UNIWERSYTET WARSZAWSKI
Wydział Matematyki, Informatyki i Mechaniki

***Michał Mikołajczyk***
Nr albumu: 171827

# Redukcja liczby reguł decyzyjnych

Praca magisterska
na kierunku MATEMATYKA

Praca wykonana pod kierunkiem
*prof. dr hab. Andrzeja Skowrona*
Instytut Matematyki

Warszawa, maj 2001 r.

Pracę przedkładam do oceny

| Data | Podpis autora pracy |
| --- | --- |

Praca jest gotowa do oceny przez recenzenta

| Data | Podpis kierującego pracą |
| --- | --- |

**Streszczenie:** W niniejszej pracy przedstawiony został problem redukcji liczby reguł decyzyjnych niezbędnych do uzyskania aproksymacyjnego opisu pojęć o zadowalającej jakości. Zawiera ona opis konstrukcji systemu decyzyjnego opartego na niewielkiej liczbie reguł, które powstały w wyniku uogólnienia zbioru wszystkich niesprzecznych reguł decyzyjnych. Zgodnie z zasadą minimalnego opisu naturalne jest dążenie do redukcji liczby reguł, prowadzącej do krótszego opisu pojęć (patrz [13]). Na końcu pracy zostały przedstawione wyniki eksperymentów pozwalające na porównanie przedstawionego systemu z innymi znanymi z literatury systemami decyzyjnymi.

**Słowa kluczowe:** system decyzyjny, reguły decyzyjne, klasyfikacja, indukcyjne definiowanie pojęć

**Klasyfikacja tematyczna**[1]**:** 94A17, 03B48, 62C20, 60J20, 68T05, 68W20, 68W25

Praca była realizowana w ramach grantu KBN 8T1102519.

## Podziękowania

Chciałbym gorąco podziękować mojemu opiekunowi naukowemu prof. dr hab. Andrzejowi Skowronowi, za wszystkie uwagi i bezcenne rady podczas pisania niniejszej pracy.

Chciałbym także podziękować dr Janowi G. Bazanowi za udostępnienie programu liczącego i skracającego reguły, z którego korzystałem w przeprowadzonych eksperymentach.

[1] według AMS Mathematical Subject Classification 2000

# Spis treści

# 1 Wstęp

Coraz częściej zdajemy sobie sprawę, że informacje (różnego rodzaju) odgrywają w naszym życiu coraz większą rolę. Staramy się je gromadzić i analizować. Jednak im więcej zgromadzimy informacji, tym więcej mamy z nimi kłopotów. Nadchodzi zawsze taki moment, w którym okazuje się, że zgromadziliśmy zbyt wiele danych i nie jesteśmy w stanie poradzić sobie ręcznie z ich analizą. Wtedy sięgamy po systemy decyzyjne, które pozwalają na automatyczną analizę zebranych informacji.

Postawmy się w sytuacji naukowca, który bada reakcję delfinów na rozmaite dźwięki. Kilka prostych eksperymentów pokazuje, że zwierzęta te reagują na pewne dźwięki, na inne zaś – nie. Naukowiec stawia sobie pytanie dlaczego tak jest i postanawia sprawdzić od czego to zależy przeprowadzając całą serię eksperymentów. W tym celu wyznaczył 30 cech charakteryzujących każdy dźwięk, a następnie prezentował je kolejno delfinom, notując ich reakcję lub jej brak. Szybko okazało się, że reakcja badanych ssaków nie zależy od żadnej pojedynczej cechy. Oznacza to, że dopiero pewne ich zestawy (podzbiory) decydują o tym czy „spodobają" się delfinom, czy też – nie.

Załóżmy, że każda z cech ma tylko dwie wartości: 1 – cecha występuje i 0 – cecha nie występuje. Wszystkich cech (zwanych dalej atrybutami) jest 30. Zatem aby posiadać pełną wiedzę o reakcji delfinów na dźwięki należy przeprowadzić $2^{30}$ doświadczeń. Jeśli będziemy wykonywali jedno doświadczenie na sekundę, wówczas zajmie to nam 34 lata ! Po wykonaniu tego prostego wyliczenia nasz naukowiec postanawia wykonać około 10 tysięcy doświadczeń z losowym przydziałem cech, co zajmie mu jeden dzień, a następnie otrzymane dane wprowadzić do systemu decyzyjnego.

Po wprowadzeniu danych i ich automatycznej obróbce, naukowiec może zadawać systemowi decyzyjnemu pytania: „Czy delfiny zareagują na następujący zestaw cech:...?". Po zautomatyzowaniu tego procesu komputer był w stanie odpowiadać na 100 takich pytań w ciągu jednej sekundy. Każda odpowiedź klasyfikowała podany zestaw cech jako te, na które delfin zareaguje, lub przeciwnie jako te na które nie zareaguje. Na przetworzenie wszystkich możliwych kombinacji trzeba by jednak czekać 4 miesiące. Po upływie tego czasu stałoby przed nami nowe wyzwanie: ogromna liczba pytań w postaci zestawów cech i odpowiedzi przydzielonych przez maszynę. Dodatkowym utrudnieniem byłby fakt, że system decyzyjny dysponował zaledwie niewielkim podzbiorem wszystkich możliwości, zatem nie był w stanie stworzyć dokładnego modelu obejmującego całość zagadnienia. Oznacza to, że nie można uniknąć błędów, które utrudniałyby dalszą analizę otrzymanych wyników.

Naukowiec postanawia więc wyciągnąć z systemu decyzyjnego reguły, przy pomocy których system ten dokonuje klasyfikacji dźwięków. Reguły te mają postać: {Zbiór cech} $\Rightarrow$ Decyzja. Niestety jest ich bardzo dużo i ich analiza jest bardzo trudna. Załóżmy jednak, że udało się zredukować liczbę reguł tak, że nadal stosunkowo dobrze opisują reakcje delfinów. Teraz można się im dokładniej przyjrzeć (jest ich bowiem stosunkowo niewiele) i wyciągnąć wnioski. Na drugi dzień z uzyskanymi w ten sposób nowymi hipotezami, naukowiec udaje się do oceanarium, aby sprawdzić je w praktyce. Wystarczy zaledwie kilka doświadczeń, aby potwierdzić lub zaprzeczyć naszym przypuszczeniom. Część hipotez może okazać się nieprawdziwa, jednak wiele z nich znajdzie potwierdzenie w praktyce. W ten oto sposób nasz naukowiec nie potrzebował 34 lat, ani nawet 4 miesięcy, lecz zaledwie paru dni na zanalizowanie postawionego zagadnienia.

Od dziesięcioleci ludzie poszukują metod aproksymacyjnego opisu pojęć (zobacz np. [5]–[9]), które określone są jedynie przez wyniki empirycznych doświadczeń, chcąc je w ten sposób lepiej poznać i zrozumieć. Poszukiwany opis powinien nie tylko dobrze odzwierciedlać rzeczywistość lecz również być możliwie najprostszy. Uproszczenie opisu, nawet

kosztem jego jakości – czyli dopuszczenia niewielkiego błędu, jest również bardzo wartościowe, gdyż może umożliwić nie tylko lepsze zrozumienie opisanego pojęcia, ale często również prowadzi do aproksymacyjnego opisu pojęcia o lepszej jakości. Znalezienie równowagi pomiędzy jakością opisu, a stopniem jego skomplikowania jest bardzo trudne. Rozwój techniki, dzięki której mamy dostęp do coraz większych ilości informacji powoduje, że coraz częściej spotykamy się z potrzebą aproksymacyjnego opisu otaczającej nas rzeczywistości.

W poniższej pracy chciałbym przedstawić problem redukcji liczby reguł decyzyjnych. Jest to istotny kierunek mający na celu usprawnienie analizy danych. Przedstawię przykłady istniejących rozwiązań tego problemu wraz z możliwościami ich rozbudowy, oraz propozycję systemu decyzyjnego, który nie tylko redukuje liczbę reguł decyzyjnych, lecz również dokonuje ich uogólnienia. Na końcu pracy przedstawię wyniki licznych doświadczeń i dokonam porównania wszystkich przedstawionych w pracy systemów decyzyjnych.

# 2 Wprowadzenie

## 2.1 Zarys problemu

Zakładamy, że dostępne są pewne dane, zgromadzone poprzez wielokrotne wykonanie pewnego doświadczenia. Dane te są przedstawione w postaci *tablicy decyzyjnej*:

| – | $a_1$ ... $a_N$ | d |
|---|---|---|
| $x_1$ | | $d_1$ |
| $\vdots$ | | $\vdots$ |
| $x_{k+m}$ | | $d_{k+m}$ |

Kolejne doświadczenia zostały oznaczone jako $x_1, \ldots, x_{k+m}$ i stanowią zbiór przykładów $\mathbb{X}$. Atrybuty $a_1, \ldots, a_N$ opisują parametry doświadczenia, zaś $d_j$ jego wyniki (*klasę decyzyjną*). Zbiór wszystkich możliwych decyzji (klas decyzyjnych) oznaczamy literą $\mathbb{D}$. Na podstawie tak przygotowanych danych potrafimy wyznaczyć wszystkie możliwe *reguły decyzyjne* mające postać koniunkcji:

$$(a_{1_r} = w^r_{1_r} \wedge \ldots \wedge a_{N_r} = w^r_{N_r}) \Rightarrow \text{KlasaDecyzyjna}.$$

Gdzie $1 \leqslant 1_r < \ldots < N_r \leqslant N$. Liczba użytych atrybutów, oraz ich wartości (oznaczone jako: $w^r_{i_r}$) są specyficzne dla danej reguły $r$. Każda reguła wyznacza funkcję $r\colon \mathbb{X} \to \{d^r, d^?\}$ określającą przynależność obiektu $x \in \mathbb{X}$ do klasy decyzyjnej $d^r \in \mathbb{D}$, którą potrafi rozpoznawać ta reguła. Funkcja ta określona jest następująco:

$$r(x) = \begin{cases} d^r & \text{gdy warunek } (a_{1_r}(x) = w^r_{1_r} \wedge \ldots \wedge a_{N_r}(x) = w^r_{N_r}) \text{ jest spełniony,} \\ d^? & \text{gdy warunek } (a_{1_r}(x) = w^r_{1_r} \wedge \ldots \wedge a_{N_r}(x) = w^r_{N_r}) \text{ nie jest spełniony.} \end{cases}$$

$$a_i(x) = \text{wartość i-tego atrybutu w obiekcie } x$$

Jeśli $r(x) = d^?$ to przy użyciu tej reguły nie potrafimy nic powiedzieć o przynależności $x$ do jakiejkolwiek klasy decyzyjnej ze zbioru $\mathbb{D}$.

*Systemem decyzyjnym* nazywamy algorytm, który korzystając z dostępnych już reguł potrafi dokonać *klasyfikacji* pewnego obiektu (wektora wartości atrybutów) $x_*$ do klasy decyzyjnej ze zbioru $\mathbb{D}$. Sprawa jest prosta, gdy wszystkie reguły wskazują jedną klasę decyzyjną, to znaczy: $\exists_{d^*} \forall_i \left(r_i(x_*) = d^* \vee r_i(x_*) = d^?\right)$, oraz: $\exists_i\, r_i(x_*) \neq d^?$. Często jednak

obiekt $x_*$ będzie rozpoznawany przez reguły reprezentujące różne klasy decyzyjne. System decyzyjny ma za zadanie rozstrzygać takie konflikty.

Najprostszą strategią rozstrzygania konfliktów jest głosowanie. Każda reguła, jeśli rozpozna obiekt $x_*$ (czyli $r(x_*) = d^r$), oddaje głos na reprezentowaną klasę decyzyjną ($d^r$). Regułom możemy przypisać pewne wagi i to właśnie te wagi brać pod uwagę podczas głosowania. Na przykład jeśli nasza reguła ma wagę $\frac{1}{2}$ wówczas może oddać $\frac{1}{2}$ głosu, jeśli zaś ma wagę 2 wówczas może oddać aż dwa głosy. Nadawanie regułom wag jest stosowane, aby wyróżnić pewne reguły, które uważamy za lepsze, oraz aby wyrównać szanse klas decyzyjnych, które mają mniej reguł od pozostałych.

Chcielibyśmy, korzystając z systemu decyzyjnego, móc dokonywać klasyfikacji obiektów, które nie znajdują się w naszym zbiorze przykładów $\mathbb{X}$. Dzięki temu moglibyśmy przewidywać wyniki kolejnych, często kosztownych, doświadczeń bez konieczności ich wykonywania. Naturalnie musimy pogodzić się z faktem, iż nasz system decyzyjny może popełniać czasem błędy lub nie być w stanie dokonać klasyfikacji pewnych obiektów.

Jeśli nasz system decyzyjny korzysta ze wszystkich reguł, wówczas może okazać się, że klasyfikacja zajmuje bardzo dużo czasu. Wynika to z konieczności sprawdzenia spełnialności wielu koniunkcji.

Możemy zdecydować się na rezygnację z części reguł. Dzięki temu skraca się czas klasyfikacji. Jednocześnie pozbywając się najsłabszych reguł upraszczamy problem rozstrzygania konfliktów, dzięki czemu zyskujemy na skuteczności systemu. Niestety pozbywanie się reguł może doprowadzić do sytuacji, w której nasz system decyzyjny bardzo często nie będzie potrafił sklasyfikować nowych obiektów, a to bardzo poważna wada.

W rozdziale 4 przedstawiam stosowane obecnie metody redukcji reguł, zaś w rozdziale 5 przedstawiam propozycję systemu opartego nie na pozbywaniu się reguł, lecz na ich uogólnianiu i łączeniu.

Pierwszą rzeczą, jaką należy wykonać przy redukcji liczby reguł decyzyjnych, jest ocena ich jakości. Abyśmy mogli dokonać obiektywnej oceny podzielimy zbiór dostępnych przykładów na dwie części: $x_1, \ldots, x_k$ – z nich wygenerujemy reguły decyzyjne $(r_1, \ldots, r_n)$ i $x_{k+1}, \ldots, x_{k+m}$ – one posłużą do testowania skuteczności reguł. Tak więc naszą tablicę decyzyjną można przedstawić w następujący sposób:

| – | $a_1$ | $\ldots$ | $a_N$ | d |
|---|---|---|---|---|
| $x_1$ | | | | $d_1$ |
| $\vdots$ | | | | $\vdots$ |
| $x_k$ | | | | $d_k$ |
| $x_{k+1}$ | | | | $d_{k+1}$ |
| $\vdots$ | | | | $\vdots$ |
| $x_{k+m}$ | | | | $d_{k+m}$ |

Stosując kolejne reguły do wszystkich przykładów otrzymujemy następującą tabelę:

| – | $r_1$ | $\ldots$ | $r_n$ | d |
|---|---|---|---|---|
| $x_1$ | | | | $d_1$ |
| $\vdots$ | | | | $\vdots$ |
| $x_k$ | | | | $d_k$ |
| $x_{k+1}$ | | | | $d_{k+1}$ |
| $\vdots$ | | | | $\vdots$ |
| $x_{k+m}$ | | | | $d_{k+m}$ |
| $d^r$ | $d^{r_1}$ | $\ldots$ | $d^{r_n}$ | – |

Zakładamy, że powyższa tablica jest niesprzeczna, to znaczy nie ma w niej dwóch wierszy, które mają te same wartości dla $a_1, \ldots, a_N$ ale różne wartości $d$. Wtedy można rozważać reguły niesprzeczne na zbiorze, z którego zostały wyznaczone. W szczególności można rozważać minimalne reguły decyzyjne [2], to znaczy zawierające małą liczbę czynników koniunkcji po lewej stronie reguły gwarantującą zachowanie niesprzeczności.

Jednak reguły te zastosowane do $x_{k+1}, \ldots, x_{k+m}$ mogą popełniać błędy. Dzieje się tak dlatego, że zostały wygenerowane nie ze wszystkich możliwych przykładów (wszystkich możliwych wartościowań atrybutów) lecz z pewnego niewielkiego podzbioru. Na podstawie tego podzbioru $(x_1, \ldots, x_k)$ reguły aproksymują zachowanie się klas decyzyjnych na zbiorze wszystkich możliwych przykładów.

## 2.2 Systemy decyzyjne

Klasyfikacja nowych obiektów może odbywać się na wiele sposobów. Każda ze znanych metod ma swoje wady i zalety, oraz każda może okazać się lepsza od innych dla pewnych specyficznych danych. Należy pamiętać, że system decyzyjny jest jedynie pewną aproksymacją i nie istnieje system rozwiązujący wszystkie problemy, który nie popełniałby żadnych błędów.

### 2.2.1 System KNN

KNN, czyli $k$ najbliższych sąsiadów, to jeden z najprostszych systemów decyzyjnych. Jego zastosowanie nie wymaga żadnych przygotowań, ani uczenia się z danych początkowych. Klasyfikacja odbywa się poprzez wybór $k$ najbliższych sąsiadów badanego obiektu $x_*$. Jeżeli przyjmiemy, że $k$-ty, najbardziej oddalony przykład od $x_*$, znajduje się od niego w odległości $d^k_{KNN}$, wówczas wybieramy wszystkie przykłady $x$ takie, że: $d_{KNN}(x, x_*) \leqslant d^k_{KNN}$. Zatem może się zdarzyć, że wybierzemy więcej niż $k$ sąsiadów.

W najprostszej wersji KNN:

$$d_{KNN}(x, x_*) = \sum_{i=1}^{N} neq(a_i(x), a_i(x_*))$$

$$neq(a, b) = \begin{cases} 0 & \text{jeżeli: } a = b \\ 1 & \text{jeżeli: } a \neq b \end{cases}$$

Po wybraniu sąsiadów $x_*$, sprawdzamy do jakich należą klas decyzyjnych. Badany obiekt $x_*$ jest przypisywany do „najpopularniejszej" klasy decyzyjnej.

W metodzie tej nie korzystamy z reguł decyzyjnych, lecz z przykładów. Ma to dwie ogromne zalety: nie ma potrzeby czasochłonnego wyliczania reguł oraz bez trudu możemy dodawać do naszego zbioru przykładów nowe przypadki.

Bardzo ważnym elementem tej metody jest odpowiedni dobór funkcji $d_{KNN}$ do analizowanych danych, co może być niezwykle trudne i czasochłonne.

Metoda ta, uzyskuje często zaskakująco dużą skuteczność. Niestety wymaga wykonania wielu obliczeń, co ma duży wpływ na czas potrzebny do klasyfikacji nowego obiektu.

### 2.2.2 System klasyczny oparty na regułach decyzyjnych

System klasyczny korzysta ze zbioru wszystkich (niesprzecznych) reguł decyzyjnych, wygenerowanych w oparciu o wszystkie dostępne przykłady $(x_1, \ldots, x_{k+m})$. Algorytm wyliczenia wszystkich reguł jest dosyć czasochłonny. Jest jednak wykonywany tylko jeden

raz. Sama klasyfikacja nowego obiektu jest już znacznie prostsza. Odbywa się ona poprzez głosowanie wszystkich reguł. Każda reguła może oddać głos na klasę decyzyjną , którą reprezentuje (jeśli rozpozna $x_*$) lub nie oddać żadnego głosu (jeśli nie rozpozna $x_*$). Obiekt $x_*$ jest przydzielany do klasy, która zbierze najwięcej głosów.

Ponieważ klasy decyzyjne nie są reprezentowane przez taką samą liczbę reguł, oraz reguły te mają różną skuteczność; należy każdej regule przyporządkować pewną liczbę (zwaną wagą), która będzie brała udział w głosowaniu:

$$Waga(r_i) = \frac{Q_W(r_i)}{\sum_{\{r:d^r=d^{r_i}\}} Q_W(r)}$$

Gdzie $d^r$ to klasa decyzyjna reprezentowana przez $r$, a $Q_W$ jest funkcją oceniającą reguły. Na przykład możemy przyjąć:

$$Q_W(r) = \text{liczba prawidłowo rozpoznawanych obiektów ze zbioru treningowego}$$

Zauważmy, że suma wag w każdej klasie decyzyjnej wynosi 1. Każda klasa ma więc równe szanse w głosowaniu. Jednocześnie przyjęte wagi faworyzują reguły, które zostały wysoko ocenione (przez $Q_W$).

System ten bywa bardzo skuteczny. Jeżeli dla naszych danych mamy dużo reguł decyzyjnych, wówczas potrzebujemy dużo czasu na klasyfikację nowego obiektu.

# 3 Redukcja liczby reguł decyzyjnych

## 3.1 Motywacja

W punkcie 2.2.2 przedstawiłem budowę systemu opartego na wszystkich regułach decyzyjnych. Niestety często tych reguł jest bardzo dużo, co wydłuża czas klasyfikacji nowego obiektu. Istnieją zastosowania, w których trzeba błyskawicznie podejmować decyzję. Na przykład wyobraźmy sobie robota, który przez pół godziny „zastanawia" się jak wykonać kolejny krok. W takich zastosowaniach system klasyczny okazuje się zbyt powolny. Redukując liczbę reguł biorących udział w głosowaniu, redukujemy również czas konieczny na klasyfikację.

Innym ważnym powodem, dla którego chcemy mieć możliwie najmniej reguł decyzyjnych, jest czytelność zbioru reguł, oraz chęć uzyskania możliwie prostego opisu tablicy decyzyjnej. Jeśli nasza tablica decyzyjna opisuje wyniki jakiegoś eksperymentu naukowego, wówczas reguły decyzyjne mogą posłużyć do opisu tego doświadczenia, co może dostarczyć wielu cennych informacji.

## 3.2 Podstawowe trudności

Redukcja liczby reguł decyzyjnych może nastąpić poprzez eliminację części z nich lub poprzez ich łączenie.

Eliminacja części reguł pociąga za sobą ryzyko nie rozpoznania wielu nowych obiektów. Innym problemem jest radykalna zmiana „układu sił" w głosowaniu. System klasyczny w dużej mierze zawdzięcza swą skuteczność wielości reguł decyzyjnych.

Łączenie reguł decyzyjnych może spowodować powstanie reguł, które bardzo często będą błędnie klasyfikowały nowe obiekty. Załóżmy, że posiadamy $n$ reguł postaci: $\beta_1 \Rightarrow d^{r_1}, \ldots, \beta_n \Rightarrow d^{r_n}$ $(d_i \in \mathbb{D})$. Łącząc dwie reguły $r_k : \beta_k \Rightarrow d$ i $r_l : \beta_l \Rightarrow d$, otrzymamy jedną: $r_{k,l} : \beta_k \vee \beta_l \Rightarrow d$. Okazuje się, że nowy obiekt $x_*$ jest rozpoznawany tylko przez

jedną z nich (tzn. spełniony jest warunek: $\beta_k \wedge \neg\beta_l$). Zatem w klasycznym głosowaniu głos oddałaby tylko jedna reguła mająca mniejszą wagę niż $r_{k,l}$. Po połączeniu reguła bierze udział w głosowaniu, gdy spełniony jest choć jeden z warunków $\beta_k$, $\beta_l$. Jeśli reguła $r_k : \beta_k \Rightarrow d$ popełniła błąd, wówczas nie ma to dużego znaczenia w systemie klasycznym, gdyż jest to jedna z wielu reguł. Natomiast dla nowego systemu opartego o połączone reguły, stanowi to poważne zagrożenie, gdyż system ten zawiera mniej reguł, a ponadto reguła $\beta_k \vee \beta_l \Rightarrow d$ ma większą wagę w głosowaniu niż każda z reguł składowych ($r_k$ i $r_l$), co zwiększa prawdopodobieństwo błędnej klasyfikacji.

# 4 Przykłady znanych metod redukcji liczby reguł

## 4.1 Redukcja poprzez ocenę jakości reguł - systemy QbF

Zdecydowana większość stosowanych obecnie metod, polega na wyborze pewnego podzbioru ze zbioru wszystkich reguł decyzyjnych. W tym celu należy ocenić przydatność posiadanych reguł decyzyjnych, a następnie wybrać najlepsze. Proces ten nosi nazwę Quality-based Filtering – QbF (patrz [1], [4]).

Załóżmy, że posiadamy zbiór przykładów $\mathbb{X} = \{x_1, \ldots, x_{k+m}\}$ wraz z przypisanymi im decyzjami (na przykład przez eksperta), w oparciu o pierwsze $k$ przykładów $(x_1, \ldots, x_k)$ wyznaczono reguły decyzyjne: $r_1, \ldots, r_n$. Otrzymane reguły oceniamy poprzez analizę ich zachowania na wszystkich przykładach ze zbioru $\mathbb{X}$. W tym celu przyjmijmy oznaczenia[2]:

$$N_R(r) = |\{x_i \in \mathbb{X} : r(x_i) = d^r\}|$$

$$N_d(r) = |\{x_i \in \mathbb{X} : d_i = d^r\}|$$

$$N_{RP}(r) = |\{x_i \in \mathbb{X} : r(x_i) = d^r \wedge d_i = d^r\}|$$

$$N_{RB}(r) = |\{x_i \in \mathbb{X} : r(x_i) = d^r \wedge d_i \neq d^r\}|$$

$$N_{NB}(r) = |\{x_i \in \mathbb{X} : r(x_i) = d^? \wedge d_i = d^r\}|$$

$$N_{NP}(r) = |\{x_i \in \mathbb{X} : r(x_i) = d^? \wedge d_i \neq d^r\}|$$

$$accuracy(r) = \frac{N_{RP}(r)}{N_R(r)}$$

$$coverage(r) = \frac{N_{RP}(r)}{N_d(r)}$$

Zauważmy następujące zależności:

$$N_R(r) = N_{RP}(r) + N_{RB}(r)$$

$$N_d(r) = N_{RP}(r) + N_{NB}(r)$$

$$N_{RP}(r) + N_{RB}(r) + N_{NB}(r) + N_{NP}(r) = k$$

Przedstawione oznaczenia są niezbędne do zdefiniowania przykładowych funkcji opisujących jakość reguł decyzyjnych: (patrz [4])

[2] $|A|$ oznacza moc zbioru $A$

$$
\begin{aligned}
\text{Michalski}: &\quad Q_1(\mu, r) = \mu * accuracy(r) + (1-\mu) * coverage(r) \\
\text{Torgo}: &\quad Q_2(r) = Q_1(\tfrac{1}{2} + \tfrac{1}{4} accuracy(r), r) \\
\text{Brazdil}: &\quad Q_3(r) = accuracy(r) * e^{coverage(r)-1} \\
\text{Pearson}: &\quad Q_4(r) = \tfrac{N_{RP}(r)N_{NP}(r) - N_{RB}(r)N_{NB}(r)}{N_R(r)N_d(r)(k-N_R(r))(k-N_d(r))} \\
\text{G2}: &\quad Q_5(r) = 2\left(N_{RP}(r)\ln\left(\tfrac{kN_{RP}(r)}{N_R(r)N_d(r)}\right) + N_{RB}(r)\ln\left(\tfrac{kN_{RB}(r)}{N_R(r)(k-N_d(r))}\right)\right) \\
\text{J}: &\quad Q_6(r) = Q_5(r)/2k \\
\text{Cohen}: &\quad Q_7(r) = \tfrac{k(N_{RP}(r)+N_{NP}(r)) - N_R(r)N_d(r)}{k^2 - N_R(r)N_d(r) - (k-N_R(r))(k-N_d(r))} = \tfrac{k(N_{RP}(r)+N_{NP}(r)) - N_R(r)N_d(r)}{k(N_R(r)+N_d(r)) - 2N_R(r)N_d(r)} \\
\text{Coleman}: &\quad Q_8(r) = \tfrac{kN_{RP}(r) - N_R(r)N_d(r)}{N_R(r)(k-N_d(r))} \\
\text{C1}: &\quad Q_9(r) = \tfrac{1}{3} Q_8(r)\,(2 + Q_7(r)) \\
\text{C2}: &\quad Q_{10}(r) = \tfrac{1}{2} Q_8(r)\,(1 + coverage(r)) \\
\text{Kononenko}: &\quad Q_{11}(r) = -\log_2 \tfrac{N_d(r)}{k} + \log_2 accuracy(r) \\
\text{Wsparcie}: &\quad Q_{12}(r) = N_{RP}(r) \\
1: &\quad Q_{13}(r) = 1
\end{aligned}
$$

Wybór właściwej funkcji zależy od naszych potrzeb i rodzaju analizowanych danych. Przy jej pomocy oceniamy wszystkie reguły decyzyjne ze zbioru $\{r_1, \ldots, r_n\}$, a następnie wybieramy tylko te najlepsze. Możemy na przykład założyć, że interesuje nas tylko 20 najlepszych reguł i dokładnie tyle wybrać. Można też wybierać reguły tak, aby pokryć cały zbiór $\mathbb{X}$; dostając w ten sposób, zależnie od danych, podzbiór reguł o mocy większej lub mniejszej od przykładowych 20 reguł.

Może się zdarzyć, że wybrane reguły nie pokryją całego zbioru przykładów $\mathbb{X}$ (dotyczy to przypadku, w którym wybieramy ustaloną liczbę reguł), to znaczy nie będą w stanie sklasyfikować wszystkich $x_i$. Można jednak tego uniknąć poprzez powiększenie wybranego podzbioru o dodatkowe reguły.

Klasyfikacja nowego obiektu odbywa się poprzez głosowanie wybranych reguł. Jako wagi można przyjąć wartości: $Q_*(r_i)$ unormowane tak, aby każda klasa decyzyjna miała równe szanse w głosowaniu.

System QbF można rozbudować poprzez dodanie możliwości wyboru funkcji oceniającej, jak również poprzez skracanie reguł początkowych, co przedstawiłem w punktach 4.2 i 4.3.

## 4.2 System QbF z doborem funkcji oceniającej

Przez wiele lat próbowano zbudować najskuteczniejszą funkcję oceniającą, która dawałaby najlepsze wyniki na każdej możliwej tablicy decyzyjnej. W końcu zdano sobie sprawę, że funkcję oceniającą jakość reguł należy dobierać w zależności od charakteru analizowanych danych.

System QbF z doborem funkcji oceniającej polega na przetestowaniu skuteczności systemów QbF opartych na jednej funkcji i wyborze najlepszej z nich.

## 4.3 Skracanie reguł

Zbiór wszystkich reguł decyzyjnych zawiera często reguły o bardzo rozbudowanej koniunkcji (mających dużo atrybutów znaczących), które rozpoznają bardzo niewiele obiektów. Pozbycie się zbyt rozbudowanych reguł może spowodować, że stracimy zbyt wiele istotnych informacji. Można jednak operować na skróconych regułach [2], czyli takich które posiadają krótszy opis (na przykład zamiast: $a_1 = 1 \wedge a_3 = 2 \wedge a_6 = 7 \Rightarrow d$ można użyć reguły: $a_1 = 1 \wedge a_3 = 2 \Rightarrow d$) kosztem dopuszczenia błędów (np: 30%). Należy się

spodziewać, że krótsze reguły będzie można łatwiej łączyć tworząc w ten sposób bardziej wydajny system reprezentantów (zobacz rozdział 5).

# 5 Propozycja: system reprezentantów

Zaproponowany przeze mnie *system reprezentantów* ma na celu zmniejszenie liczby reguł decyzyjnych poprzez ich łączenie. W ten sposób ograniczamy liczbę reguł decyzyjnych, które muszą być sprawdzone przy klasyfikacji każdego nowego przypadku. Łączenie reguł powoduje jednocześnie ich uogólnianie, co może mieć praktyczne zastosowanie w analizie danych. Wadą tego systemu jest konieczność przeprowadzenia wielu skomplikowanych obliczeń. Jest to jednak operacja jednorazowa i uzyskany w ten sposób system decyzyjny może być używany do klasyfikacji wielu obiektów w bardzo krótkim czasie.

## 5.1 Idea systemu reprezentantów

Poniżej przedstawiam konstrukcję systemu tworzącego zbiór nowych reguł zwanych dalej *reprezentantami*.

Schemat budowy systemu reprezentantów:

1. Podział zbioru $\mathbb{X}$ na dwa podzbiory: $\{x_1, \ldots, x_k\}$, oraz $\{x_{k+1}, \ldots, x_{k+m}\}$
   Podział zbioru dostępnych przykładów $\mathbb{X}$ ma na celu wyodrębnienie zbioru obiektów, z których będą wyliczane reguły decyzyjne i zbioru służącego do lepszej oceny zachowania się reguł na nowych danych.

2. Wyliczenie wszystkich niesprzecznych reguł ze zbioru $\{x_1, \ldots, x_k\}$
   Wyliczane reguły mają taką samą postać jak w systemie klasycznym.

3. Skracanie reguł poprzez eliminację elementów koniunkcji
   Skracanie reguł ma na celu eliminację zbyt szczegółowych i tym samym trudnych do uogólnienia reguł.

4. Podział reguł na klasy decyzyjne
   Następne punkty należy wykonać dla każdej klasy decyzyjnej osobno.

5. Podział każdej klasy decyzyjnej na grupy reguł, tak aby w każdej grupie znalazły się reguły o podobnej skuteczności na danych $x_1, \ldots, x_{k+m}$
   Grupowanie reguł ma zapobiegać późniejszemu łączeniu mocnych reguł ze słabymi.

6. Podział każdej grupy na podgrupy reguł, tak aby w każdej podgrupie znalazły się reguły o podobnej strukturze logicznej
   Tworzenie podgrup ułatwia łączenie reguł i zapobiega powstawaniu zbyt dużych błędów w późniejszej klasyfikacji.

7. Budowa reprezentantów poprzez łączenie reguł w obrębie podgrup
   Z każdej podgrupy tworzony jest jeden reprezentant (nowa reguła) poprzez połączenie występujących w niej reguł.

8. Ocena powstałych reprezentantów i wyznaczenie wag do głosowania
   Należy ocenić zbiór nowo powstałych reguł, aby można było dobrać wagi do głosowania.

9. Klasyfikacja nowego obiektu odbywa się poprzez głosowanie wszystkich reprezentantów z uwzględnieniem przydzielonych im wag.

## 5.2 Problem wyboru reguł do łączenia

Pierwszym pomysłem jaki się nasuwa jest połączenie wszystkich reguł w ramach danej klasy decyzyjnej. Jeśli nasz zbiór reguł przedstawimy następująco: $\beta_1 \Rightarrow d, \ldots, \beta_n \Rightarrow d$, wówczas nasza „super-reguła" miałaby postać: $\beta_1 \vee \ldots \vee \beta_n \Rightarrow d$. Postępując w ten sposób z każdą klasą decyzyjną otrzymalibyśmy zaledwie parę reguł decyzyjnych (dokładnie tyle ile jest możliwych decyzji). Aby każda klasa decyzyjna miała równe szanse w głosowaniu, nasze „super-reguły" musiałaby otrzymać taką samą wagę, np. 1. Pozornie wszystko układa się doskonale – zbudowaliśmy system decyzyjny oparty na niewielkiej liczbie reguł. Teraz chcielibyśmy dokonać klasyfikacji nowego przypadku $x_*$. Ponieważ w systemie klasycznym niezwykle rzadko zdarza się aby wszystkie reguły opowiadały się za jedną klasą decyzyjną (tzn. $\exists!_{d\in\mathbb{D}}\forall_i\ r_i(x_*) \in \{d, d^?\}$ oraz $\exists_i r_i(x_*) \neq d^?$) to należy się spodziewać, że zazwyczaj kilka „super-reguł" rozpozna $x_*$ jako element swojej klasy. Tym samym nie będziemy potrafili na drodze takiego głosowania przydzielić nowego obiektu do żadnej z klas decyzyjnych. Opisany problem może się zdarzyć również w systemie klasycznym, jednak duża liczba reguł biorąca udział w głosowaniu powoduje, że jest to niezwykle mało prawdopodobne.

Grupowanie i łączenie reguł decyzyjnych może spowodować, że otrzymane reguły będą bardzo słabe (będą popełniały wiele błędów[3]) i w efekcie będą całkowicie nieprzydatne. Należy pamiętać, że połączenie dwóch silnych reguł może dać słabą regułę i na odwrót – połączenie dwóch słabych reguł może dać silną regułę. Jak to możliwe ? W pierwszym przypadku (dwie silne reguły) może okazać się, że każda z reguł popełnia niewiele błędów[4], jednak nie pokrywają się one wskutek czego po połączeniu mamy regułę, która popełnia dwukrotnie więcej błędów. W drugim przypadku mamy dwie reguły popełniające wiele błędów[5] ale po połączeniu reguły te doskonale się uzupełniają tworząc jedną silną regułę. Zatem najrozsądniej byłoby łączyć te reguły, które popełniają podobne błędy lub uzupełniają się nawzajem.

W proponowanym systemie decyzyjnym podział na grupy ma za zadanie odseparować mocne reguły od słabych (popełniających wiele błędów). Dalszy podział reguł na podgrupy ma zapewnić możliwie duże podobieństwo struktur logicznych łączonych reguł. W ten sposób mam nadzieję ustrzec się większości opisanych problemów.

## 5.3 Grupowanie reguł

W punkcie 5.3.2 przedstawiam algorytm podziału reguł na grupy. W algorytmie tym wyznaczam skuteczność każdej reguły, a następnie dzielę otrzymane wartości na podzbiory, co jednoznacznie zadaje podział reguł na grupy. W ten sposób omijam konieczność liczenia odległości pomiędzy każdymi dwoma regułami.

Aby jednak precyzyjniej wyznaczyć grupy reguł, można rozbudować przedstawiony algorytm do pełnego algorytmu centroidów. Niezbędne będzie wówczas zdefiniowanie funkcji odległości pomiędzy dwoma regułami uwzględniającej zachowanie się reguł na nowych

[3]są dwa rodzaje błędów: I – błędne zakwalifikowanie przykładu do danej klasy decyzyjnej i II – nie zakwalifikowanie przykładu pomimo jego przynależności do rozpatrywanej klasy decyzyjnej

[4]głównie błędów typu I

[5]głównie błędów typu II

danych. Przykładową konstrukcję takiej funkcji i jej własności przedstawiam w punkcie 5.3.1.

### 5.3.1 Przykład funkcji odległości

**Definicja 1** *Odległością pomiędzy dwoma decyzjami* $x, y \in \mathbb{D} \cup \{d^?\}$ *nazwiemy funkcję:*

$$\varrho_q(x,y) = \begin{cases} 0 & \text{jeżeli } x = y \\ q & \text{jeżeli } x \neq y \wedge (x = d^? \vee y = d^?) \\ 1 & \text{w pozostałych przypadkach} \end{cases}$$

*gdzie* $q \in \left\langle \frac{1}{2}, 1 \right\rangle$.

**Uwaga 1** *Funkcja* $\varrho_q(x,y)$ *posiada następujące własności:*

*(1)* $\varrho_q(x,y) \geqslant 0$

*(2)* $\varrho_q(x,y) = \varrho_q(y,x)$

*(3)* $\varrho_q(x,y) = 0 \iff x = y$

*(4)* $\varrho_q(x,y) + \varrho_q(y,z) \geqslant \varrho_q(x,z)$.

Dowód

Punkty (1), (2) i (3) wynikają wprost z definicji funkcji odległości. Poniżej przedstawiam prosty dowód własności (4) (przez wyczerpywanie).

- Jeśli $\varrho_q(x,z) = 0$, to warunek (4) zachodzi na mocy własności (1).
- Jeśli $\varrho_q(x,z) = q$, to warunek (4) również zachodzi; ponieważ gdyby $\varrho_q(x,y) = 0$ i $\varrho_q(y,z) = 0$, to $x = y$ i $y = z$ (z własności (3)), zatem $x = z$ co oznacza, że $\varrho_q(x,z) = 0$ (z własności (3)) a to jest sprzeczne z założeniem.
- Jeśli $\varrho_q(x,z) = 1$ i $q \neq 1$, to $x \neq d^?$ i $z \neq d^?$ (z definicji $\varrho_q$). Jeśli $y = d^?$ to własność (4) jest spełniona, ponieważ $q + q \geqslant 1$ ($\varrho_q(x,d^?) = \varrho_q(d^?,z) = q$). Jeśli $y \neq d^?$ wówczas własność (4) również jest spełniona, gdyż $\varrho_q(x,y), \varrho_q(y,z) \in \{0,1\}$, gdyby więc $\varrho_q(x,y) = 0$ i $\varrho_q(y,z) = 0$, to $x = y$ i $y = z$ (z własności (3)), zatem $x = z$ co oznacza, że $\varrho_q(x,z) = 0$ (z własności (3)) a to jest sprzeczne z założeniem.
- Jeśli $\varrho_q(x,z) = 1$ i $q = 1$, to własność (4) jest spełniona, ponieważ gdyby $\varrho_q(x,y) = 0$ i $\varrho_q(y,z) = 0$, to $x = y$ i $y = z$ (z własności (3)), zatem $x = z$ co oznacza, że $\varrho_q(x,z) = 0$ (z własności (3)) a to jest sprzeczne z założeniem.

■

Teraz możemy już zdefiniować odległość pomiędzy dwoma regułami.

**Definicja 2** *Przyjmijmy następujące oznaczenia:*

- $\overline{r_i} = (r_i(x_{k+1}), \ldots, r_i(x_{k+m}))$
- $\overline{d} = (d_{k+1}, \ldots, d_{k+m})$
- $\overline{\varrho_q}(\overline{r_{i_1}}, \overline{r_{i_2}}) = \sum_{j=1}^{m} \varrho_q(r_{i_1}(x_{k+j}), r_{i_2}(x_{k+j}))$
- $\overline{\varrho_q}(\overline{r_i}, \overline{d}) = \sum_{j=1}^{m} \varrho_q(r_i(x_{k+j}), d_{k+j})$

Z uwagi 1 natychmiast wynika następujący wniosek:

**Wniosek 1** *Prawdziwe są następujące nierówności:*

*(1)* $\overline{\varrho_q}(\overline{r_1}, \overline{d}) + \overline{\varrho_q}(\overline{r_2}, \overline{d}) \geqslant \overline{\varrho_q}(\overline{r_1}, \overline{r_2})$

*(2)* $\overline{\varrho_q}(\overline{r_1}, \overline{r_2}) \geqslant \left|\overline{\varrho_q}(\overline{r_1}, \overline{d}) - \overline{\varrho_q}(\overline{r_2}, \overline{d})\right|$

Dowód
Dowód sprowadza się do kilkukrotnego zastosowania nierówności trójkąta.

■

Powyższa funkcja może posłużyć do oceny jakości reguł:

$$Q_0(r) = \overline{\varrho_q}(\overline{r}, \overline{d})$$

Funkcja $Q_0$ może być z powodzeniem wykorzystywana w systemach QbF.

### 5.3.2 Algorytm podziału na grupy

Zbiór wygenerowanych reguł dzielimy na klasy decyzyjne, a następnie na $K$ grup[6] przy pomocy jednowymiarowego algorytmu centroidów. Podział reguł na grupy ma za zadanie oddzielić reguły popełniające dużo błędów od tych, które popełniają ich niewiele. Do wyznaczenia ilości popełnianych błędów użyjemy funkcji $Q_R$. Ponieważ ilość popełnianych błędów nie jest jedynym wyznacznikiem jakości reguł, użyjemy dodatkowej funkcji oceniającej $Q_W$, dzięki której będziemy mogli dokonać dokładniejszego podziału na grupy. Funkcje $Q_R$ i $Q_W$ są parametrami algorytmu podziału na grupy. Przykładowo można przyjąć $Q_R(r) = Q_0(r)$ (funkcja $Q_0$ została zdefiniowana w punkcie 5.3.1), oraz $Q_W(r) = Q_{12}(r)$ (funkcja $Q_{12}$ została zdefiniowana w punkcie 4.1). Wybór takich samych[7] funkcji $Q_R$ i $Q_W$, prowadził w przeprowadzonych eksperymentach do zwiększenia liczby błędów.

Algorytm podziału na grupy:

1. Każdej regule $r_i$ przyporządkowujemy: $o_i = Q_R(r_i)$ i $c_i = Q_W(r_i)$, dobieramy także wartości $v_0$ i $v_K$ w taki sposób aby $\forall_i\, v_0 < o_i < v_K$
   Każda reguła jest reprezentowana w przedziale $(v_0, v_K)$ przez odpowiadającą jej wartość $o_i$.

2. Wybieramy losowo punkty $w_0, \ldots, w_{K-1}$, $v_0 < w_i < v_K$

3. Obliczamy $v_1, \ldots, v_{K-1}$ ze wzoru $v_i = \frac{w_{i-1}+w_i}{2}$

4. Obliczamy środki ciężkości powstałych odcinków:

$$w_i^* = \frac{\sum_{\{j:o_j \in \langle v_i, v_{i+1})\}} c_j o_j}{\sum_{\{j:o_j \in \langle v_i, v_{i+1})\}} c_j}$$

   Dla $i = 0, \ldots, K-1$ $w_i^*$ jest nowym środkiem ciężkości odcinka $\langle v_i, v_{i+1})$.

5. Obliczamy błąd dopasowania odcinków:

$$\mathsf{ERR}(w_0^*, \ldots, w_{K-1}^*) = \sum_{i=1}^{K} \sum_{\{j:o_j \in \langle v_i, v_{i+1})\}} c_j (o_j - w_i^*)^2$$

[6] na grupy dzielimy zbiory reguł reprezentujących tę samą klasę decyzyjną
[7] to znaczy: $\forall_r\, Q_R(r) = Q_W(r)$

6. Jeśli ERR$(w_0^*, \ldots, w_{K-1}^*) <$ ERR$(w_0, \ldots, w_{K-1})$, to: $w_i := w_i^*$ i wracamy do punktu 3.

Otrzymane punkty $v_0 < v_1 < v_2 < \ldots < v_K$ zadają podział na $K$ przedziałów. Reguły, których oceny $o_i$ znalazły się w jednym przedziale, tworzą grupę reguł. Należy jeszcze odpowiednio dobrać parametr K. Pomoże nam w tym następujący algorytm:

Algorytm doboru liczby grup:

1. Niech $K$ będzie liczbą grup, na początku: $K :=$ liczba reguł do podziału na grupy

2. Obieramy $c_i$: $c_i = Q_W(r_i)$

3. Wybieramy początkową wartość $\varepsilon$, na przykład: $\varepsilon = \frac{1}{3} max_{i,j}\{|Q_R(r_i) - Q_R(r_j)|\}$
   $\varepsilon$ określa maksymalną odległość pomiędzy regułami, które będziemy uznawali za „sąsiadujące" ze sobą. Współczynnik $\frac{1}{3}$ został dobrany eksperymentalnie.

4. Dla reguł $r_1, \ldots, r_K$ obliczamy $C_i$:

$$C_i := \sum_{\{j:|Q_R(r_i)-Q_R(r_j)|<\varepsilon\}} \frac{c_j}{d_C(r_i, r_j)}$$

   gdzie

$$d_C(r_i, r_j) = \begin{cases} |Q_R(r_i) - Q_R(r_j)| & \text{jeżeli } Q_R(r_i) \neq Q_R(r_j) \\ 1 & \text{jeżeli } Q_R(r_i) = Q_R(r_j) \end{cases}$$

   Wartość $C_i$ jest tym większa im więcej reguł $r_j$ z wysokimi ocenami $c_j$ znajduje się blisko (w sensie odległości $d_C$) reguły $r_i$.

5. Przepisujemy wartości $C_i$ na $c_i$: $c_i := C_i$

6. Dobieramy $\delta$, na przykład: $\delta = \frac{\sum_{i=1}^{K} c_i}{1.2K}$
   $\delta$ określa minimalną ocenę $c_i$, dla której uznamy, że reguła $r_i$ ma wystarczająco dużo silnych sąsiadów. Współczynnik 1.2 został dobrany eksperymentalnie.

7. Reguły $(r_1, \ldots, r_K)$ są sortowane tak aby na początku znalazły się te, które nie spełniają warunku $c_i < \delta$, nowe $K := K - |\{r_i \in \{r_1, \ldots, r_K\} : c_i < \delta\}|$
   Reguły, dla których $c_i < \delta$ zostają odrzucone, a $K :=$ liczba reguł, które pozostały.

8. Jeśli $K$ zostało zmienione w punkcie 7, oraz $K > 1$, wówczas zmniejszamy $\varepsilon$ (na przykład: $\varepsilon := 0.8 \cdot \varepsilon$) i powracamy do punktu 4.
   Współczynnik 0.8 został dobrany eksperymentalnie. Algorytm wykonywany jest tak długo, aż w punkcie 7-mym nie zostanie odrzucona żadna reguła lub $K$ przyjmie wartość $\leqslant 1$ – wówczas $K := 1$. Ponieważ $K$ przyjmuje tylko wartości całkowite i w punktach 4–7 jego wartość nie zwiększa się, mamy pewność, że algorytm zakończy się w skończonej liczbie kroków.

Otrzymane $K$ określa liczbę grup. Poza doborem liczby grup uzyskujemy początkowe wartości środków ciężkości (zamiast losowych w 2-gim punkcie algorytmu podziału na grupy):

$$w_i = Q_R(r_{i+1}) \text{ dla: } i = 0, \ldots, K-1$$

## 5.4 Tworzenie podgrup reguł

Podział grup reguł na podgrupy, ma za zadanie odseparować od siebie reguły o różnych strukturach logicznych, których późniejsze łączenie byłoby trudne, a powstałe w ten sposób reguły mało wartościowe.

### 5.4.1 Struktura logiczna reguł

Nasze reguły mają postać koniunkcji:

$$(a_{1_r} = w^r_{1_r} \wedge \ldots \wedge a_{N_r} = w^r_{N_r}) \Rightarrow \text{KlasaDecyzyjna}.$$

Rozpatrzymy na prostym przykładzie problem porównywania struktur logicznych. Został już wykonany podział na grupy reguł, a naszym zadaniem jest przetworzenie jednej z otrzymanych grup. Zakładamy, że tablica z danymi zawiera $N = 9$ atrybutów opisujących każdy przykład, oraz że mamy grupę pięciu reguł decyzyjnych:

$r_1$: $(a_1 = 1 \wedge a_2 = 3 \wedge a_5 = 1 \wedge a_8 = 2) \Rightarrow$ Decyzja1
$r_2$: $(a_1 = 1 \wedge a_2 = 3 \wedge a_4 = 1 \wedge a_7 = 2 \wedge a_9 = 1) \Rightarrow$ Decyzja1
$r_3$: $(a_1 = 2 \wedge a_2 = 3 \wedge a_5 = 1 \wedge a_8 = 2) \Rightarrow$ Decyzja1
$r_4$: $(a_2 = 2 \wedge a_4 = 1) \Rightarrow$ Decyzja1
$r_5$: $(a_3 = 2 \wedge a_4 = 1 \wedge a_9 = 2) \Rightarrow$ Decyzja1

Chcielibyśmy wyłonić podgrupy składające się z takich reguł, które łatwo można by łączyć. W tym celu musimy porównać ich struktury logiczne. Aby ułatwić to zadanie, przedstawmy reguły w postaci ciągów $\left(r^A(i)\right)_{i=1}^N$, zawierających kolejne wartości atrybutów wymagane przez regułę lub znak „$\star$" w przypadku, gdy dany atrybut nie ma znaczenia dla rozpatrywanej reguły decyzyjnej:

$$r^A(i) = \begin{cases} w^r_i & \text{jeżeli } a_i \text{ jest atrybutem znaczącym} \\ \star & \text{jeżeli reguła nie zawiera: } a_i = w^r_i \end{cases}$$

W naszym przykładzie reguły przyjmą następującą postać:

$r_1$: 1 3 $\star$ $\star$ 1 $\star$ $\star$ 2 $\star$ $\Rightarrow$ Decyzja1
$r_2$: 1 3 $\star$ 1 $\star$ $\star$ 2 $\star$ 1 $\Rightarrow$ Decyzja1
$r_3$: 2 3 $\star$ $\star$ 1 $\star$ $\star$ 2 $\star$ $\Rightarrow$ Decyzja1
$r_4$: $\star$ 2 $\star$ 1 $\star$ $\star$ $\star$ $\star$ $\star$ $\Rightarrow$ Decyzja1
$r_5$: $\star$ $\star$ 2 1 $\star$ $\star$ $\star$ $\star$ 2 $\Rightarrow$ Decyzja1

### 5.4.2 Przykład odległości struktur logicznych

**Definicja 3** *Odległością pomiędzy dwoma regułami, uwzględniającą ich strukturę logiczną nazwiemy funkcję:*

$$d_L(r_1, r_2) = |s_\star(r_1) - s_\star(r_2)| + \sum_{i=1}^{N} d_{L0}(r_1^A(i), r_2^A(i))$$

*gdzie:*

$$s_\star(r) = |\{i : r^A(i) = \star\}|$$

$$d_{L0}(a, b) = \begin{cases} 0 & \textit{jeżeli: } a = b \\ 1 & \textit{jeżeli: } a \neq b \wedge (a \neq \star \wedge b \neq \star) \\ 2 & \textit{jeżeli: } a \neq b \wedge (a = \star \vee b = \star) \end{cases}$$

Zgodnie z podaną definicją bliskie są reguły, które posiadają taką samą liczbę atrybutów znaczących, najlepiej gdyby były to te same atrybuty. Różnice wartości na atrybutach znaczących mają mniejsze znaczenie, gdyż mają mniejszy wpływ na łatwość łączenia reguł, niż wcześniej wspomniane cechy.

Przedstawiony powyżej przykład jest jedną z wielu możliwych funkcji, które mogą być użyte podczas grupowania i łączenia reguł. Funkcję odległości pomiędzy regułami, uwzględniającą ich strukturę logiczną, można dobierać indywidualnie do analizowanych danych.

### 5.4.3 Podział na podgrupy

Do podziału grup reguł na $K$ podgrup należy użyć standardowego algorytmu centroidów. Parametrami poniższego algorytmu są funkcje $Q_W$ i $d$. Funkcja $Q_W$ ocenia jakość reguł decyzyjnych, a funkcja $d$ wyznacza odległość pomiędzy dwoma regułami z uwzględnieniem ich struktury logicznej.

Algorytm podziału na podgrupy:

1. Każdej regule $r_i$ przyporządkowujemy wagę $Q_W(r_i)$.
2. Wybieramy losowo punkty $w_0, \ldots, w_{K-1} \in \{\mathbb{N} \cup \{\star\}\}^N$, reprezentujące reguły z dzielonej grupy.
3. Każdą regułę przydzielamy do najbliższego środka ciężkości $w_i$, używając do tego celu funkcji odległości $d$. W ten sposób zadajemy podział na $K$ zbiorów
4. Wybranie nowych środków ciężkości $w_i^*$ spośród reguł przypisanych w poprzednim punkcie do $w_i$, tak aby zminimalizować $\sum_{r \in Otoczenie(w_i^*)} Q_W(r) \cdot d(r, w_i^*)$,
   gdzie $Otoczenie(w_i^*) = \{$reguły przydzielone w punkcie 3 do $w_i\}$
5. Obliczamy błąd dopasowania:
$$\mathrm{ERR}(w_0^*, \ldots, w_{n-1}^*) = \sum_i \sum_{r \in Otoczenie(w_i^*)} Q_W(r) \cdot d(r, w_i^*)$$
6. Jeśli $\mathrm{ERR}(w_0^*, \ldots, w_{K-1}^*) < \mathrm{ERR}(w_0, \ldots, w_{K-1})$, to zamieniamy wagi: $w_i := w_i^*$ i wracamy do punktu 3.

Przyporządkowanie każdej reguły do najbliższego środka ciężkości $w_i$ zadaje podział na $K$ podgrup. Przykładowe funkcje będące parametrami tego algorytmu to:

$$Q_W(r) = Q_{12}(r)$$

$$d(r_1, r_2) = dL(r_1, r_2) + \overline{\varrho_q}(\overline{r_1}, \overline{r_2}) + 1$$

Podgrupy są budowane za pomocą algorytmu centroidów. Istotnym elementem tego algorytmu jest dobór liczby podgrup, oraz początkowych środków ciężkości. Poniżej przedstawiam algorytm rozwiązujący oba te problemy.

Algorytm doboru liczby podgrup:

1. Niech $K$ będzie liczbą podgrup, na początku: $K =$ liczba reguł do podziału na podgrupy

2. Obieramy $c_i$: $c_i = Q_W(r_i)$

3. Wybieramy początkową wartość $\varepsilon$, na przykład: $\varepsilon = \frac{1}{3} max_{i,j}\{d(r_i, r_j)\}$
   $\varepsilon$ określa maksymalną odległość pomiędzy regułami, które będziemy uznawali za „sąsiadujące" ze sobą. Współczynnik $\frac{1}{3}$ został dobrany eksperymentalnie.

4. Dla reguł $r_1, \ldots, r_K$ obliczamy $C_i$:

$$C_i := \sum_{\{j: d(r_i, r_j) < \varepsilon\}} \frac{c_j}{d_C(r_i, r_j)}$$

   gdzie

$$d_C(r_i, r_j) = \begin{cases} d(r_i, r_j) & \text{jeżeli } d(r_i, r_j) \neq 0 \\ 1 & \text{jeżeli } d(r_i, r_j) = 0 \end{cases}$$

   Wartość $C_i$ jest tym większa im więcej reguł $r_j$ z wysokimi ocenami $c_j$ znajduje się blisko (w sensie odległości $d_C$) reguły $r_i$.

5. Przepisujemy wartości $C_i$ na $c_i$: $c_i := C_i$

6. Dobieramy $\delta$, na przykład: $\delta = \frac{\sum_{i=1}^{K} c_i}{1.2K}$
   $\delta$ określa minimalną ocenę $c_i$, dla której uznamy, że reguła $r_i$ ma wystarczająco dużo silnych sąsiadów. Współczynnik 1.2 został dobrany eksperymentalnie.

7. Reguły $(r_1, \ldots, r_K)$ są sortowane tak aby na początku znalazły się te, które nie spełniają warunku $c_i < \delta$, nowe $K := K - |\{r_i \in \{r_1, \ldots, r_K\} : c_i < \delta\}|$
   Reguły, dla których $c_i < \delta$ zostają odrzucone, a $K :=$ liczba reguł, które pozostały.

8. Jeśli $K$ zostało zmienione w punkcie 7, oraz $K > 1$, wówczas zmniejszamy $\varepsilon$ (na przykład: $\varepsilon := 0.8 \cdot \varepsilon$) i powracamy do punktu 4.
   Współczynnik 0.8 został dobrany eksperymentalnie. Algorytm wykonywany jest tak długo, aż w punkcie 7-mym nie zostanie odrzucona żadna reguła lub $K$ przyjmie wartość $\leqslant 1$ – wówczas $K := 1$. Ponieważ $K$ przyjmuje tylko wartości całkowite i w punktach 4–7 jego wartość nie zwiększa się, mamy pewność, że algorytm zakończy się w skończonej liczbie kroków.

Otrzymane $K$ określa liczbę podgrup, zaś reguły: $r_1, \ldots, r_K$ są początkowymi środkami ciężkości dla algorytmu centroidów, które należy obrać zamiast losowych w punkcie 2-gim algorytmu podziału na podgrupy:

$$w_i = r_{i+1} \quad \text{dla: } i = 0, \ldots, K-1$$

## 5.5 Łączenie reguł

Z każdej podgrupy poprzez łączenie reguł otrzymamy dokładnie jednego *reprezentanta*. Budowa klasycznych reguł decyzyjnych dopuszcza maksymalnie jedną ustaloną wartość na każdym testowanym atrybucie:

$$r^A(i) = \begin{cases} w_i^r \in \mathbb{N} & \text{jeżeli } a_i \text{ jest atrybutem znaczącym} \\ \star & \text{jeżeli reguła nie zawiera: } a_i = w_i^r \end{cases}$$

Budowa reprezentantów dopuszcza wiele wartości dla każdego z atrybutów:

$$R^A(i) \subset \mathbb{N} \cup \{\star\}$$

Łączenie reguł odbywa się poprzez rozbudowę dopuszczalnego zbioru wartości dla każdego z atrybutów:

$$\forall_i \; R^A(i) = r_1^A(i) \cup r_2^A(i) \cup \ldots \cup r_n^A(i)$$

Gdzie $r_1, \ldots, r_n$ są regułami, tworzącymi podgrupę. Dodatkowo ustalana jest maksymalna wartość $s_\star(r)$ dla łączonych reguł. Jest ona zapamiętywana jako maksymalna dopuszczalna liczba wolnych ($r^A(i) = \star$) atrybutów:

$$W_{max}^R = \max_{i=1,\ldots,n} \{s_\star(r_i)\}$$

Podobnie jak klasyczne reguły, każdy reprezentant wyznacza funkcję $R{:}\,\mathbb{X} \to \{d^R, d^?\}$ określającą przynależność obiektu $x \in \mathbb{X}$ do klasy decyzyjnej $d^R \in \mathbb{D}$. Funkcja ta określona jest przez następujący algorytm:

Algorytm klasyfikacji przy pomocy reprezentanta:

1. Niech: $W_A = W_{max}^R$
2. Niech: $i = 1$
3. Jeśli $a_i(x) \in R^A(i)$ to idź do punktu 5
4. Jeśli $\star \in R^A(i)$ to $W_A := W_A - 1$,
   w przeciwnym przypadku $R(x) := d^?$ i kończymy algorytm
5. Jeśli $i < N$ to $i := i + 1$ i wracamy do punktu 3
6. $R(x) := \begin{cases} d^R & \text{jeżeli } W_A \geqslant 0 \\ d^? & \text{jeżeli } W_A < 0 \end{cases}$

Pominięcie ograniczenia liczby wolnych atrybutów spowodowałoby zbyt duże uogólnienie pociągające za sobą liczne błędy w klasyfikacji.

Aby zilustrować budowę reprezentantów, oraz zasadę ich działania, rozważmy następujący przykład:

Jeżeli podgrupa zawiera następujące reguły:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $r_1$: | 1 | 3 | $\star$ | 1 | $\star$ | $\star$ | 2 | $\Rightarrow$ Decyzja1 |
| $r_2$: | 2 | 3 | $\star$ | $\star$ | 1 | $\star$ | 2 | $\Rightarrow$ Decyzja1 |
| $r_3$: | 5 | 3 | $\star$ | 1 | 1 | $\star$ | 4 | $\Rightarrow$ Decyzja1 |

wówczas reprezentant tej podgrupy będzie miał postać:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| R: | {1,2,5} | 3 | $\star$ | {1,$\star$} | {$\star$,1} | $\star$ | {2,4} | $\Rightarrow$ Decyzja1 wolne atrybuty: $W_{max}^R = 3$ |

Następujące przykłady zostaną sklasyfikowane, przez powyższego reprezentanta, jako obiekty należące do klasy „Decyzja1”:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $x_1$: | 1 | 3 | 3 | 1 | 2 | 3 | 2 |
| $x_2$: | 2 | 3 | 1 | 2 | 1 | 5 | 2 |
| $x_3$: | 2 | 3 | 6 | 1 | 1 | 3 | 4 |
| $x_4$: | 5 | 3 | 4 | 1 | 5 | 3 | 2 |

## 5.6 Klasyfikacja

Klasyfikacja nowego przypadku $x_*$ odbywa się na drodze głosowania reprezentantów. Najpierw jednak musimy wyznaczyć wagi do głosowania. W tym celu trzeba dokonać oceny reguł. Może do tego posłużyć jedna z funkcji opisanych w punkcie 4.1. Następnie wyliczamy wagi, podobnie jak w systemie klasycznym:

$$Waga(R_i) = \frac{Q_K(R_i)}{\sum_{\{R:d^R=d^{R_i}\}} Q_K(R)}$$

Gdzie $Q_K(R)$ jest funkcją oceniającą regułę $R$. Dodatkowo możemy zażądać aby klasa decyzyjna, która uzyskała największe poparcie w głosowaniu miała sumę głosów nie mniejszą niż:

$$\frac{p \cdot \sum_{\{i:R_i(x_*)\neq d^?\}} Waga(R_i)}{|\mathbb{D}|}$$

Gdzie $p \in \langle 1, 2)$ jest pewnym parametrem, możemy przykładowo przyjąć $p = 1.05$. Jeśli powyższy warunek nie jest spełniony wówczas system decyzyjny nie rozpoznaje klasyfikowanego obiektu $x_*$. Dzięki takiemu zabiegowi można ustrzec się nadmiaru błędnie klasyfikowanych przypadków, kosztem zwiększenia liczby obiektów, które nie zostaną sklasyfikowane.

## 5.7 Parametry algorytmu reprezentantów

Przedstawiony system decyzyjny może być dopasowywany do potrzeb analizowanych danych poprzez odpowiedni dobór poniższych elementów algorytmu:

- podział zbioru $\mathbb{X}$ na dwa podzbiory: $\{x_1, \ldots, x_k\}$, oraz $\{x_{k+1}, \ldots, x_{k+m}\}$
- dopuszczalny błąd podczas skracania reguł
- funkcja $Q_R$ oceniająca zachowanie się reguł na nowych danych
- funkcja $Q_W$ oceniająca ogólną skuteczność reguł
- funkcja $d$ mierząca odległość pomiędzy dwoma regułami uwzględniając ich struktury logiczne
- funkcja $Q_K$ oceniająca skuteczność reprezentantów
- parametr $p$ określający wymagane poparcie przy głosowaniu reprezentantów

Znaczenie wymienionych elementów zostało omówione już wcześniej. Doświadczenia pokazały, że niewłaściwy dobór większej liczby parametrów powoduje obniżenie skuteczności systemu lub zwiększenie liczby reprezentantów.

# 6 Wyniki doświadczeń

Chciałbym przedstawić teraz wyniki przeprowadzonych przeze mnie doświadczeń. Wszystkie testowane systemy decyzyjne zostały zaimplementowane zgodnie z podanym powyżej opisem.

Test został wykonany przy użyciu metody CV5 (Cross Validation), polegającej na podziale posiadanej tablicy decyzyjnej na pięć części i wykonaniu pięciu testów opuszczając

za każdym razem inny z wyznaczonych fragmentów. Opuszczone fragmenty tablicy decyzyjnej posłużyły do oceny badanego systemu decyzyjnego. Uzyskane w ten sposób dane, z pięciu doświadczeń, zostały uśrednione i wpisane do poniższych tabel.

Poniżej przedstawiam charakterystykę danych testowych, oraz kompletne wyniki uzyskane przez każdy z opisywanych wyżej systemów. Ze względu na dużą liczbę przedstawionych wyników niezbędne okazało się zestawienie statystyczne, które przedstawiłem na końcu pracy.

## 6.1 Dane testowe

Do przeprowadzenia testów użyłem następujących tablic testowych:

| Nazwa tablicy decyzyjnej | Liczba przykładów | atryb. war. | klas dec. | Średnia liczba wart. atrybutów | Rozkład klas decyzyjnych w tablicy |
|---|---|---|---|---|---|
| Austra0 | 690 | 14 | 2 | 83.4 | 44.5%, 55.5% |
| Austra1 | 345 | 14 | 2 | 55.0 | 56.2%, 43.8% |
| Austra2 | 345 | 14 | 2 | 56.8 | 54.8%, 45.2% |
| Diab0 | 768 | 8 | 2 | 156.4 | 65.1%, 34.9% |
| Diab1 | 384 | 8 | 2 | 111.9 | 65.6%, 34.4% |
| Diab2 | 384 | 8 | 2 | 111.5 | 64.6%, 35.4% |
| Heart0 | 270 | 13 | 2 | 29.5 | 55.6%, 44.4% |
| Heart1 | 135 | 13 | 2 | 22.6 | 56.3%, 43.7% |
| Heart2 | 135 | 13 | 2 | 22.8 | 45.2%, 54.8% |
| Irys | 120 | 4 | 3 | 29.0 | 32.5%, 31.7%, 35.8% |
| Kan 1 | 84 | 16 | 17 | 8.1 | 3.6%,7.1%,7.1%,15.5%, 1.2%,3.6%,2.4%,7.1%, 1.2%,2.4%,2.4%,6.0%, 22.6%,2.4%,1.2%,9.5%, 4.8% |
| Kan 12 | 84 | 17 | 15 | 8.0 | 21.4%,4.8%,42.9%,4.8%, 6.0%,3.6%,1.2%,3.6%, 4.8%,1.2%,1.2%,1.2%, 1.2%,1.2%,1.2% |
| Kan 2 | 84 | 16 | 14 | 7.5 | 25.0%,41.7%,1.2%,2.4%, 3.6%,2.4%,1.2%,4.8%, 6.0%,3.6%,4.8%,1.2%, 1.2%,1.2% |
| Kan 21 | 84 | 17 | 15 | 8.1 | 32.1%,16.7%,6.0%,1.2%, 7.1%,2.4%,1.2%,4.8%, 9.5%,4.8%,3.6%,7.1%, 1.2%,1.2%,1.2% |
| Lymn0 | 148 | 18 | 4 | 3.3 | 54.7%,41.2%,2.7%,1.4% |
| Lymn1 | 74 | 18 | 4 | 3.2 | 37.8%,58.1%,2.7%,1.4% |
| Lymn2 | 74 | 18 | 4 | 3.3 | 44.6%,51.4%,2.7%,1.4% |
| Monk1dat | 124 | 6 | 2 | 2.8 | 50.0%, 50.0% |
| Monk1tes | 432 | 6 | 2 | 2.8 | 50.0%, 50.0% |
| Monk2dat | 169 | 6 | 2 | 2.8 | 37.9%, 62.1% |
| Monk2tes | 432 | 6 | 2 | 2.8 | 67.1%, 32.9% |
| Monk3dat | 122 | 6 | 2 | 2.8 | 50.8%, 49.2% |
| Monk3tes | 432 | 6 | 2 | 2.8 | 52.8%, 47.2% |
| Tttt | 615 | 8 | 2 | 140.1 | 65.2%, 34.8% |

## 6.2 System KNN

Dla systemu KNN przyjąłem $k = 1$.

| Tablica decyzyjna | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| Austra0 | 84.06% | 15.94% | 0.00% | 0.00 | 0.28 s |
| Austra1 | 81.16% | 18.84% | 0.00% | 0.00 | 0.08 s |
| Austra2 | 84.35% | 15.65% | 0.00% | 0.00 | 0.07 s |
| Diab0 | 65.89% | 34.11% | 0.00% | 0.00 | 0.24 s |
| Diab1 | 62.50% | 37.50% | 0.00% | 0.00 | 0.06 s |
| Diab2 | 62.76% | 37.24% | 0.00% | 0.00 | 0.06 s |
| Heart0 | 80.74% | 19.26% | 0.00% | 0.00 | 0.04 s |
| Heart1 | 82.96% | 17.04% | 0.00% | 0.00 | 0.02 s |
| Heart2 | 77.78% | 22.22% | 0.00% | 0.00 | $<$ 0.01 s |
| Irys | 87.50% | 12.50% | 0.00% | 0.00 | $<$ 0.01 s |
| Kan 1 | 25.00% | 75.00% | 0.00% | 0.00 | 0.04 s |
| Kan 12 | 38.10% | 59.52% | 2.38% | 0.00 | 0.02 s |
| Kan 2 | 44.05% | 55.95% | 0.00% | 0.00 | 0.02 s |
| Kan 21 | 34.52% | 65.48% | 0.00% | 0.00 | 0.02 s |
| Lymn0 | 79.73% | 20.27% | 0.00% | 0.00 | 0.18 s |
| Lymn1 | 70.27% | 29.73% | 0.00% | 0.00 | 0.01 s |
| Lymn2 | 77.03% | 22.97% | 0.00% | 0.00 | $<$ 0.01 s |
| Monk1dat | 78.23% | 21.77% | 0.00% | 0.00 | 0.01 s |
| Monk1tes | 99.77% | 0.23% | 0.00% | 0.00 | 0.05 s |
| Monk2dat | 59.17% | 40.83% | 0.00% | 0.00 | 0.02 s |
| Monk2tes | 58.56% | 41.44% | 0.00% | 0.00 | 0.06 s |
| Monk3dat | 81.97% | 18.03% | 0.00% | 0.00 | $<$ 0.01 s |
| Monk3tes | 98.15% | 1.85% | 0.00% | 0.00 | 0.05 s |
| Tttt | 66.67% | 33.33% | 0.00% | 0.00 | 0.16 s |
| średnio | 70.04% | 29.86% | 0.10% | 0.00 | 0.06 s |

## 6.3 System klasyczny

Wyniki systemu klasycznego ze standardowym głosowaniem:

| Tablica decyzyjna | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| Austra0 | 84.78% | 14.78% | 0.43% | 9268.60 | 0.38 s |
| Austra1 | 80.29% | 19.71% | 0.00% | 4482.00 | 0.11 s |
| Austra2 | 86.09% | 13.91% | 0.00% | 4082.80 | 0.09 s |
| Diab0 | 67.58% | 32.29% | 0.13% | 6651.60 | 0.23 s |
| Diab1 | 64.58% | 35.42% | 0.00% | 2987.40 | 0.02 s |
| Diab2 | 64.84% | 34.90% | 0.26% | 2894.60 | 0.04 s |
| Heart0 | 80.37% | 19.63% | 0.00% | 4027.40 | 0.10 s |
| Heart1 | 81.48% | 18.52% | 0.00% | 1623.40 | 0.02 s |
| Heart2 | 78.52% | 21.48% | 0.00% | 1724.00 | $<$ 0.01 s |
| Irys | 87.50% | 11.67% | 0.83% | 169.00 | 0.01 s |
| Kan 1 | 20.24% | 79.76% | 0.00% | 3118.00 | 0.04 s |
| Kan 12 | 27.38% | 72.62% | 0.00% | 3310.60 | 0.02 s |
| Kan 2 | 34.52% | 65.48% | 0.00% | 2423.80 | 0.04 s |
| Kan 21 | 32.14% | 67.86% | 0.00% | 3777.80 | 0.02 s |
| Lymn0 | 81.76% | 18.24% | 0.00% | 5324.80 | 0.12 s |
| Lymn1 | 75.68% | 24.32% | 0.00% | 2167.20 | 0.02 s |
| Lymn2 | 71.62% | 28.38% | 0.00% | 2494.00 | 0.02 s |
| Monk1dat | 81.45% | 18.55% | 0.00% | 155.60 | $<$ 0.01 s |
| Monk1tes | 100.00% | 0.00% | 0.00% | 73.80 | $<$ 0.01 s |
| Monk2dat | 61.54% | 38.46% | 0.00% | 231.00 | $<$ 0.01 s |
| Monk2tes | 50.69% | 49.31% | 0.00% | 208.80 | 0.01 s |
| Monk3dat | 88.52% | 11.48% | 0.00% | 133.20 | $<$ 0.01 s |
| Monk3tes | 99.77% | 0.23% | 0.00% | 43.40 | $<$ 0.01 s |
| Tttt | 68.13% | 31.54% | 0.33% | 5168.40 | 0.12 s |
| średnio | 69.56% | 30.36% | 0.08% | 2772.55 | 0.06 s |

Poniżej przedstawiam wyniki systemu klasycznego opartego na skróconych regułach. Maksymalny dopuszczalny błąd dla każdej skracanej reguły wynosi 20%.

| Tablica decyzyjna | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| Austra0 | 84.06% | 15.94% | 0.00% | 6552.00 | 0.29 s |
| Austra1 | 82.90% | 17.10% | 0.00% | 3345.40 | 0.09 s |
| Austra2 | 86.09% | 13.91% | 0.00% | 2836.60 | 0.07 s |
| Diab0 | 67.45% | 32.42% | 0.13% | 5218.40 | 0.18 s |
| Diab1 | 66.67% | 33.33% | 0.00% | 2432.00 | 0.03 s |
| Diab2 | 63.80% | 36.20% | 0.00% | 2342.20 | 0.03 s |
| Heart0 | 82.96% | 17.04% | 0.00% | 2873.40 | 0.05 s |
| Heart1 | 79.26% | 20.74% | 0.00% | 1159.40 | 0.01 s |
| Heart2 | 81.48% | 18.52% | 0.00% | 1358.60 | $<$ 0.01 s |
| Irys | 89.17% | 10.83% | 0.00% | 143.80 | $<$ 0.01 s |
| Kan 1 | 20.24% | 79.76% | 0.00% | 3089.40 | 0.05 s |
| Kan 12 | 35.71% | 64.29% | 0.00% | 3086.80 | 0.04 s |
| Kan 2 | 35.71% | 64.29% | 0.00% | 2251.60 | 0.03 s |
| Kan 21 | 33.33% | 66.67% | 0.00% | 3667.20 | 0.05 s |
| Lymn0 | 77.70% | 22.30% | 0.00% | 3160.40 | 0.10 s |
| Lymn1 | 81.08% | 18.92% | 0.00% | 1349.20 | $<$ 0.01 s |
| Lymn2 | 79.73% | 20.27% | 0.00% | 1693.60 | $<$ 0.01 s |
| Monk1dat | 79.84% | 20.16% | 0.00% | 100.20 | $<$ 0.01 s |
| Monk1tes | 100.00% | 0.00% | 0.00% | 61.00 | $<$ 0.01 s |
| Monk2dat | 60.95% | 39.05% | 0.00% | 154.00 | $<$ 0.01 s |
| Monk2tes | 62.96% | 37.04% | 0.00% | 163.00 | $<$ 0.01 s |
| Monk3dat | 89.34% | 10.66% | 0.00% | 79.40 | $<$ 0.01 s |
| Monk3tes | 97.22% | 2.78% | 0.00% | 32.00 | $<$ 0.01 s |
| Tttt | 68.29% | 31.54% | 0.16% | 4103.20 | 0.13 s |
| średnio | 71.08% | 28.91% | 0.01% | 2135.53 | 0.05 s |

Jak możemy zaobserwować, skuteczność systemu po skróceniu reguł, wzrosła. Każda pojedyncza reguła popełnia teraz więcej błędów, ale dzięki ich dużej liczbie i stosowanemu przy klasyfikacji głosowaniu efekt ten nie jest odczuwalny. Jednocześnie wszystkie reguły uległy uogólnieniu dzięki czemu zmniejszyła się liczba nierozpoznawanych obiektów. Skrócenie reguł w naturalny sposób spowodowało zmniejszenie ich liczebności.

## 6.4 Systemy QbF

W wykonanych przeze mnie doświadczeniach po ocenie reguł wybrano 20 najlepszych, a następnie dodawano kolejne reguły, które zostały ocenione tak samo jak najsłabsza reguła z pierwszej dwudziestki. Jako górne ograniczenie liczby reguł przyjąłem: max(60,33% wszystkich reguł). W testach brał udział również system QbF 13 oparty na funkcji $Q_{13}$.Działa on tutaj następująco: wybiera największą dopuszczalną liczbę reguł (dowolnych) i używa ich (tak jak pozostałe systemy QbF) do klasyfikacji. Poniżej przedstawiam średnie wyniki uzyskane przez wszystkie omawiane systemy QbF:

| System decyzyjny | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| QbF 1 | 39.18% | 16.91% | 43.91% | 42.75 | 0.02 s |
| QbF 2 | 39.02% | 16.97% | 44.02% | 42.61 | 0.04 s |
| QbF 3 | 39.11% | 16.96% | 43.94% | 42.70 | 0.04 s |
| QbF 4 | 27.07% | 18.01% | 54.92% | 48.24 | 0.05 s |
| QbF 5 | 49.12% | 13.73% | 37.15% | 24.52 | 0.06 s |
| QbF 6 | 49.12% | 13.73% | 37.15% | 24.52 | 0.06 s |
| QbF 7 | 23.63% | 19.79% | 56.59% | 67.47 | 0.06 s |
| QbF 8 | 57.41% | 32.20% | 10.39% | 590.84 | 0.08 s |
| QbF 9 | 24.46% | 19.62% | 55.92% | 68.97 | 0.07 s |
| QbF 10 | 38.67% | 16.94% | 44.39% | 42.98 | 0.08 s |
| QbF 11 | 29.28% | 34.89% | 35.83% | 427.73 | 0.08 s |
| QbF 12 | 53.37% | 14.02% | 32.61% | 25.25 | 0.09 s |
| QbF 13 | 52.26% | 35.73% | 12.01% | 590.92 | 0.10 s |
| średnio | 38.76% | 21.44% | 39.80% | 149.36 | 0.06 s |

## 6.5 System QbF z doborem miary

Poniżej przedstawiam wyniki systemu QbF dobierającego funkcję oceniającą reguły w zależności od danych:

| Tablica decyzyjna | Przypadki rozpoznane prawidłowo | błędnie | nierozpoznane | Liczba reguł | średni czas klasyfikacji | System QbF |
|---|---|---|---|---|---|---|
| Austra0 | 80.58% | 18.12% | 1.30% | 2015.00 | 0.07 s | QbF 13 |
| Austra1 | 79.42% | 20.58% | 0.00% | 983.20 | $<$ 0.01 s | QbF 13 |
| Austra2 | 81.45% | 16.81% | 1.74% | 895.20 | $<$ 0.01 s | QbF 8 |
| Diab0 | 48.44% | 28.65% | 22.92% | 1442.00 | 0.01 s | QbF 13 |
| Diab1 | 26.56% | 10.68% | 62.76% | 29.40 | $<$ 0.01 s | QbF 12 |
| Diab2 | 44.27% | 26.04% | 29.69% | 612.00 | 0.03 s | QbF 13 |
| Heart0 | 74.44% | 23.70% | 1.85% | 838.60 | $<$ 0.01 s | QbF 8 |
| Heart1 | 66.67% | 13.33% | 20.00% | 21.80 | $<$ 0.01 s | QbF 12 |
| Heart2 | 68.15% | 14.07% | 17.78% | 22.80 | $<$ 0.01 s | QbF 12 |
| Irys | 71.67% | 6.67% | 21.67% | 26.60 | $<$ 0.01 s | QbF 12 |
| Kan 1 | 9.52% | 48.81% | 41.67% | 26.20 | 0.27 s | QbF 5 |
| Kan 12 | 33.33% | 22.62% | 44.05% | 23.00 | 0.23 s | QbF 12 |
| Kan 2 | 35.71% | 20.24% | 44.05% | 27.20 | 0.16 s | QbF 12 |
| Kan 21 | 23.81% | 41.67% | 34.52% | 43.60 | 0.30 s | QbF 12 |
| Lymn0 | 66.89% | 11.49% | 21.62% | 24.80 | 1.02 s | QbF 12 |
| Lymn1 | 67.57% | 16.22% | 16.22% | 22.80 | 0.11 s | QbF 5 |
| Lymn2 | 74.32% | 14.86% | 10.81% | 23.80 | 0.11 s | QbF 5 |
| Monk1dat | 79.03% | 4.03% | 16.94% | 22.20 | $<$ 0.01 s | QbF 5 |
| Monk1tes | 93.52% | 0.00% | 6.48% | 21.80 | $<$ 0.01 s | QbF 12 |
| Monk2dat | 56.80% | 20.71% | 22.49% | 24.00 | $<$ 0.01 s | QbF 12 |
| Monk2tes | 40.51% | 2.55% | 56.94% | 20.80 | $<$ 0.01 s | QbF 1 |
| Monk3dat | 80.33% | 6.56% | 13.11% | 21.00 | $<$ 0.01 s | QbF 5 |
| Monk3tes | 98.84% | 1.16% | 0.00% | 20.80 | $<$ 0.01 s | QbF 5 |
| Tttt | 41.95% | 27.48% | 30.57% | 1128.20 | 0.02 s | QbF 8 |
| średnio | 60.16% | 17.38% | 22.47% | 347.37 | 0.10 s | – |

Jak widać średnia skuteczność tego systemu znacznie przewyższa skuteczność systemów QbF opartych tylko na jednej funkcji. Przykład ten pokazuje jak dużą rolę odgrywa umiejętny dobór stosowanych metod w zależności od analizowanych danych.

## 6.6 System QbF ze skracaniem reguł

Poniższa tabela prezentuje wyniki systemu QbF dobierającego funkcję oceniającą i opartego na skróconych regułach. Maksymalny dopuszczalny błąd dla każdej skracanej reguły wynosi 20%.

| Tablica decyzyjna | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji | System QbF |
|---|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | | |
| Austra0 | 84.20% | 12.61% | 3.19% | 24.00 | 0.01 s | QbF 4 |
| Austra1 | 80.58% | 12.46% | 6.96% | 21.80 | $<$ 0.01 s | QbF 10 |
| Austra2 | 85.51% | 12.75% | 1.74% | 27.80 | $<$ 0.01 s | QbF 4 |
| Diab0 | 42.71% | 15.36% | 41.93% | 20.80 | 0.01 s | QbF 12 |
| Diab1 | 42.97% | 17.19% | 39.84% | 25.20 | $<$ 0.01 s | QbF 12 |
| Diab2 | 36.46% | 14.84% | 48.70% | 29.00 | $<$ 0.01 s | QbF 5 |
| Heart0 | 82.96% | 17.04% | 0.00% | 20.20 | $<$ 0.01 s | QbF 1 |
| Heart1 | 85.19% | 13.33% | 1.48% | 20.20 | 0.01 s | QbF 2 |
| Heart2 | 82.22% | 17.78% | 0.00% | 20.60 | $<$ 0.01 s | QbF 3 |
| Irys | 71.67% | 7.50% | 20.83% | 24.80 | 0.03 s | QbF 5 |
| Kan 1 | 9.52% | 47.62% | 42.86% | 26.00 | 0.33 s | QbF 5 |
| Kan 12 | 39.29% | 36.90% | 23.81% | 28.20 | 0.21 s | QbF 12 |
| Kan 2 | 35.71% | 33.33% | 30.95% | 25.00 | 0.16 s | QbF 12 |
| Kan 21 | 23.81% | 35.71% | 40.48% | 22.60 | 0.26 s | QbF 12 |
| Lymn0 | 75.68% | 18.92% | 5.41% | 25.00 | 0.68 s | QbF 5 |
| Lymn1 | 75.68% | 24.32% | 0.00% | 22.40 | 0.08 s | QbF 12 |
| Lymn2 | 74.32% | 13.51% | 12.16% | 22.60 | 0.13 s | QbF 5 |
| Monk1dat | 81.45% | 6.45% | 12.10% | 21.60 | $<$ 0.01 s | QbF 5 |
| Monk1tes | 95.37% | 0.00% | 4.63% | 20.20 | $<$ 0.01 s | QbF 5 |
| Monk2dat | 52.07% | 24.85% | 23.08% | 31.20 | $<$ 0.01 s | QbF 1 |
| Monk2tes | 56.94% | 30.79% | 12.27% | 20.60 | $<$ 0.01 s | QbF 12 |
| Monk3dat | 88.52% | 9.84% | 1.64% | 21.00 | $<$ 0.01 s | QbF 5 |
| Monk3tes | 97.22% | 2.78% | 0.00% | 20.00 | $<$ 0.01 s | QbF 5 |
| Tttt | 37.89% | 14.15% | 47.97% | 23.40 | $<$ 0.01 s | QbF 12 |
| średnio | 64.08% | 18.33% | 17.58% | 23.51 | 0.08 s | – |

Jak widać rozszerzenie systemu QbF o dobór miary i skracanie reguł daje doskonałe rezultaty. Nie tylko wzrosła skuteczność systemu, lecz również znacznie zmalała liczba reguł.

## 6.7 System reprezentantów

W punkcie tym pragnę zaprezentować wyniki systemu reprezentantów. Do każdej tablicy decyzyjnej zastały dobrane inne parametry wymienione w punkcie 5.7.

| Tablica decyzyjna | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| Austra0 | 85.51% | 14.49% | 0.00% | 24.80 | 0.01 s |
| Austra1 | 80.00% | 20.00% | 0.00% | 30.60 | $<$ 0.01 s |
| Austra2 | 87.25% | 12.75% | 0.00% | 26.40 | $<$ 0.01 s |
| Diab0 | 66.02% | 33.98% | 0.00% | 146.60 | $<$ 0.01 s |
| Diab1 | 62.76% | 37.24% | 0.00% | 90.40 | $<$ 0.01 s |
| Diab2 | 67.45% | 32.55% | 0.00% | 77.80 | 0.02 s |
| Heart0 | 83.70% | 16.30% | 0.00% | 13.40 | $<$ 0.01 s |
| Heart1 | 80.00% | 20.00% | 0.00% | 15.40 | $<$ 0.01 s |
| Heart2 | 75.56% | 24.44% | 0.00% | 11.00 | $<$ 0.01 s |
| Irys | 92.50% | 7.50% | 0.00% | 24.80 | 0.07 s |
| Kan 1 | 14.29% | 83.33% | 2.38% | 150.00 | $<$ 0.01 s |
| Kan 12 | 39.29% | 60.71% | 0.00% | 116.60 | 0.01 s |
| Kan 2 | 40.48% | 59.52% | 0.00% | 100.00 | $<$ 0.01 s |
| Kan 21 | 33.33% | 66.67% | 0.00% | 127.40 | $<$ 0.01 s |
| Lymn0 | 81.76% | 18.24% | 0.00% | 49.40 | 0.16 s |
| Lymn1 | 81.08% | 18.92% | 0.00% | 44.00 | $<$ 0.01 s |
| Lymn2 | 83.78% | 16.22% | 0.00% | 30.00 | 0.06 s |
| Monk1dat | 90.32% | 9.68% | 0.00% | 59.00 | $<$ 0.01 s |
| Monk1tes | 100.00% | 0.00% | 0.00% | 49.40 | 0.01 s |
| Monk2dat | 68.64% | 28.40% | 2.96% | 67.20 | $<$ 0.01 s |
| Monk2tes | 66.20% | 33.80% | 0.00% | 22.60 | $<$ 0.01 s |
| Monk3dat | 93.44% | 6.56% | 0.00% | 27.60 | $<$ 0.01 s |
| Monk3tes | 97.22% | 2.78% | 0.00% | 13.20 | $<$ 0.01 s |
| Tttt | 65.69% | 34.31% | 0.00% | 45.40 | $<$ 0.01 s |
| średnio | 72.34% | 27.43% | 0.22% | 56.79 | 0.02 s |

Prezentowany system reprezentantów podobnie jak system klasyczny, charakteryzuje się niewielką liczbą nierozpoznanych obiektów. Oba te systemy popełniają jednak więcej błędów niż system QbF wzbogacony o dobór miary i skracanie reguł.

## 6.8 Podsumowanie wyników doświadczeń

W kolejnych punktach chciałbym przedstawić zestawienia statystyczne przedstawionych wyników doświadczeń. Pomogą one w porównaniu różnych systemów decyzyjnych.

### 6.8.1 Średnie wyniki dla tablic decyzyjnych

| Tablica decyzyjna | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| Austra0 | 55.80% | 11.41% | 32.79% | 1209.00 | 0.06 s |
| Austra1 | 60.19% | 14.61% | 25.20% | 603.56 | 0.02 s |
| Austra2 | 61.84% | 12.46% | 25.70% | 540.70 | 0.02 s |
| Diab0 | 30.48% | 16.49% | 53.03% | 936.27 | 0.04 s |
| Diab1 | 29.84% | 19.48% | 50.68% | 410.79 | 0.01 s |
| Diab2 | 31.72% | 18.89% | 49.39% | 411.83 | 0.01 s |
| Heart0 | 59.41% | 16.91% | 23.68% | 527.23 | 0.01 s |
| Heart1 | 62.52% | 17.67% | 19.82% | 207.08 | $<$ 0.01 s |
| Heart2 | 63.11% | 20.96% | 15.93% | 227.18 | $<$ 0.01 s |
| Irys | 63.83% | 7.96% | 28.21% | 39.88 | 0.02 s |
| Kan 1 | 8.15% | 59.76% | 32.08% | 447.83 | 0.15 s |
| Kan 12 | 17.68% | 47.56% | 34.76% | 436.49 | 0.11 s |
| Kan 2 | 19.58% | 51.07% | 29.35% | 337.73 | 0.08 s |
| Kan 21 | 14.23% | 48.99% | 36.78% | 514.24 | 0.15 s |
| Lymn0 | 42.43% | 19.05% | 38.51% | 569.93 | 0.56 s |
| Lymn1 | 45.41% | 23.24% | 31.35% | 253.45 | 0.05 s |
| Lymn2 | 44.80% | 29.59% | 25.61% | 302.27 | 0.07 s |
| Monk1dat | 69.44% | 16.81% | 13.75% | 39.11 | $<$ 0.01 s |
| Monk1tes | 84.50% | 4.43% | 11.06% | 29.47 | $<$ 0.01 s |
| Monk2dat | 44.32% | 29.35% | 26.33% | 49.07 | $<$ 0.01 s |
| Monk2tes | 38.17% | 21.88% | 39.95% | 46.72 | $<$ 0.01 s |
| Monk3dat | 73.89% | 12.58% | 13.52% | 32.30 | $<$ 0.01 s |
| Monk3tes | 88.65% | 4.83% | 6.53% | 21.83 | $<$ 0.01 s |
| Tttt | 29.97% | 16.91% | 53.12% | 718.16 | 0.03 s |

### 6.8.2 Średnia skuteczność przedstawionych systemów decyzyjnych

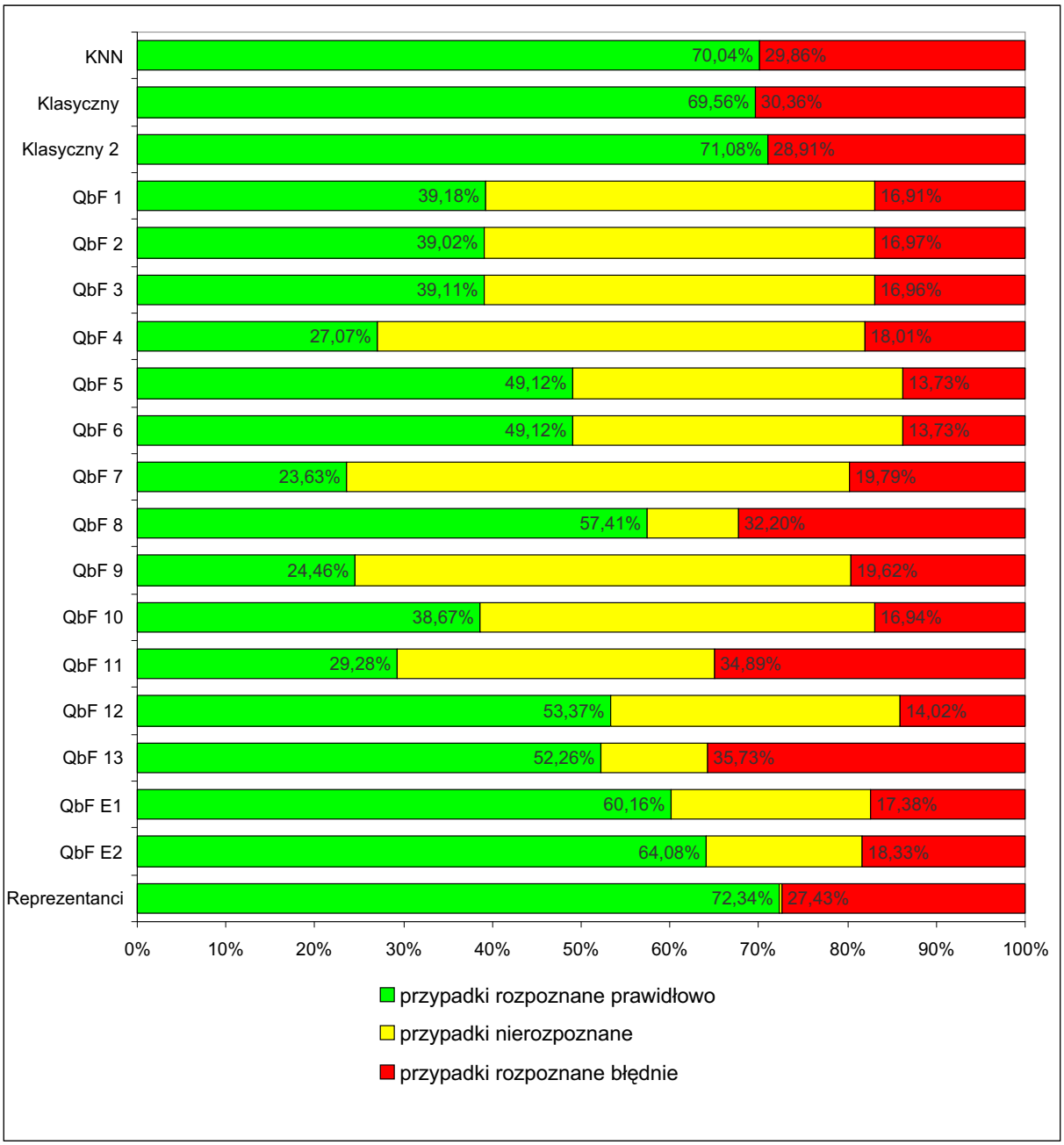
| System decyzyjny | Przypadki rozpoznane | | | Liczba reguł | średni czas klasyfikacji |
|---|---|---|---|---|---|
| | prawidłowo | błędnie | nierozpoznane | | |
| KNN | 70.04% | 29.86% | 0.10% | 0.00 | 0.06 s |
| Klasyczny | 69.56% | 30.36% | 0.08% | 2772.55 | 0.06 s |
| Klasyczny 2 | 71.08% | 28.91% | 0.01% | 2135.53 | 0.05 s |
| QbF 1 | 39.18% | 16.91% | 43.91% | 42.75 | 0.02 s |
| QbF 2 | 39.02% | 16.97% | 44.02% | 42.61 | 0.04 s |
| QbF 3 | 39.11% | 16.96% | 43.94% | 42.70 | 0.04 s |
| QbF 4 | 27.07% | 18.01% | 54.92% | 48.24 | 0.05 s |
| QbF 5 | 49.12% | 13.73% | 37.15% | 24.52 | 0.06 s |
| QbF 6 | 49.12% | 13.73% | 37.15% | 24.52 | 0.06 s |
| QbF 7 | 23.63% | 19.79% | 56.59% | 67.47 | 0.06 s |
| QbF 8 | 57.41% | 32.20% | 10.39% | 590.84 | 0.08 s |
| QbF 9 | 24.46% | 19.62% | 55.92% | 68.97 | 0.07 s |
| QbF 10 | 38.67% | 16.94% | 44.39% | 42.98 | 0.08 s |
| QbF 11 | 29.28% | 34.89% | 35.83% | 427.73 | 0.08 s |
| QbF 12 | 53.37% | 14.02% | 32.61% | 25.25 | 0.09 s |
| QbF 13 | 52.26% | 35.73% | 12.01% | 590.92 | 0.10 s |
| QbF E1 | 60.16% | 17.38% | 22.47% | 347.37 | 0.10 s |
| QbF E2 | 64.08% | 18.33% | 17.58% | 23.51 | 0.08 s |
| Reprezentanci | 72.34% | 27.43% | 0.22% | 56.79 | 0.02 s |

Przyjęte oznaczenia systemów:

- KNN – system KNN dla $k = 1$ i najprostszej miary odległości
- Klasyczny – system klasyczny z prostym głosowaniem
- Klasyczny 2 – system klasyczny oparty na skróconych regułach
- QbF n – system QbF używający do selekcji reguł funkcji $Q_n$
- QbF E1 – system QbF rozbudowany o dobór funkcji $Q_n$
- QbF E2 – system QbF wzbogacony o dobór funkcji oceniającej i skracanie reguł
- Reprezentanci – system reprezentantów przedstawiony w rozdziale 5

### 6.8.3 Reprezentacja graficzna

Poniżej przedstawiam reprezentację graficzną uzyskanych wyników. Taka forma prezentacji ułatwia porównanie przedstawionych metod.



# 7 Podsumowanie i wnioski

Przedstawione wyniki doświadczeń pokazują, że zmniejszenie liczby reguł decyzyjnych nie zawsze wiąże się z utratą jakości klasyfikacji. Łatwo też zauważyć, że skracanie reguł połączone ze znanymi już metodami również daje doskonałe rezultaty.

Zaproponowany system reprezentantów może z powodzeniem konkurować ze wszystkimi przedstawionymi metodami. Dla wielu tablic decyzyjnych uzyskał on najlepsze rezultaty. Jego dodatkowym atutem jest możliwość adaptowania go do różnego rodzaju danych poprzez odpowiedni dobór parametrów wymienionych w punkcie 5.7.

System reprezentantów jest oparty o bardziej rozbudowane reguły niż systemy klasyczne. Pozwala to na ogólniejszy opis aproksymowanych pojęć przy jednoczesnym zachowaniu jego wysokiej jakości.

Problem redukcji i łączenia reguł jest bardzo obszerny i trudno go wyczerpać w ramach pracy magisterskiej. Mam jednak nadzieję, że moja praca i uzyskane wyniki przyczynią się do dalszych badań w tym zakresie.

# Bibliografia


[1] Thomas Ågotnes, „Filtering large propositional rule sets while retaining classifier performance", Department of Computer and Information Science, Norwegian University of Science and Technology, 1999

[2] Jan G. Bazan, „Metody wnioskowań aproksymacyjnych dla syntezy algorytmów decyzyjnych", praca doktorska, Instytut Matematyki UW, 1998

[3] Jan G. Bazan, Andrzej Skowron, Piotr Synak, „Dynamic reducts as a tool for extracting laws from decision tables", Proc. International Symposium on Methodologies for Intelligent Systems, Lecture Notes in Artificial Intelligence Vol.869, Springer-Verlag, 1994, 346–355

[4] I. Bruha, „Quality of decision rules", Machine Learning and Statistics (The Interface, rozdział 5.), 1997

[5] R. C. Holte, „Very simple classification rules perform well on most commonly used datasets", Machine Learning 1993, 11:63-91

[6] Huan Liu, Hireshi Motoda, „Feature extraction construction and selection a data mining perspective", Kluwer Academic Publishers, 1998

[7] Huan Liu, Hireshi Motoda, „Feature selection for knowlege discovery and data mining", Kluwer Academic Publishers, 1998

[8] D. Michie, D.J. Spiegelhalter, C.C. Taylor, „Machine learning, neural and statistical classification", Ellis Horwood series in Artificial Intelligence, 1994

[9] T. Mitchell, „Machine Learning", McGraw-Hill New York, 1997

[10] Hoa S. Nguyen, „Data regularity analysis and applications in data mining", Uniwersytet Warszawski, 1999

[11] Z. Pawlak, „Rough sets – Theoretical aspects of reasoning about data", Kluwer Academic Publishers, Dordrecht, 1991

[12] L. Polkowski, A. Skowron, „Rough sets in knowledge discovery", Physica Verlag Vol. 1-2, Heidelberg, 1998

[13] J. Rissanen, „Modeling by the shortest data description", Authomatica 14, 1978, 465-471