowanie 22,05 kHz. Do celów analizy zostały oznaczone momenty czasów początkowych wyrazów lub fragmentów zdań (w tysiącach próbek). Dane te zostały zamieszczone w tab. 7.6.

Pierwszym etapem badań porównawczych zawartych w niniejszym rozdziale była analiza cepstralna, mająca na celu uzyskanie parametrów formantów samogłosek: częstotliwości i wartości poziomów. Problem, który pojawił się na samym wstępie polegał na konieczności ustalenia parametrów analizy cepstralnej: szerokości analizowanego pasma, rzędu wygładzania cepstralnego oraz zastosowania preemfazy, tak aby uzyskane wyniki były możliwie najbliższe znanym z literatury (tabl. 7.7 i 7.8). Ustalono długość okna analizy jako 1024 próbki, co zapewnia rozdzielczość analizy widmowej około 23,41 Hz, wystarczającą dla wyodrębnienia poszczególnych składowych harmonicznych tonu krtaniowego. Wyniki analizy widmowej dla samogłoski A zostały przedstawione na rys. 7.4 (dla pasma 0 – 11,025 kHz) i rys. 7.5 (dla pasma 0 – 3,6 kHz). Podobne wyniki, ale z

zastosowaniem preemfazy 6 dB/oktawę przedstawiono na rys. 7.6 i rys. 7.7. Ograniczenie pasma wynika z lokalizacji dwóch najsilniejszych (i najważniejszych) formantów.

W oparciu o powyższe wyniki wykonano szczegółowe serie analiz cepstralnych (wykorzystujących transformację kosinusową), których przykładowe wyniki w postaci graficznej zostały pokazane na rys. 7.8 (wykres cepstrum mocy) oraz na rys. 7.9 i 7.10 (wykresy widma wygładzonego cepstralnie), a także rys. 7.11 i 7.12 obrazujące obecność maksimów widma wygładzonego cepstralnie w zależności od rzędu wygładzania, co jest pomocne do obrania parametrów analizy. Do analiz porównawczych samogłosek wybrano pasmo 0 – 5,5 kHz, rząd analizy cepstralnej = 18 oraz wstępną preemfazę 6 dB/oktawę dla sygnału.

Z powyższych analiz można uzyskać następujące wyniki: częstotliwość tonu krtaniowego (parametr F0) oraz częstotliwości i poziomy czterech pierwszych formantów (parametry F1, F2, F3 i F4 i odpowiednio L1, L2, L3 i L4). Skorzystano przy tym z interpolacji kwadratowej, tak aby uniknąć ograniczeń analizy wynikających ze skończonej rozdzielczości w dziedzinie częstotliwości (dla estymacji parametrów F1, F2, F3 i F4) oraz w dziedzinie czasu na wykresie cepstrum (dla estymacji parametru F0).

Drugim etapem była analiza porównawcza wybranych głosek szumowych z zastosowaniem zarówno analizy cepstralnej dla uzyskania parametrów formantów, jak i analizy statystycznej, dającej w efekcie wartości momentów widmowych oraz wartości parametrów pochodnych: szerokość widma, kurtosis oraz parametr związany z charakterem szumowym/sygnałowym (tj. wspomniana wcześniej płaskość widma, ang. *spectral flatness measure*). Przykładowe wyniki takich analiz zostały pokazane na rys. 7.13 (analiza widmowa i statystyczna), rys. 7.14 (widmo wygładzone cepstralnie) oraz rys. 7.15 (zależność maksimów widma od rzędu wygładzania).

Etapem trzecim było zastosowanie analizy wykorzystującej technikę predykcji liniowej oraz pojęcie płaszczyzny zespolonej dla wyznaczania biegunów transmitancji traktu głosowego, a także wygładzonego widma pokazującego formanty. Przyjęto rząd LPC = 8. Przykładowe widmo LPC dla głoski szumowej zostało podane na rys. 7.16. Obserwowane na tym wykresie maksima wskazują na obecność biegunów w dziedzinie zespolonej, oparta na wynikach tej analizy wielokrotna predykcja pozwala na stworzenie sygnału odpowiadającego założonemu modelowi. Dla uwypuklenia maksimów i dokładniejszej estymacji częstotliwości formantu konieczna jest analiza wewnątrz jednostkowego koła. Jest to możliwe np. w analizie o promieniu mniejszym od jedności. Prowadzi to do uzyskania widma, w którym maksima są zarysowane ostrzej (rys. 7.17). Stopniowo zmniejszając promień analizy, można osiągnąć sytuację, w której okrąg analizy przecina biegun (granica stabilności) i uzyskuje się inny obraz widma LPC (7.18 i 7.19).

Tab. 7.6. Zestawienie analizowanych fragmentów mowy

		1.SND		2.SND	
1	1	24	14	18	16
2	2	45	14	42	16
3	3	70	14	67	15
4	4	96	16	93	20
5	5	127	13	120	17
6	6	150	16	146	17
7	7	183	14	177	18
8	8	220	16	205	16
9	9	246	18	230	19
10	10	275	18	259	21
11	11	307	21	289	24
12	12	342	21	324	22
13	13	379	21	358	24
14	14	411	23	392	30
15	15	453	20	431	23
16	16	484	22	466	26
17	17	516	24	503	28
18	18	559	23	539	25
19	19	597	24	570	27
20	20	638	19	601	21
21	alfa	691	17	649	16
22	beta	723	14	682	17
23	gamma	750	17	707	20
24	delta	780	16	749	18
25	zeta	814	14	769	16
26	a	862	8	813	10
27	b	887	12	833	12
28	c	920	10	855	10
29	ć	945	7	875	10
30	d	969	10	895	11
31	e	996	7	917	9
32	f	1026	13	936	13
33	g	1061	12	958	11
34	h	1091	10	978	13
35	i	1121	8	1002	8
36	j	1149	12	1022	13
37	k	1182	8	1043	11
38	l	1212	10	1063	11
39	ł	1237	14	1081	12
40	m	1271	13	1102	11
41	n	1301	11	1123	10
42	ń	1331	8	1141	11
43	o	1359	10	1164	8
44	p	1391	10	1183	9
45	r	1421	12	1203	11
46	s	1451	11	1224	11
47	ś	1481	10	1244	10
48	t	1513	9	1265	10
49	u	1543	12	1284	8
50	w	1575	12	1302	10
51	x	1606	13	1324	11
52	z	1637	13	1344	14
53	ź	1674	8	1364	10
54	ż	1705	10	1384	10
55	W szkołach	1767	14	1426	16
56	położonych	1781	20	1441	26

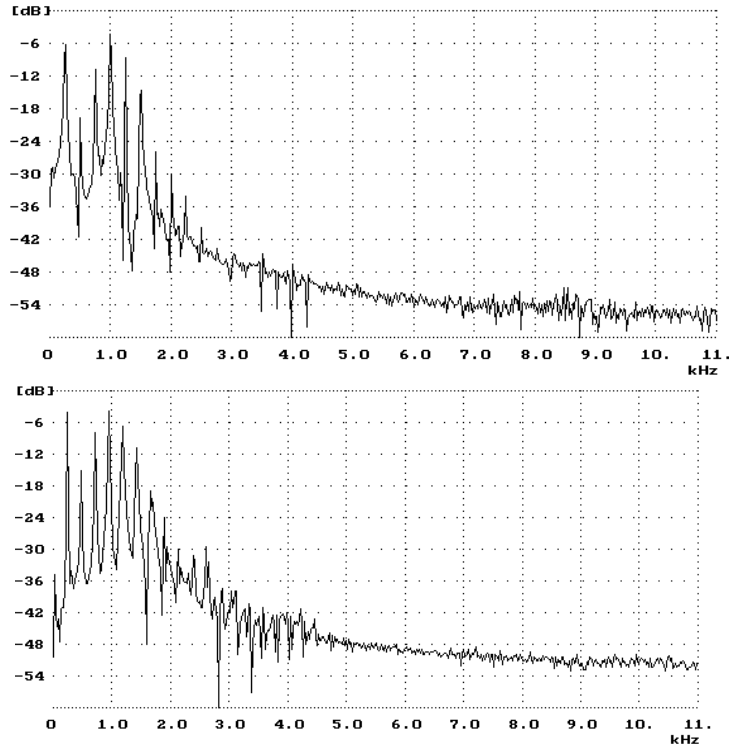
57	w małych	1801	14	1477	12
58	miejscowościach	1815	31	1489	29
59	dzieci często	1846	22	1534	29
60	ściągają	1868	15	1571	20
61	prace pisemne	1884	24	1591	27
62	W ostępie mieszka	1945	33	1651	37
63	klępa	1977	14	1694	22
64	której gęba	2000	24	1722	27
65	przypomina sępa	2025	28	1749	36
66	Ten słoń	2129	15	1818	24
67	nazywa się Niuniek	2147	31	1850	33
68	Moja mama	2193	18	1911	24
69	ma małe mieszkanie	2220	27	1935	36
70	Wąż	2269	13	1998	15
71	syczy głośniej	2292	25	2026	34
72	niż	2326	5	2069	14
73	można byłoby	2331	19	2094	22
74	się spodziewać	2350	22	2116	25

Tab. 7.7. Przykładowe częstotliwości formantowe oraz amplitudy względne (w dB) dla wybranych fonemów mowy polskiej (wg. Basztury)

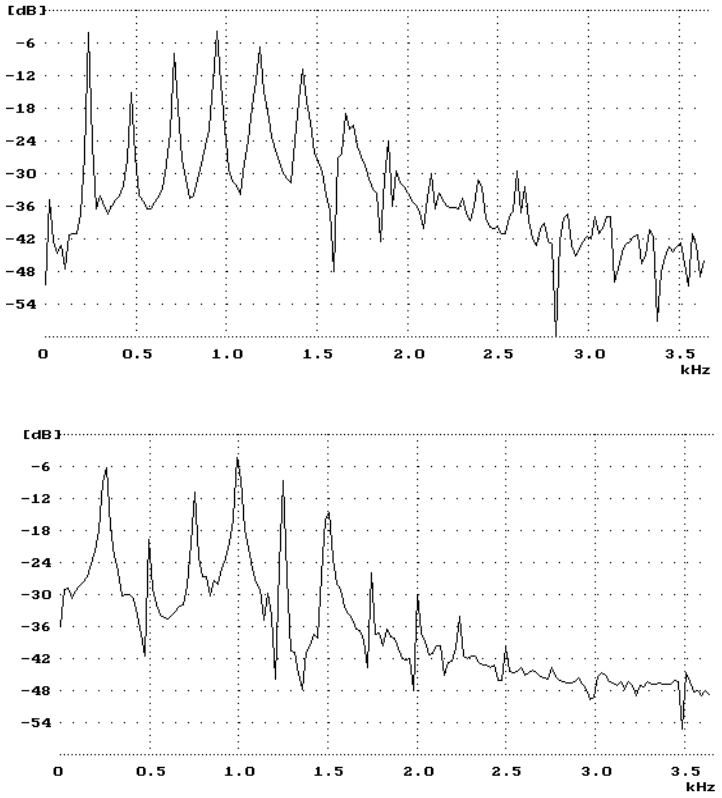
a 780 1150 (-7) 2700 (-25) 3500 (-25)
e 380 2640 (-12) 3000 (-16) 3600 (-20)
i 210 2750 (-15) 3500 (-15) 4200 (-27)
o 400 730 (-3) 2300 (-30) 3200 (-35)
u 270 615 (-13) 2200 (-40) 3150 (-50)
y 240 1550 (-7) 2400 (-20) 3300 (-30)
w 600 1700 (+9) 2900 (+7) 4100 (-1)
sz --- 2300 (-9) 2900 (-8) 3600 (0)
h 500 1700 (+12) 2500 (+2) 4200 (-5)
z --- 1750 (-6) 2950 (-10) 4300 (0)

Tab. 7.8. Częstotliwości pierwszego i drugiego formantu dla samogłosek polskich (wg. Jassema)

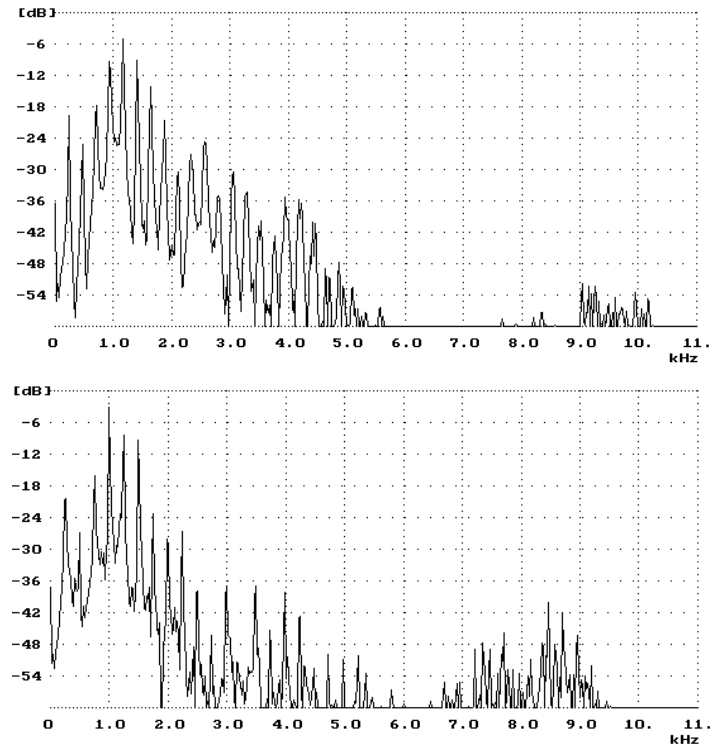
a 680-800 1300-1500
e 550-700 1680-2050
i 150-260 2250-2750
o 350-630 870-1020
u 280-360 570-820
y 220-370 1930-2300



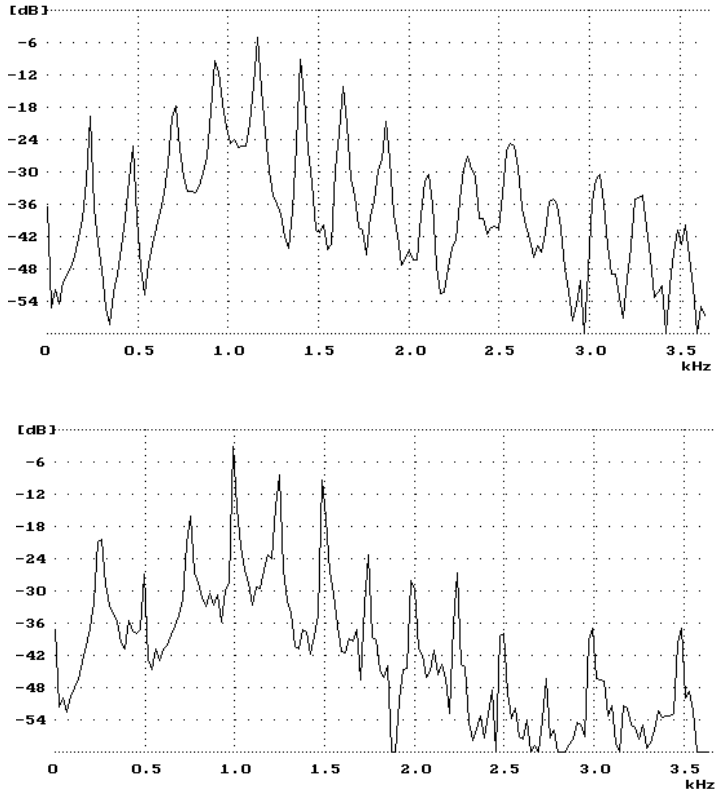
Rys. 7.4. Wynik analizy widmowej dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 11,025 kHz, bez preemfazy



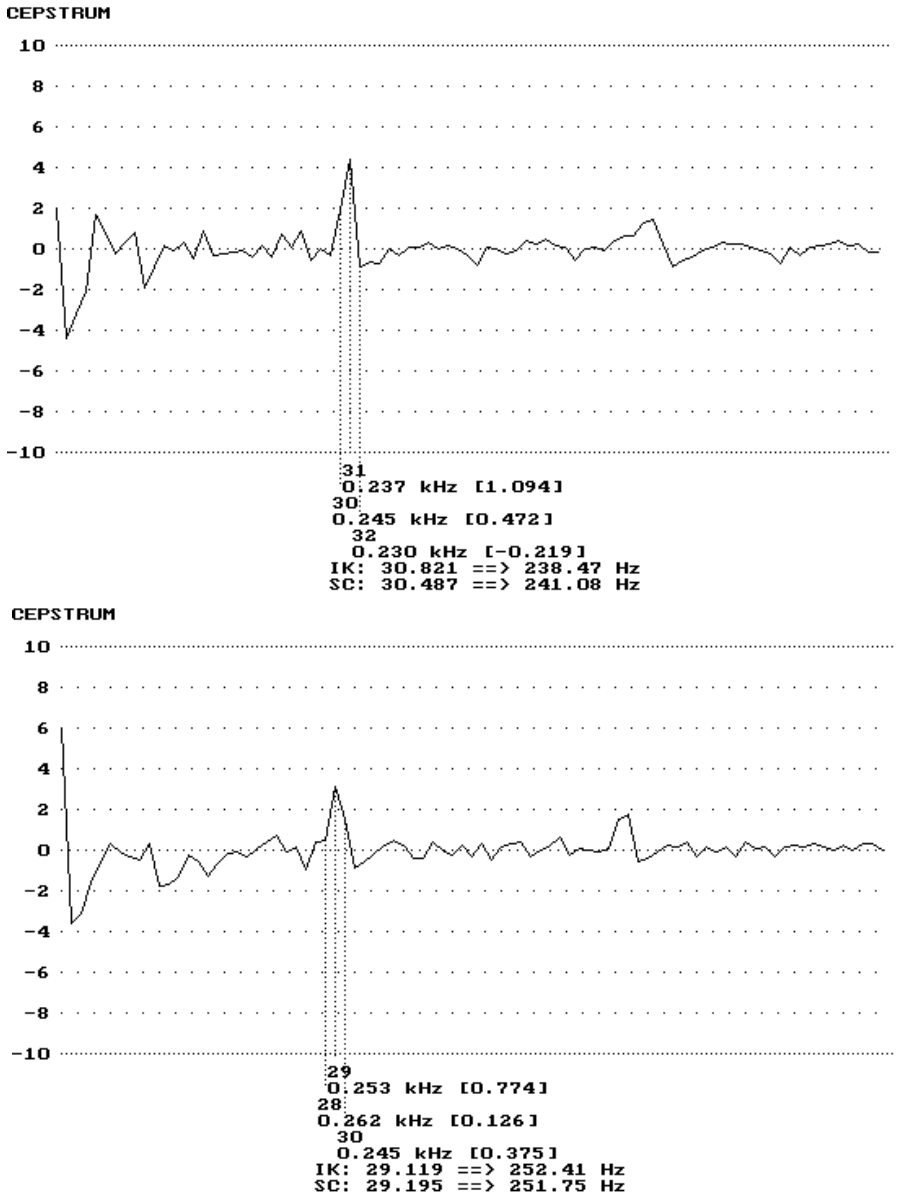
Rys. 7.5. Wynik analizy widmowej dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 3,6 kHz, bez preemfazy



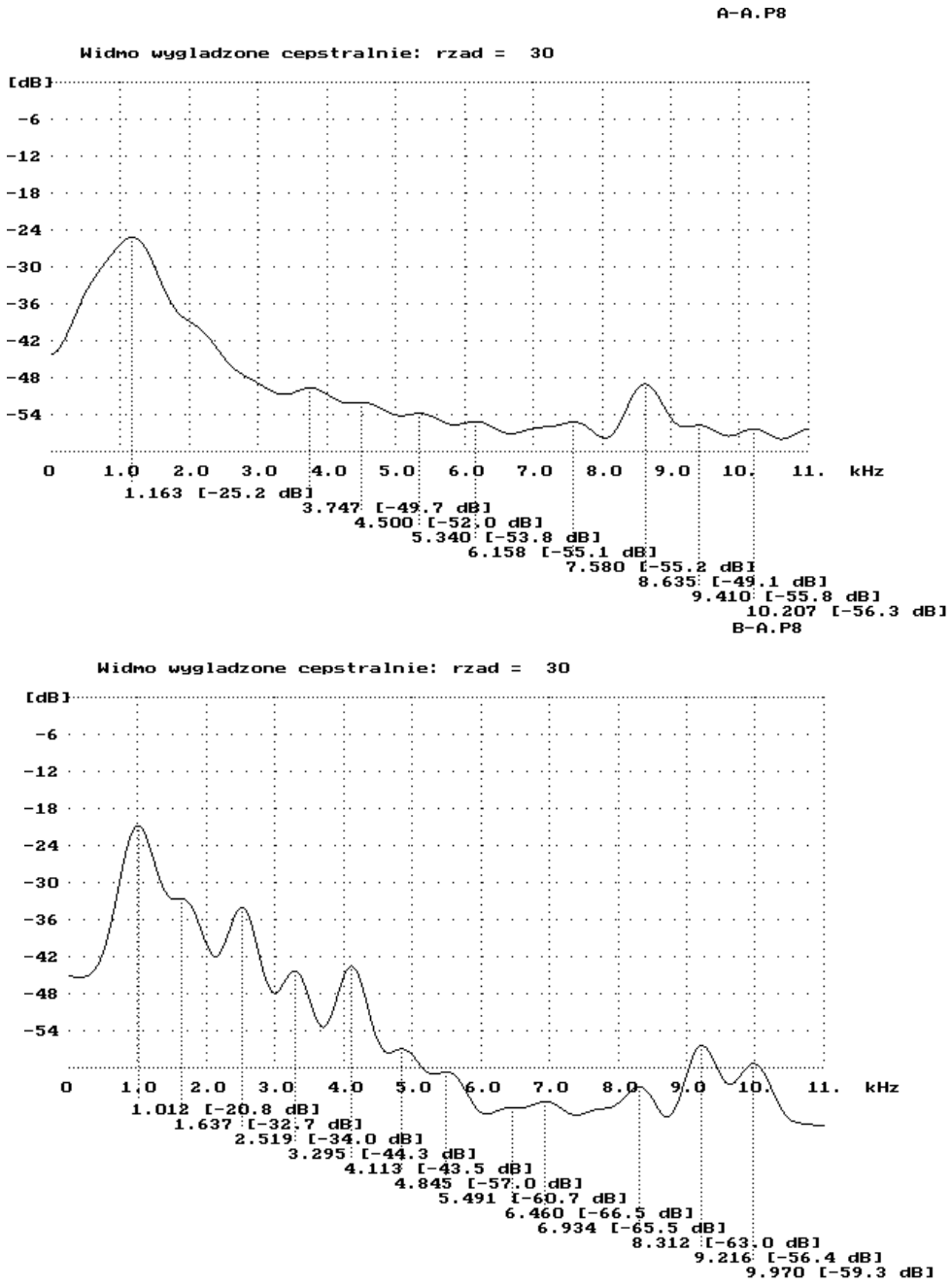
Rys. 7.6. Wynik analizy widmowej dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 11,025 kHz, preemfaza 6dB/oktawę



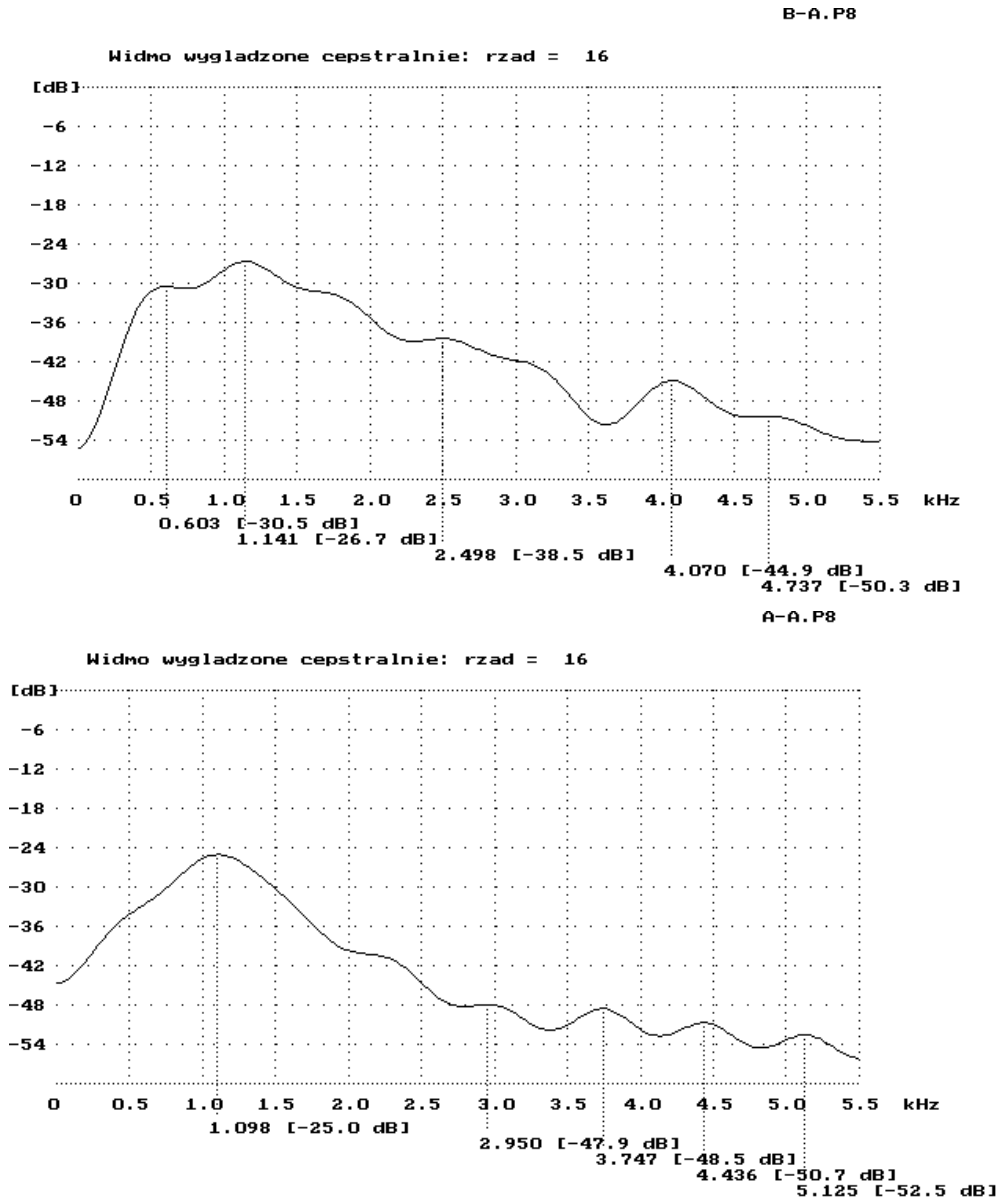
Rys. 7.7. Wynik analizy widmowej dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 3,6 kHz, preemfaza 6dB/oktawę



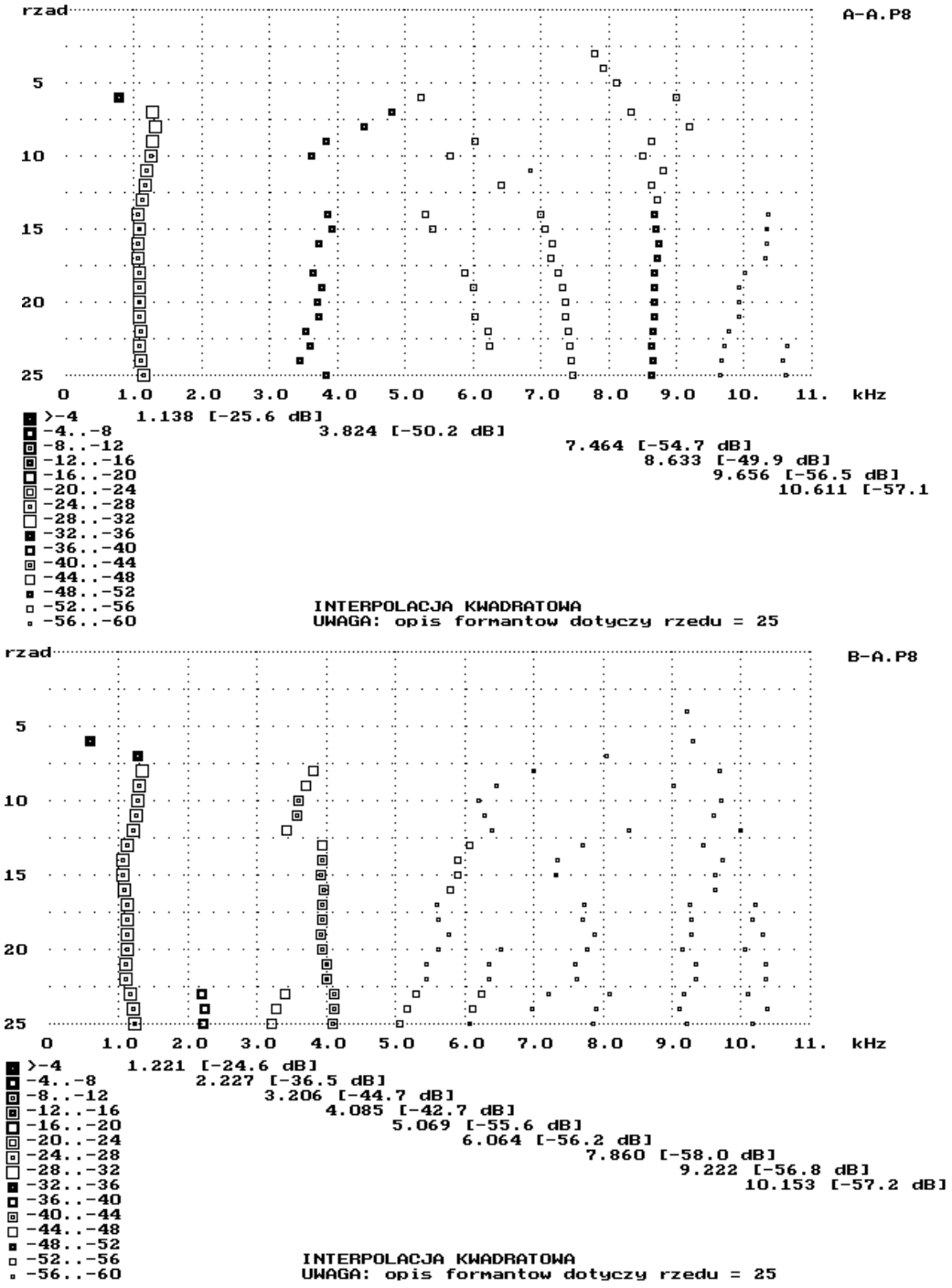
Rys. 7.8. Wykres cepstum mocy dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 3,6 kHz



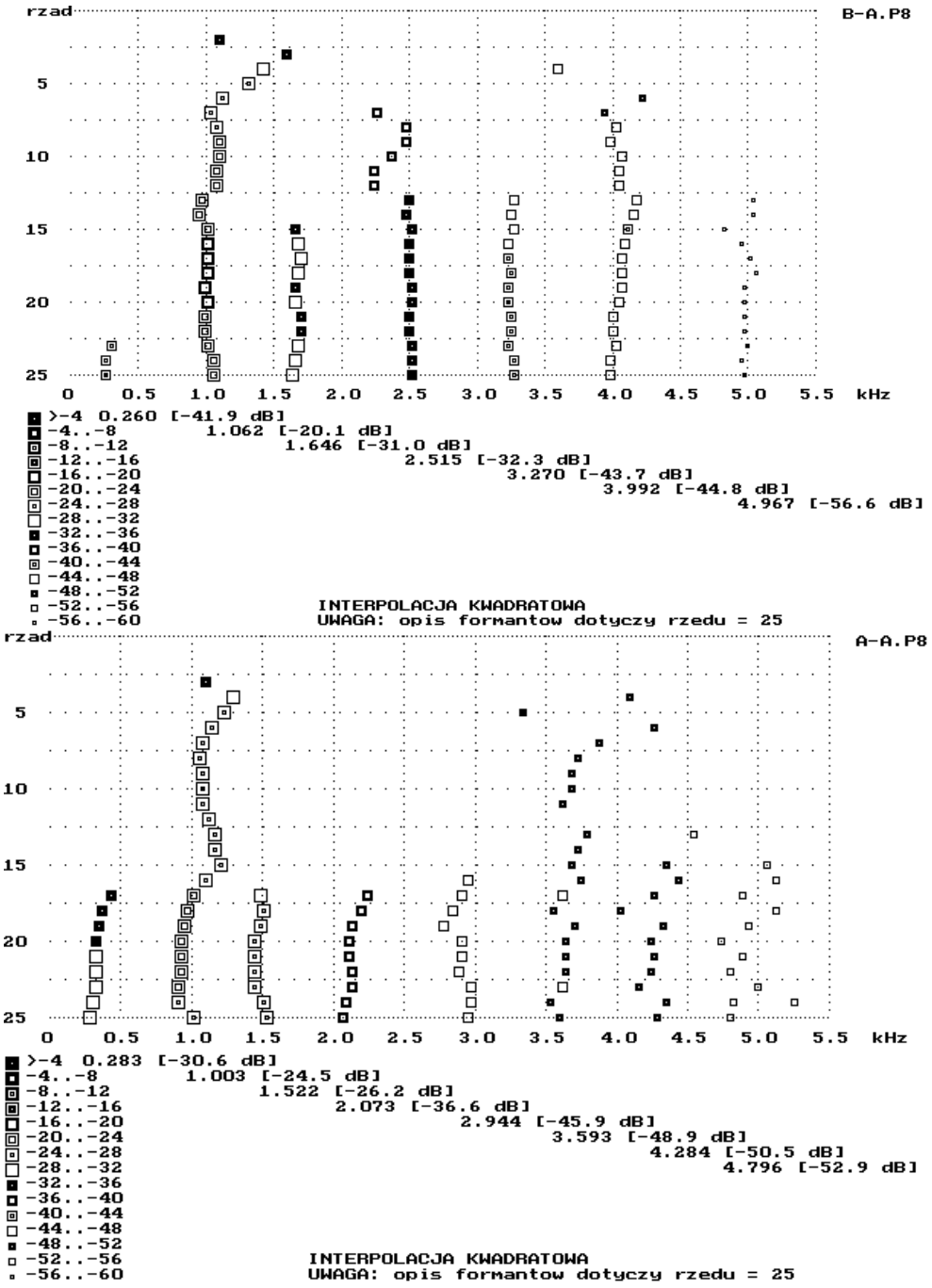
Rys. 7.9. Widmo wygładzone cepstralnie dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 11,025 kHz



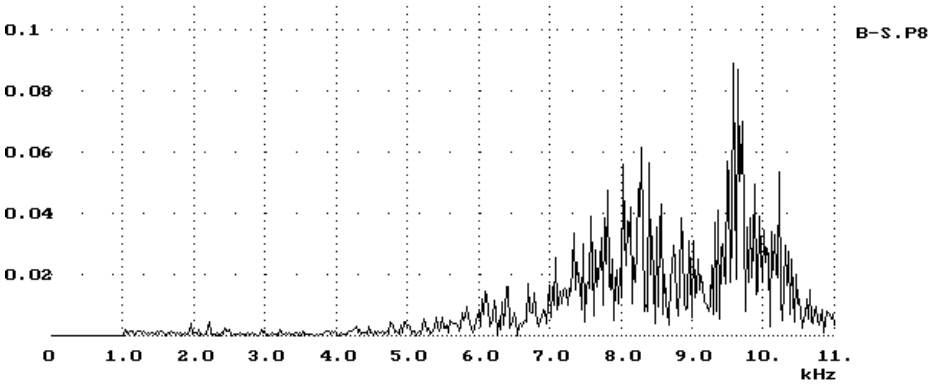
Rys. 7.10. Widmo wygładzone cepstralnie dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 5,5 kHz



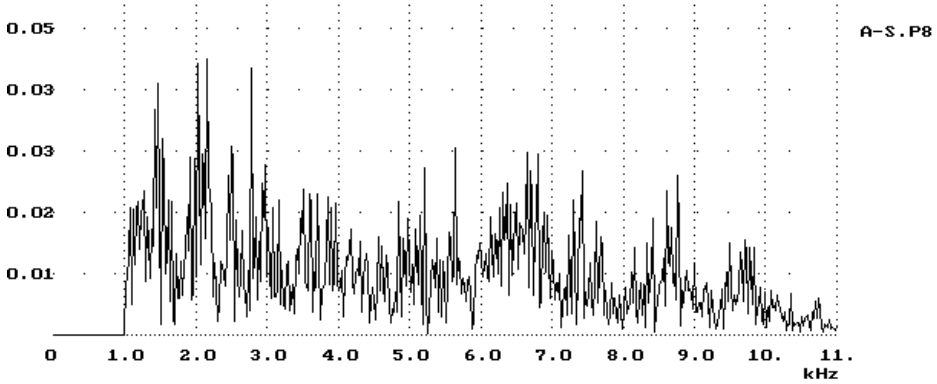
Rys. 7.11. Wykres zależności parametrów maksimów widma wygładzonego cepstralnie od rzędu wygładzania dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 11,025 kHz



Rys. 7.12. Wykres zależności parametrów maksimów widma wygładzonego cepstralnie od rzędu wygładzania dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej). Pasmo analizy 0 – 5,5 kHz

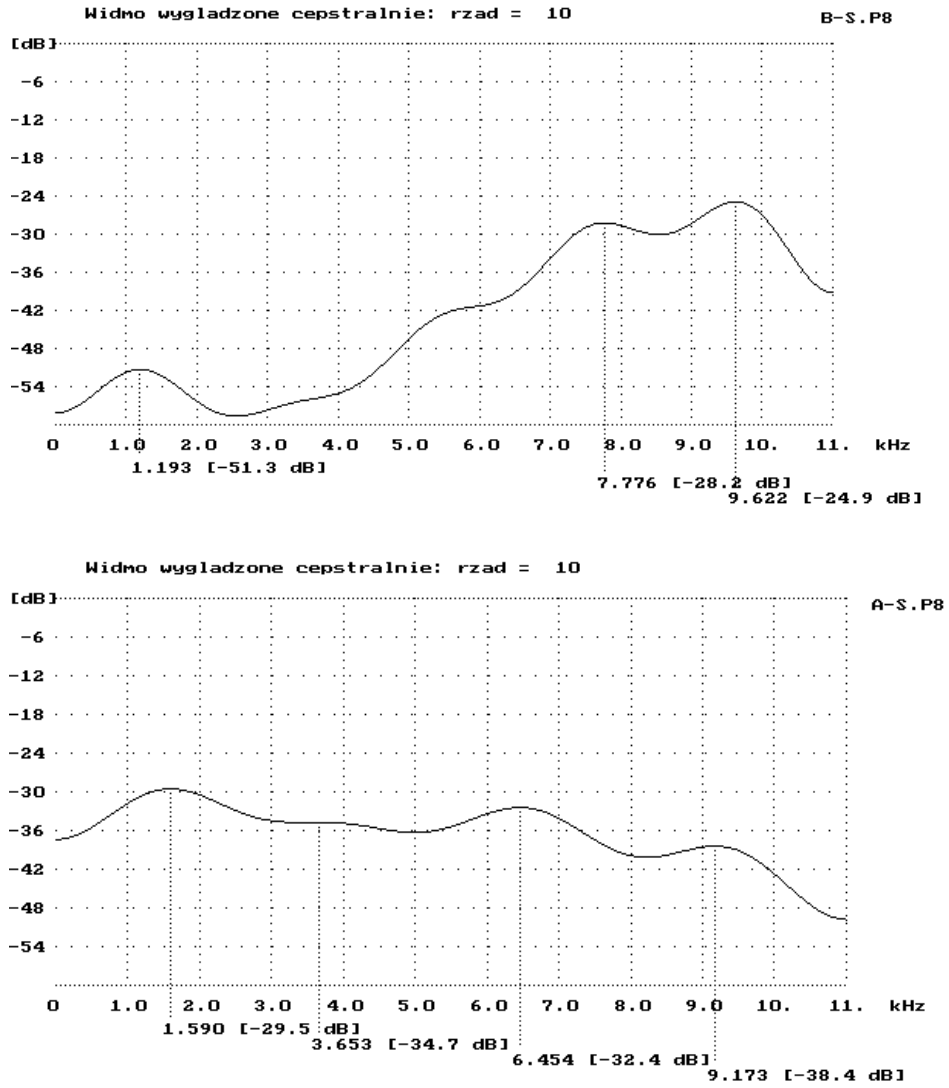


Ograniczenie pasma: 1012.06 Hz
MOMENT WIDMOWY RZEDU ZEROWEGO = 4.64 [rozdzielczosc analizy = 21.53 Hz]
MOMENT WIDMOWY UNORMOWANY RZEDU PIERWSZEGO = 8.40 kHz
MOMENT WIDMOWY CENTRALNY UNORMOWANY RZEDU DRUGIEGO = 2.895 kHz^2
[szer. widma = 1.701 kHz]
MOMENT WIDMOWY CENTRALNY UNORMOWANY RZEDU TRZECIEGO = -7.038 kHz^3
MOMENT WIDMOWY CENTRALNY UNORMOWANY RZEDU CZWARTEGO = 49.922 kHz^4
KURTOSIS = 5.957
SPECTRAL FLATNESS MEASURE = -14.050 dB

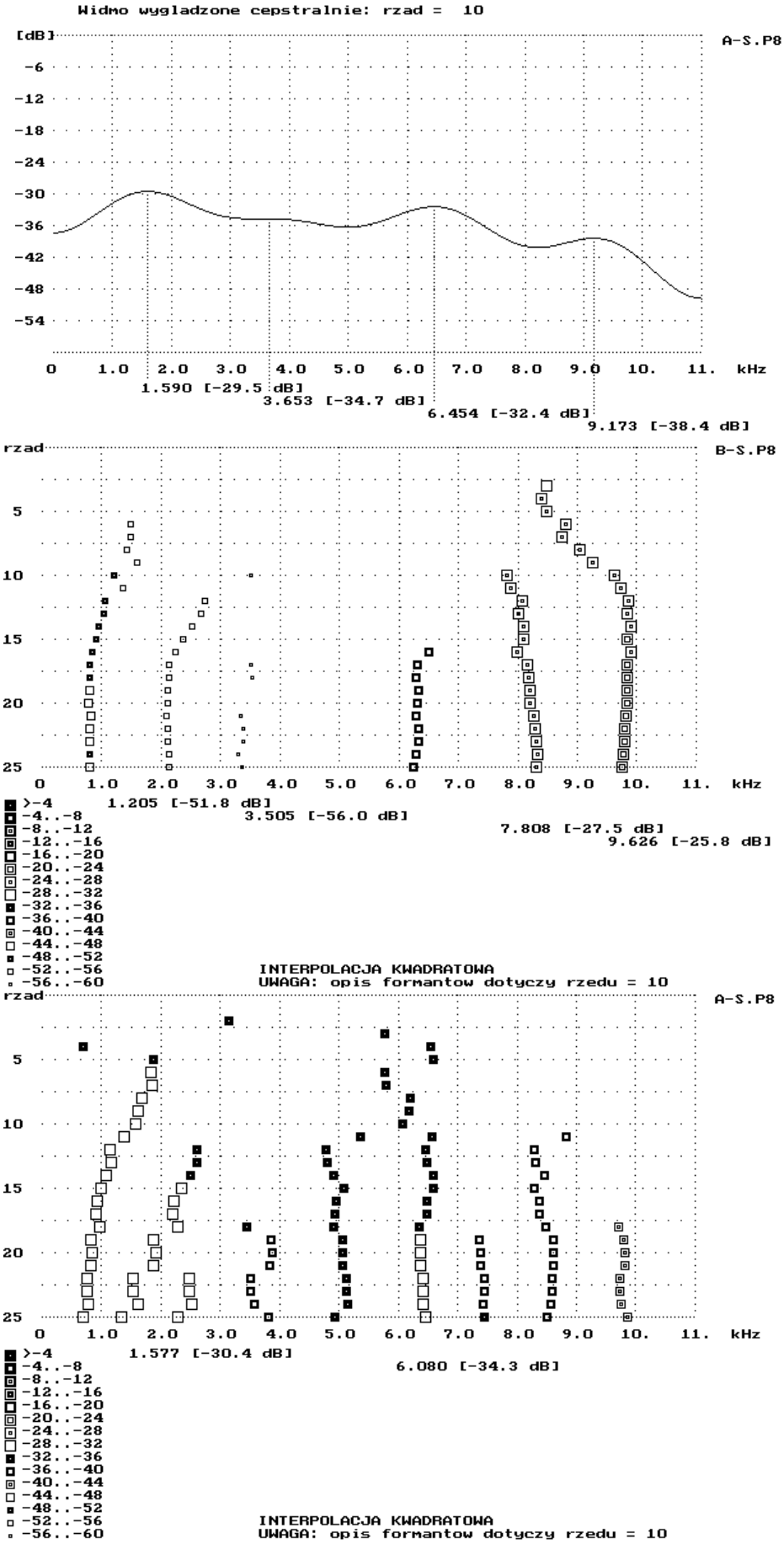


Ograniczenie pasma: 1012.06 Hz
MOMENT WIDMOWY RZEDU ZEROWEGO = 4.79 [rozdzielczosc analizy = 21.53 Hz]
MOMENT WIDMOWY UNORMOWANY RZEDU PIERWSZEGO = 5.04 kHz
MOMENT WIDMOWY CENTRALNY UNORMOWANY RZEDU DRUGIEGO = 7.160 kHz^2
[szer. widma = 2.676 kHz]
MOMENT WIDMOWY CENTRALNY UNORMOWANY RZEDU TRZECIEGO = 4.951 kHz^3
MOMENT WIDMOWY CENTRALNY UNORMOWANY RZEDU CZWARTEGO = 98.813 kHz^4
KURTOSIS = 1.928
SPECTRAL FLATNESS MEASURE = -4.605 dB

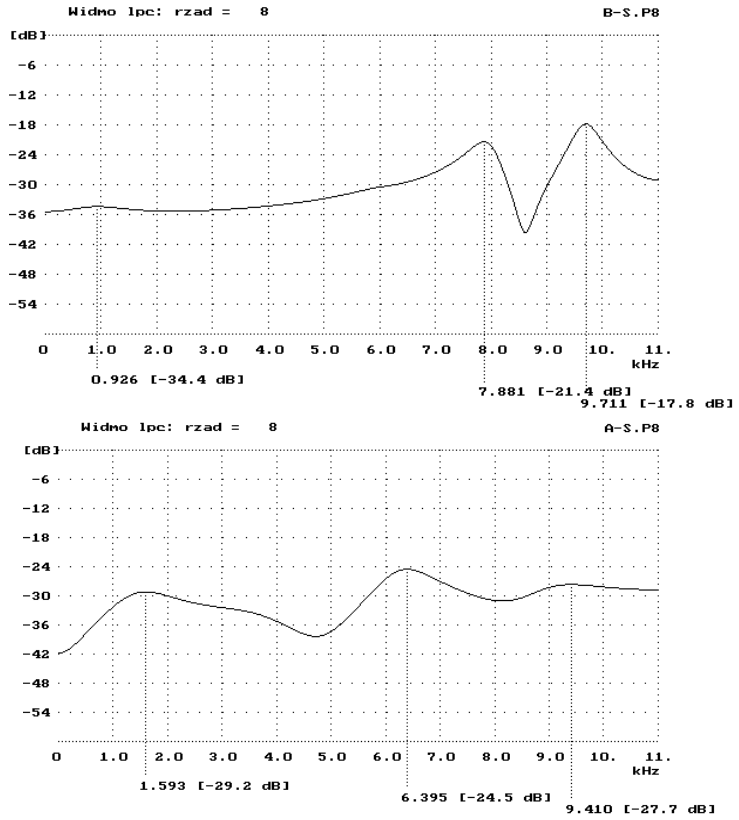
Rys. 7.13. Wyniki analizy widmowej i statystycznej dla głoski szumowej s wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)



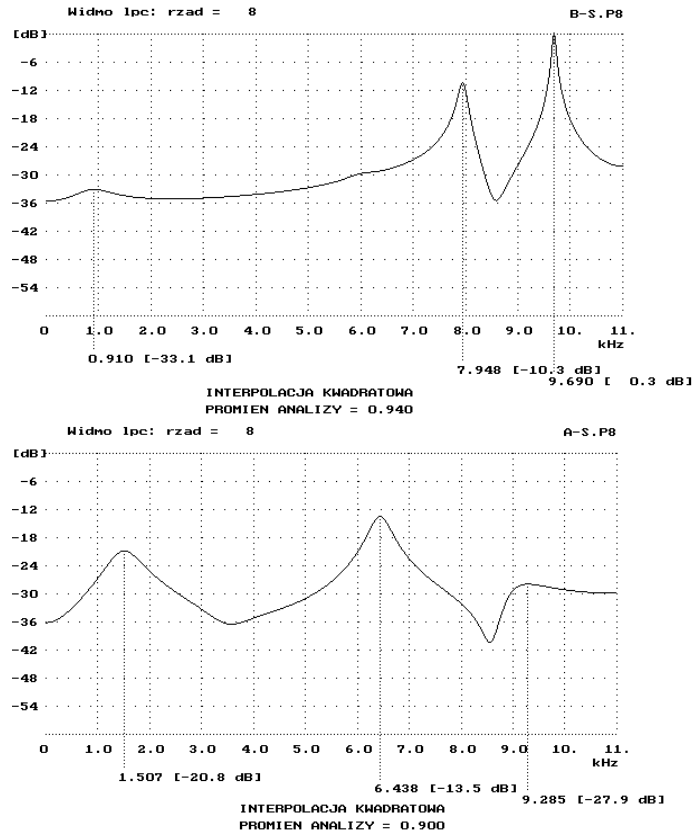
Rys. 7.14. Widmo wygładzone cepstralnie dla głoski szumowej s wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)



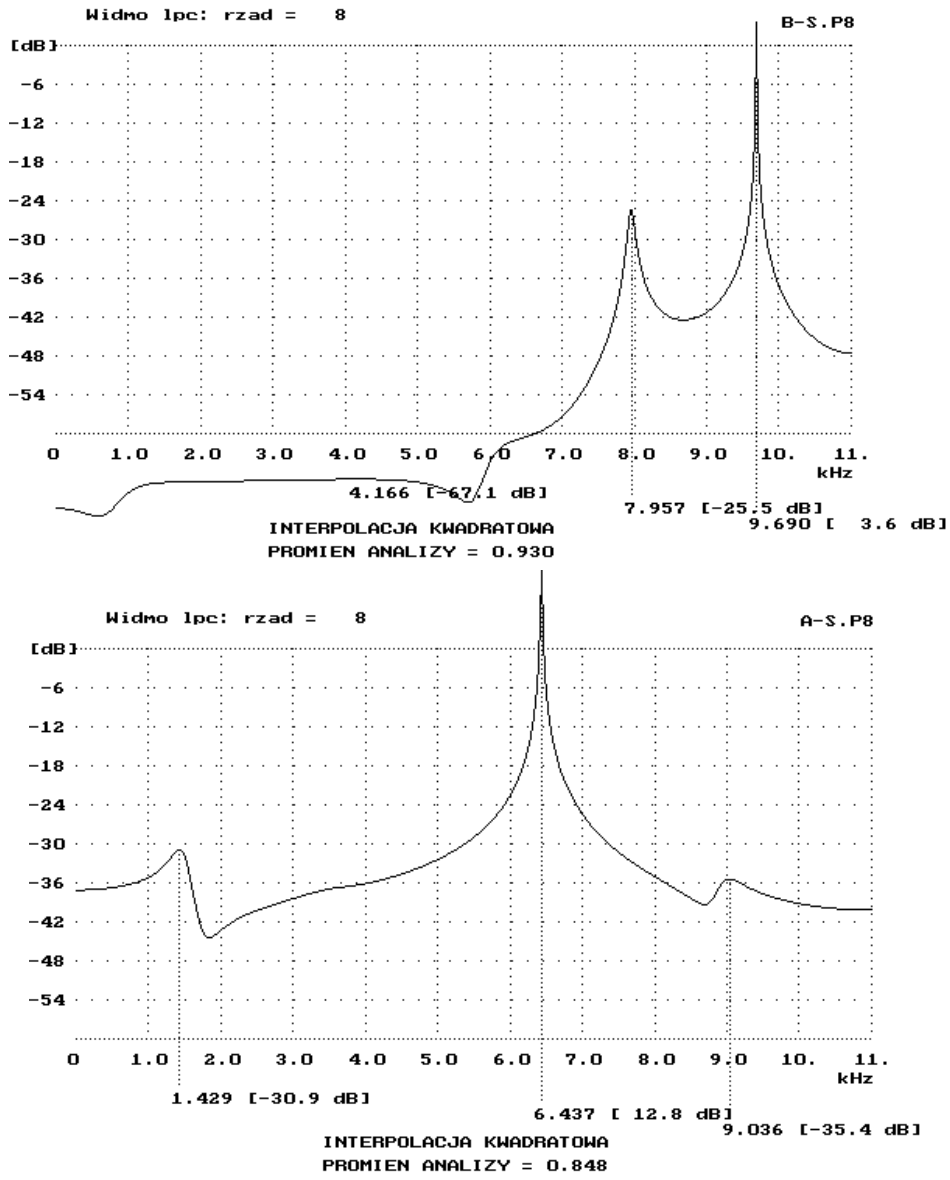
Rys. 7.15. Wykres zależności parametrów maksimów widma wygładzonego cepstralnie od rzędu wygładzania dla głoski szumowej s wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)



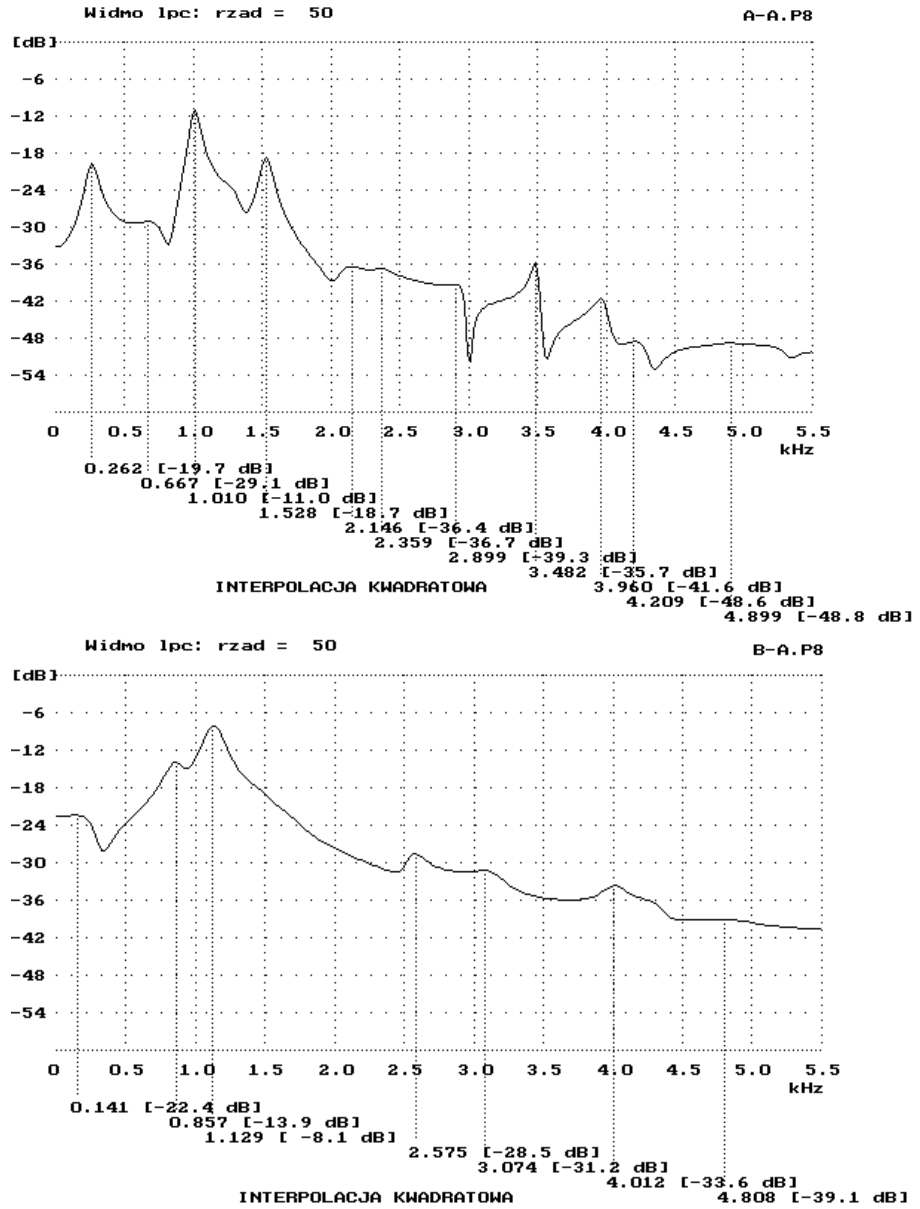
Rys. 7.16. Widmo LPC dla głoski szumowej s wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)



Rys. 7.17. Widmo LPC po zastosowaniu analizy wewnątrz koła jednostkowego płaszczyzny zespolonej dla głoski szumowej s wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)



Rys. 7.18. Widmo LPC po zastosowaniu analizy wewnątrz koła jednostkowego płaszczyzny zespolonej po przekroczeniu granicy stabilności dla głoski szumowej s wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)



Rys. 7.19. Widmo LPC dla samogłoski a wypowiedzianej poprawnie (powyżej) oraz wypowiedzianej przez osobę z rozszczepem podniebienia (poniżej)

Uzyskane szczegółowe wyniki pozwalają na ustalenie parametrów analizy, które mogą być wykorzystane w innych przypadkach głosów z problemami rozszczepu podniebienia. Ponadto na podstawie przeprowadzonych analiz możliwe jest rozróżnienie głosów prawidłowych i tych z zaburzeniami mowy. Ekstrahowane parametry analizy mogą być następnie wykorzystywane przy budowie bazy głosów z zaburzeniami, a następnie poddane analizie za pomocą algorytmów uczących.

7.7 Literatura

W temacie: klasyfikacji rozpoznawania aktywności osób z chorobą Parkinsona:

[1] Aminian K., Najafi B., Ca*pturing human motion using body-fixed sensors: outdoor measurement and clinical applications*, Computer Animation and Virtual Worlds 15, pp. 79-94, 2004.

[2] Baga D., Fotiadis D.I., Konitsiotis S., Maziewski P., Greenlaw R., Chaloglou D., Arrendondo M.T., Robledo M.G., Pastor M.A., *PERFORM: Personalised Disease Management for Chronic Neurodegenerative Diseases: The Parkinson's Disease and Amyotrophic lateral Sclerosis Cases*, eChallenges e-2009 Conf., 21-23 October 2009, Istanbul, Turkey.

[3] Bao L., Intille S.S., *Activity Recognition from User-Annotated Acceleration Data*, PERVASIVE 2004, LNCS 3001, pp. 1–17, 2004.

[4] Boser B.E., Guyon I., Vapnik V., *A training algorithm for optimal margin classifiers*. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144-152. ACM Press, 1992.

[5] Chang C., Lin C., *LIBSVM: a library for support vector machine*, March 2010. (http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf)

[6] Godfrey A., Conway R., Meagher D., Olaighin G., *Direct Measurement of Human Movement by Accelerometry*, Medical Engineering & Physics 30, pp. 1364-1386, 2008.

[7] Greenlaw R., Robledo M.G., Estrada J.J., Pansera M., Konitsiotis S., Baga D., Maziewski P., Pastor M.A., Papasava A., Chaloglou D., Zanichelli F., *PERFORM: Building and mining electronic records of neurological patients being monitored in the home*, World Congress on Medical Physics and Biomedical Engineering, 7-12 September 2009, Munich, Germany.

[8] WEKA software: http://weka.sourceforge.net/doc/weka/classifiers/rules/package-summary.html

[9] Huynh T., Schiele B., *Analyzing Features for Activity Recognition*, Joint sOc-EUSAI Conference, Grenoble, October 2005.

[10] Izworski A., Michałek M., Tadeusiewicz R., Rudzińska M., Bulka J., Wochlik I., *Acquisition and Interpretation of Upper Limbs Tremor Signal in Parkinsonian Disease,* Proceedings of the 4th WSEAS International Conference on Electronics, Signal Processing and Control (ESPOCO 2005), Rio de Janeiro, Brazil, April 25-27, pp. 81-85, 2005.

[11] Izworski A., Tadeusiewicz R., Rudzińska M., *Analysis and Classification of Tremor in Parkinsonian Disease in Two and Three Dimensional Space*, In: Callaos N., Lesso W., Savoie M.J., Zinn D. (eds.),

Proceedings of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando, vol. XV, pp. 203-207, 2004.

[12] Kupryjanow A., Kaszuba K., *Rozpoznawanie aktywności ruchowych pacjentów na podstawie analizy sygnałów pochodzących z trójosiowych czujników przyspieszenia*, PTETiS 2009, Zeszyty Naukowe Wydz. Elektrotechniki i Automatyki Pol. Gd., 26, 77-80, 2009.

[13] Kupryjanow A., Kaszuba K., Czyżewski A., *Influence of accelerometer signal pre-processing and classification method on human activity recognition*, Elektronika, 3,18-23, 2010.

[14] Kupryjanow A., Kostek B., *Rozpoznawanie ruchu rąk oraz chodu pacjentów na podstawie analizy sygnałów pochodzących z trójosiowych czujników przyspieszenia*, Zeszyty Naukowe Wydziału ETI Pol. Gd., 8, 215-218, 2010.

[15] Kupryjanow A., Kunka B., Kostek B., *UPDRS tests for Diagnosis of Parkinson's Disease Employing Virtual-Touchpad*, 4th International Workshop on Management and Interaction with Multimodal Information Content MIMIC, Bilbao, 2010.

[16] Kupryjanow A., Kunka B., Czyżewski A., *Virtual touchpad - video-based multimodal interface*, Zeszyty Naukowe Wydziału ETI Pol. Gd., 8, 219-224, 2010.

[17] Lee S.W., Mase K., *Activity and Location Recognitions Using Wearable Sensors*, Pervasive Computing July-September, pp. 24-32, 2002.

[18] Lombriser C., Bharatula N., Troste G., Roggen D., *On-body activity recognition in a dynamic sensor network*, Proceedings of the ICST 2nd International Conference on Body Area Networks, Article No. 17, Florence, Italy, 2007.

[19] Mathie M.J., Coster A.C.F., Lovell N.H., Celler B.G., *Accelerometry: providing an integrated, practical method for long-term, ambulatory monitoring of human movement*, Physiological measurement 25, pp. R1-R20, 2004.

[20] Maziewski P., Kupryjanow A., Kaszuba K., Czyżewski A., *Accelerometer signal pre-processing influence on human activity recognition*, Proc. Intern. Conf. NTAV/SPA, New Trends in Audio and Video, Signal Processing: Algorithms, Architectures, Arrangements and Applications, 95-99, Poznań, 24-25.2009.

[21] Maziewski P., Suchomski P., Kostek B., Czyżewski A., *An Intuitive Graphical User Interface for the Parkinson’s Disease Patients*, Proc. 4th Intern. IEEE EMBS Conf. on Neural Engineering, Antalya, Turkey, April 29 - May 2, 2009.

[22] Okuno R., Yokoe M., Akazawa K., Abe K., Sakoda S., *Finger Taps Movement Acceleration Measurement System for Quantitative Diagnosis of Parkinson's disease*, Proc. 28th Annual Intern. IEEE EMBS Conf., pp. 6623-6626, 2006.

[23] Prati A., Mikić I., Trivedi M., Cucchiara R., *Detecting moving shadows: formulation, algorithm and evaluation*, http://www.ivanamikic.com/TPAMIshadow.pdf

[24] Ravi N., Dandekar N., Mysore P., Littman M., *Activity Recognition from Accelerometer Data*, Proceedings of the Seventeenth Conference on Innovative Applications of Artificial Intelligence (IAAI-05), pp. 1541-1546, 2005.

[25] Rudzińska M., Gatkowska I., Mirek E., Szczudlik A., Po*radnik, Choroba Parkinsona, Leczenie farmakologiczne i rehabilitacja*,

http://www.neuro.cm-uj.krakow.pl/pacjenci/poradnik%20choroby%20Parkinsona.pdf.

[26] Shima K., Tsuji T., Kan E., Kandori A., Yokoe M., Sakoda S., *Measurement and Evaluation of Finger Tapping Movements Using Magnetic Sensors*, Proc. 30th Annual International IEEE EMBS Conference, pp. 5628-5631, 2008.

[27] Szczepaniak P.S., Tadeusiewicz R., *The Role of Artificial Intelligence, Knowledge and Wisdom in Automatic Image Understanding*. Journal of Applied Computer Science, Vol. 18, No. 1, pp. 75-85, 2010.

[28] *Shimmer, Sensing Health with Intelligence Modularity, Mobility and Experimental Reusability.* RealTime Technologies Manual, September 2008.

[29] Tadeusiewicz R., Gąciarz T., Borowik B., Leper B., *Odkrywanie właściwości sieci neuronowych*, Wydawnictwo Polskiej Akademii Umiejętności, Kraków 2007.

[30] Tadeusiewicz R., *Automatic Understanding of Signals*, Proc. Intelligent Information Systems, pp. 577-590, 2004.

[31] Tadeusiewicz R., *Neural network as a tool for medical signals filtering, diagnosis aid, therapy assistance and forecasting improving*, In Dössel O., Schlegel W.C. (eds.), IFMBE Proceedings, Vol. IV: Image processing, biosignals processing, modelling and simulation, biomechanics. Springer Verlag, Berlin, Heidelberg, New York, pp. 1532–1534, 2009.

[32] Tadeusiewicz R., *New Trends in Neurocybernetics*. Computer Methods in Materials Science, Vol. 10, No. 1, pp. 1-7, 2010.

[33] *The Unified Parkinson’s Disease Rating Scale (UPDRS): Status and Recommendations*. State of the Art Review, Movement Disorders; 18(7):738-750, 2003.

[34] Vapnik V., *The Nature of Statistical Learning Theory*, Springer, 1995.

[35] White D., Wagenaar R., Ellis T., *Monitoring Activity in Individuals with Parkinson Disease: A Validity Study*, J. of Neurologic Physical Therapy, vol. 30, No. 1, 2006.

[36] Żwan P., Kaszuba K., Kostek B., *Monitoring Parkinson's disease patients employing biometric sensors and rule-based data processing*, RSCTC'2010, 110-119, Warsaw, 2010.

[37] Kostek B., Kupryjanow A., *Wykorzystanie sieci neuronowych i metody wektorów nośnych SVM w procesie rozpoznawania aktywności ruchowej pacjentów dotkniętych chorobą Parkinsona*; Sieci neuronowe w zastosowaniach biomedycznych (Tadeusiewicz R., Duch W., Korbicz J., Rutkowski L., eds.), rozdział w monografii, pp. 285 - 308, 2013.

W temacie: parametryzacja sygnału mowy:

[38] Kaczmarek A., *Analiza sygnału mowy w foniatrii*, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej, Gdańsk 2006.

[39] Kłaczyński M., *Zjawiska wibroakustyczne w kanale głosowym człowieka*, Akademia Górniczo-Hutnicza, Kraków 2007.

[40] Liberek A., *Przygotowanie bazy nagrań mowy zniekształconej*, praca inż. pod kierunkiem B. Kostek, WETI, Pol. Gd., Gdańsk 2014

[41] Tadeusiewicz R., *Sygnał Mowy*, Wydawnictwa Komunikacji i Łączności, Warszawa 1988.

[42] WEKA software http://www.cs.waikato.ac.nz/ml/weka/

[43] Żwan P., rozprawa doktorska, *System ekspercki do obiektywizacji ocen głosów śpiewaczych*, promotor B. Kostek, WETI, Pol. Gd., Gdańsk 2007.

8 Podsumowanie

Ważnym celem niniejszego skryptu było przedstawienie metod uczenia się, rozumianych jako proces zmiany zachodzącej w systemie na podstawie doświadczeń, która prowadzi do poprawy jego jakości działania. Niniejszy skrypt zawiera przegląd podstawowych metod sztucznej inteligencji wraz z przywołaną terminologią i przykładami zastosowań. Należy podkreślić, że rozwiązywanie problemów z obszarów inżynierii biomedycznej i medycyny często wymaga wykorzystania metod uczących bądź systemów decyzyjnych czy ekspertowych. Intencją autorów skryptu było zaznajomienie studentów z zagadnieniami, które leżą u podstaw metod i algorytmów sztucznej inteligencji, usystematyzowanie nabytej wiedzy, wykorzystywanej później w ramach ćwiczeń laboratoryjnych, a także pozwalającej na samodzielne wykonanie projektu z przedmiotu „Sztuczna inteligencja w medycynie”.

W analizowanych przykładach z rozdziału 7, w których przedstawiono zastosowanie metod uczących się i wykorzystanie systemu parametryzacji sygnałów biomedycznych, widać, że opisane algorytmy, dzięki wysokiej skuteczności klasyfikacji oraz dużej skalowalności, mogą być wykorzystywane do monitoringu osób chorych lub stać się podstawą w projektowaniu algorytmów wyższego poziomu, np. służących do analizy symptomów choroby w przypadku osób z chorobą Parkinsona. Może się to przekładać na bardziej obiektywne miary oceny stanu pacjenta, bazujące na wynikach zaprojektowanych testów. Istotne w tym rozdziale było również przedstawienie metod akwizycji sygnałów i reprezentacji danych.

Osobnym aspektem, nie omawianym w niniejszym skrypcie jest problem tworzenia baz medycznych. Dlatego materiałem uzupełniającym do niniejszego skryptu powinny być materiały dotyczące tworzenia baz danych. W szczególności przy projektowaniu bazy należy uwzględnić wszystkie niezbędne funkcje związane z gromadzeniem i wykorzystaniem danych wprowadzanych przez użytkowników systemów oraz danych generowanych przez same systemy, jak np. automatyczne wstawianie danych przesyłanych np. przez systemy telemedyczne; wyszukiwanie danych w bazie według zadanych kryteriów; edycja danych w bazie – modyfikacja i usuwanie wpisów; tworzenie zestawień statystycznych; zarządzanie danymi jednostek posiadających dostęp do bazy danych; autoryzacja dostępu do bazy danych (hierarchiczność dostępu do baz danych); funkcje dodatkowe, jak również zapewnienie ochrony danych osobowych.

Należałoby również wspomnieć o zagadnieniach związanych z interoperacyjnością systemów medycznych, jak również ze standaryzacją zapisów sygnałów biomedycznych. Interoperacyjność definicyjnie oznacza uzgodnienie formatów danych i sposobów ich interpretacji, aby możliwe było wymienianie danych pomiędzy poszczególnymi modułami systemów medycznych oraz łączenie ich w kontekście rekordu pojedynczego pacjenta. Przykładem standardu zapewniającego scalanie danych

zbieranych z różnych źródeł jest standard HL7, natomiast w przypadku tego standardu problematyczne jest scalanie sygnałów biomedycznych. Warto zauważyć, że proces standaryzacji nie wpływa na indywidualne cechy użytkowe aparatury i nie ogranicza jej rozwoju, natomiast ma na celu zapewnienie łatwej wymiany i przekodowania zapisów sygnałów np. archiwalnych, kodowanie aktualnie rejestrowanych sygnałów i adekwatny opis sygnałów wprowadzanych do diagnostyki medycznej. Pozytywną cechą niektórych standardów jest też kompatybilność z innym standardami oraz możliwość stosowania innych standardów wymiany informacji medycznych. Do takich należy dla przykładu standard MFER (ang. *Medical Waveform Format Encoding Rules)*.

I wreszcie warto zauważyć, że podczas projektowania algorytmów opartych na klasyfikatorach wyższego rzędu (np. sztuczne sieci neuronowe, SVM, itd.), często konieczne jest wykonanie szybkich testów pozwalających na określenie, który z klasyfikatorów daje najlepsze rezultaty w przypadku danego zadania klasyfikacji. Ważna jest także możliwość szybkiego sprawdzenia, które parametry opisujące dany problem pozwalają na skuteczną klasyfikację danych wejściowych. Narzędziem umożliwiającym wykonanie tego typu badań jest system WEKA. Warto wspomnieć, że system ten nie został stworzony specjalnie do tego celu. Jego głównym zadaniem jest wspieranie procesów eksploracji danych (ang. *data mining*). System został napisany w języku Java, a klasy, z których korzysta, są ogólnie dostępne i można wykorzystywać je we własnych projektach. Dlatego zapoznanie się studentów z tym środowiskiem jest celem pośrednim tego przedmiotu.

Wykładowi w przedmiocie: Sztuczna inteligencja w medycynie towarzyszą laboratorium oraz projekt. W ramach ćwiczeń laboratoryjnych poruszane w niniejszym skrypcie zagadnienia znajdują pełne odzwierciedlenie w poszczególnych ćwiczeniach laboratoryjnych. Podstawowymi narzędziami wykorzystywanymi w ćwiczeniach laboratoryjnych są środowisko MATLAB, WEKA oraz system RSES. Ćwiczenia laboratoryjne obejmują wprowadzenie do systemu WEKA, badanie algorytmów i struktur sieci neuronowych, pracę z systemem RSES, projektowanie prostych systemów logiki rozmytej, zastosowanie algorytmów genetycznych w zagadnieniach optymalizacji danych oraz rozpoznawanie stanu pacjenta dotkniętego chorobą Parkinsona na podstawie zarejestrowanego sygnału z wykorzystaniem SVM. Ostatnim elementem przedmiotu jest projekt, w ramach którego studenci przetwarzają dane biomedyczne pacjentów z wybranych baz udostępniających takie dane. Oba te dodatkowe elementy pozwalają więc na wykorzystanie przytoczonych zagadnień teoretycznych w praktycznych zastosowaniach.